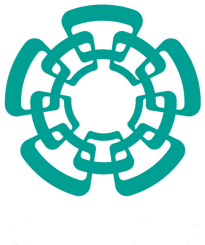


Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Cinvestav Tamaulipas

Marco de gestión de flujos de trabajo basado en bloques encadenados para la verificación continua de datos en escenarios del Internet de las Cosas

Tesis que presenta:

Cristhian Martínez Rendón

Para obtener el grado de:

**Maestro en Ciencias
en Ingeniería y Tecnologías
Computacionales**

Dr. José Luis González Compeán, Director
Dr. Hiram Galeana Zapién, Co-Director

Cd. Victoria, Tamaulipas, México. Julio, 2019

La tesis presentada por Cristhian Martínez Rendón fue aprobada por:

_____

Dr. Iván López Arévalo

Dr. Miguel Morales Sandoval

Dr. José Luis González Compeán, Director

Dr. Hiram Galeana Zapién, Co-Director

Cd. Victoria, Tamaulipas, México., 15 de Julio de 2019

A mi Madre.

Agradecimientos

- Primero que todo, a Dios gracias por la vida, la salud y demás bendiciones dadas que me permitieron realizar un estudio de Maestría.
- Gracias a mis padres por brindarme la mejor educación posible y dotarme de los mejores valores para enfrentar la vida y sus retos. A mi madre por su constante apoyo, entrega y gran amor, por ser mi motivación para crecer profesionalmente y como persona cada día más.
- A mi familia, por todo el apoyo brindado, los consejos dados y su preocupación a la distancia. Especialmente a Maria Alejandra por su constante motivación, apoyo y amor.
- A mis amigos en Colombia que a pesar de la distancia me dan su vos de aliento. A los nuevos amigos en México por hacer de mi estancia más agradable y divertida, por las experiencias vividas y sus enseñanzas.
- A mis asesores Dr. José Luis González Compeán y Dr. Hiram Galeana Zapién por su paciencia, tolerancia, conocimiento, dirección y gran apoyo durante la realización de este trabajo de tesis.
- A mis revisores Dr. Iván López Arévalo y Dr. Miguel Morales Sandoval por sus enseñanzas y apoyo para la culminación de este trabajo de tesis.
- A mis compañeros de Maestría por las experiencias vividas y su gran ayuda en mi estadía.
- Al personal administrativo del Cinvestav-Tamaulipas por su ayuda durante mi estadía.
- Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por la beca otorgada durante los dos años de maestría.
- Al Cinvestav-Tamaulipas por recibirme en su Institución y brindarme una formación profesional de calidad junto con todas las garantías para culminar satisfactoriamente mis estudios.

Índice General

Índice de Figuras

Índice de Tablas

Índice de Algoritmos

Publicaciones

Resumen

Marco de gestión de flujos de trabajo basado en bloques encadenados para la verificación continua de datos en escenarios del Internet de las Cosas

por

Cristhian Martínez Rendón
Unidad Cinvestav Tamaulipas
Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, 2019
Dr. José Luis González Compeán, Director
Dr. Hiram Galeana Zapién, Co-Director

En escenarios del Internet de las Cosas (IoT, por sus siglas en inglés) se recolectan grandes volúmenes de datos mediante diferentes tipos de sensores. El procesamiento de dichos datos comúnmente se realiza mediante esquemas de procesamiento llamados flujos de trabajo, los cuales permiten procesar datos IoT sin intervención humana. Estas estructuras de procesamiento incluyen aplicaciones que se ejecutan en etapas tales como adquisición, transformación y análisis de datos, así como la exhibición de resultados que son útiles para procesos de toma de decisiones. Los flujos de trabajo se construyen con la participación de diversas organizaciones y/o usuarios. Por tanto, resulta crucial que los participantes puedan verificar y validar que el resultado de un flujo de trabajo sea obtenido por el procesamiento de cada uno de sus componentes en la estricta secuencia previamente establecida por los participantes. Además, para llevar a cabo, con certidumbre, procesos de toma de decisiones resulta clave validar que los datos IoT entrantes y salientes de cada etapa de un flujo de trabajo no han sido alterados por terceros o por los usuarios de las organizaciones participantes. Sin embargo, realizar este tipo de verificación en flujos de trabajo desplegados en entornos de nube, implica la integración de dichos flujos con las redes de verificabilidad existentes, lo cual no es trivial y llega a convertirse en un desafío para las organizaciones. En estos escenarios no existe garantía de que las aplicaciones ejecutadas por los participantes en un flujo de trabajo realicen el registro de sus transacciones en la red de verificabilidad. Además, los problemas de eficiencia que aparecen cuando

se realiza una verificabilidad continua en flujos de trabajo sigue siendo un asunto poco abordado en la literatura. En este contexto, en el presente proyecto de tesis se propone un esquema llamado CD/CV (acrónimo para continuous delivery, continuous verifiability) para la verificabilidad de entrega continua de datos IoT en flujos de trabajo. Este esquema se basa en un modelo de extracción, transformación y carga (ETC), el cual permite captura cada acción realizada por cada etapa incluida en flujos de trabajo. El esquema CD/CV se implementó en la forma de un marco de gestión para la verificabilidad continua en flujos de trabajo. Este marco de gestión incluye un motor de entrega continua (CD) para la creación y manejo de flujos de trabajo, así como un esquema de verificabilidad (CV) para crear y desplegar redes de verificabilidad en forma automática. En tiempo de operación, el gestor realiza tares de intermediario entre el flujo de trabajo y la red de verificabilidad. Específicamente, el gestor captura la información ETC de cada acción realizada por cada etapa de un flujo de trabajo y crea transacciones que son automáticamente registradas en la red de verificabilidad. La implementación del gestor se basa en el uso de patrones de paralelismo encapsulados en contenedores virtuales, los cuales no solo resuelven problemas de portabilidad sino que también permiten reducir el impacto de las acciones de verificabilidad en el rendimiento de los flujos de trabajo. Una evaluación experimental fue conducida en la forma de estudios de caso basados en datos sobre movilidad de usuarios y datos medioambientales capturados por sensores. Los estudios de caso revelaron la factibilidad de aplicar el marco de gestión propuesto a los procesos de integración entre flujos de trabajo y redes de verificabilidad (Blockchain por su acepción en ingles). La evaluación también reveló que el marco de gestión puede reducir e incluso eliminar la sobrecarga de los procesos de verificabilidad continua en comparación con soluciones del estado del arte.

Abstract

Management Framework of workflows based on Blockchain for continuous verification of data in Internet of Things scenarios

by

Cristhian Martínez Rendón

Cinvestav Tamaulipas

Center for Research and Advanced Studies of the National Polytechnic Institute, 2019

Dr. José Luis González Compeán, Co-advisor

Dr. Hiram Galeana Zapién, Co-advisor

In the Internet of Things (IoT) scenarios, large volumes of data are collected using different types of sensors. The processing of that data is commonly done through processing schemes called workflows, which allow processing IoT data without human intervention. These processing structures include applications that run in stages such as acquisition, transformation, and data analysis, as well as the display of results that are useful for decision-making processes. Workflows are built with the participation of various organizations and/or users. Therefore, it is crucial that participants can verify and validate that the result of a workflow is obtained by processing each of its components in the strict sequence previously established by the participants. Besides, to carry out, with certainty, decision-making processes, it is important to validate that the incoming and outgoing IoT data of each stage of a workflow have not been altered by third parties or by the users of the participating organizations. However, performing this type of verification in workflows deployed in cloud environments implies the integration of these flows with existing verifiability networks, which is not trivial and becomes a challenge for organizations. In these scenarios, there is no guarantee that the applications executed by the participants in a workflow register their transactions in the verifiability network. Also, the efficiency problems that appear when continuous verifiability is made in workflows continues to be an issue that has not been addressed in the literature. In this context, we propose in the present thesis project a scheme called CD/CV (an acronym for continuous delivery, continuous verifiability) for the verifiability of continuous delivery of IoT data in workflows. This scheme is based

on an extraction, transformation, and loading (ETC) model, which allows capturing every action performed by each stage included in workflows. The CD/CV scheme was implemented in the form of a management framework for continuous verifiability in workflows. This management framework includes a continuous delivery engine (CD) for the creation and management of workflows, as well as a verifiability scheme (CV) to automatically create and deploy verifiability networks. At the time of operation, the manager performs intermediary tasks between the workflow and the verifiability network. Specifically, the manager captures the ETC information of each action performed by each stage of a workflow and creates transactions that are automatically registered in the verifiability network. The implementation of the manager is based on the use of parallelism patterns encapsulated in virtual containers, which not only solve portability problems but also reduce the impact of verifiability actions on the workflow performance. An experimental evaluation was conducted in the form of case studies based on data on user mobility and environmental data captured by sensors. The case studies revealed the feasibility of applying the proposed management framework to the processes of integration between workflows and verifiability networks or Blockchain. The evaluation also revealed that the management framework can reduce or even eliminate the overload of continuous verifiability processes compared to state-of-the-art solutions.

Nomenclatura

BC	Bloques Encadenados (Blockchain)
ETC	Extracción, Transformación y Entrega
IoT	Internet de las Cosas
MG-CD	Marco de Gestión de Entrega Continua
MG-CD/CV	Marco de Gestión de Entrega Continua y Verificación Continua
POI	Punto de interés

1

Introducción

En este capítulo se presentan los antecedentes, motivación y definición del problema de investigación abordado en la tesis. Asimismo, se definen los objetivos de la tesis y la metodología de investigación.

1.1 Antecedentes y motivación

El Internet de las Cosas (IoT, por sus siglas en inglés) es un nuevo paradigma en el cual objetos físicos con capacidades de cómputo, comunicación y sensado podrán recabar y transmitir a Internet información de variables físicas de su entorno. Existen diferentes casos de uso o áreas de aplicación de IoT, como es el caso de salud, transporte, ciudades inteligentes, medioambientales, etc.

La recolección de información en los diversos casos de uso de IoT permitirá que, a partir del análisis de los grandes volúmenes de datos recabados[1], se generen sistemas de apoyo a la decisión basados

[1]La cantidad total de datos creados (y no necesariamente almacenados) por cualquier dispositivo alcanzará los 847 ZB por año para 2021, frente a los 218 ZB por año en 2016 [57].

en el conocimiento extraído de los datos recolectados [23, 44, 45]. A manera de ejemplo, en sistemas de transporte terrestre, en donde las unidades estén equipadas con sensores GPS, la recolección y análisis de patrones de movilidad de los vehículos puede emplearse para gestionar de forma eficiente el tráfico vehicular mediante reconfiguración de semáforos en las avenidas, recomendación de rutas adecuadas entre puntos de interés, etc. [1, 11, 47].

Desde una perspectiva de arquitectura de red, un sistema IoT está conformado por tres bloques o elementos principales: a) nodo sensor, cuyas características y prestaciones dependen de cada caso de uso; b) red de comunicaciones, responsable de transmitir en tiempo y forma la información recabada; y c) infraestructura de nube, empleada para el almacenamiento/procesamiento de la información. En este sentido, el presente trabajo de investigación se ubica en el elemento de la infraestructura de nube.

1.1.1 Gestión de datos de IoT en la nube

Uno de los principales retos científico-técnicos en la gestión de datos provenientes de escenarios de IoT consiste en brindar soporte eficiente (en términos de prestaciones de calidad de servicio, QoS) en cada una de las etapas de los flujos de trabajo definidos para el procesamiento de dichos datos. Desde el acceso a los datos recolectados, pasando por el preprocesamiento y procesamiento [2], así como su exhibición, preservación y/o entrega para procesos de toma de decisiones.

En este contexto, a la fecha se han propuesto enfoques para gestionar los datos desde la etapa de adquisición en los nodos sensores hasta su almacenamiento en la infraestructura de nube. Una de las soluciones más prominentes se basa en el concepto de *flujos de trabajo*, los cuales se definen como una especificación formal de un proceso científico, que representa, optimiza y automatiza los pasos analíticos y computacionales que los científicos deben llevar desde la selección e integración de datos, el cálculo y el análisis hasta la presentación y visualización de los datos finales [67].

Independientemente del caso de uso, es posible identificar etapas genéricas que deben

[2] Ambas acciones se pueden llevar a cabo tanto en el borde como en la nube

satisfacer los flujos de trabajo en IoT, como es el caso de: adquisición, transmisión, almacenamiento/procesamiento, análisis y visualización, las cuales establecen un flujo de trabajo general para el manejo de los datos IoT [61].

A fin de ilustrar lo anterior, en la Fig. 1.1 se presenta el flujo de trabajo de manejo de datos IoT, en el cual, a partir de la adquisición de la información en la etapa de sensado, los datos se trasmiten a través de una red de acceso fija o inalámbrica hacia Internet. En particular, el almacenamiento, procesamiento y visualización de los datos se realiza en la infraestructura de nube, la cual puede ser del tipo privada o pública. En cualquier caso, dicha infraestructura debe disponer de *grid*, clústeres, o servidor(es) donde se despliegan flujos de trabajo con el fin de obtener en una etapa final un conjunto de datos derivados que puedan ser entregados y visualizados por aquellas organizaciones o entidades que tienen un interés de obtener conocimiento de los datos capturados.

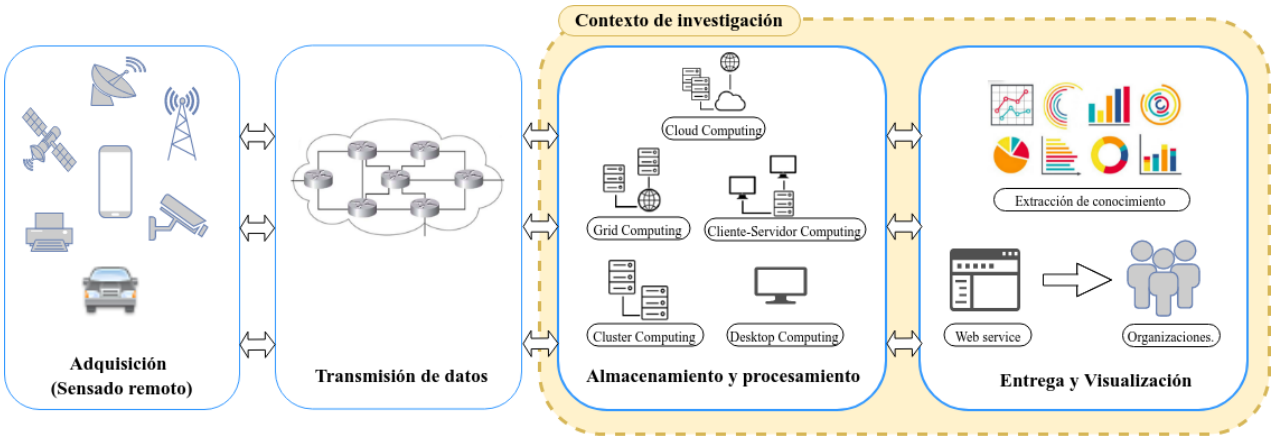


Figura 1.1: Flujo de trabajo genérico para el manejo de datos en IoT.

El manejo de datos típico que se realiza en flujos de trabajo se describe en la Fig. 1.2. La taxonomía que se ilustra en dicha figura, se establece en este trabajo de Tesis para el manejo de datos IoT, en función del trabajo presentado en [64]. En esta taxonomía, tenemos tres tipos de tratamientos de datos:

- Tratamiento para la toma de decisiones. Básicamente consiste en los procesos aplicados sobre los datos ya sea para la extracción de conocimiento o patrones en los datos (proceso

decremental), como generar valor agregado a ellos (proceso incremental).

- Tratamiento para la administración de meta-datos. Hace referencia a la gestión de los meta-datos de los datos.
- Tratamiento para establecer control de los datos. Corresponde a los procesos de seguridad que se le proporcionan a los datos, como por ejemplo confidencialidad.

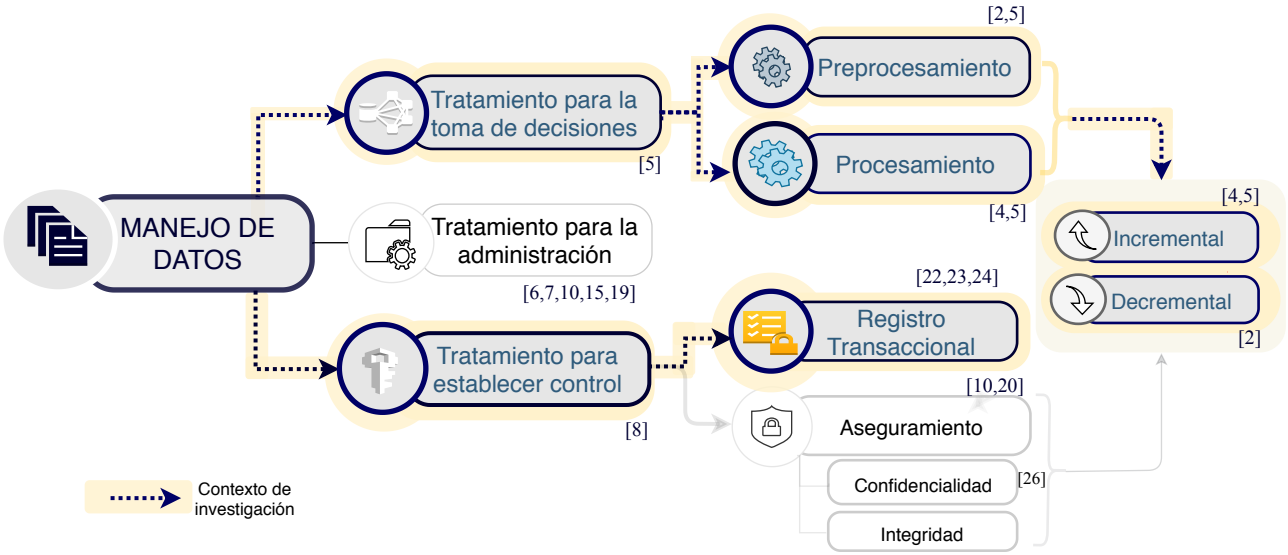


Figura 1.2: Taxonomía de técnicas de manejo de datos en escenarios IoT.

Cabe destacar que la etapa de tratamiento para la toma de decisiones puede involucrar tareas de preprocesamiento y procesamiento, a realizarse de forma incremental o decremental. En particular, el procesamiento incremental consiste en tomar un dato F y agregarle una característica adicional (plus) para conseguir un dato F', que es una versión incremental de F con un valor agregado. Ejemplo de esto podría ser procesar una imagen (dato F) para agregar una marca de agua, o agregarle seguridad mediante cifrado o *hash* para la firma, un enmascarado para el nombre del archivo, etc. (dato F'). El segundo esquema de procesamiento se le denomina decremental, donde se tiene una fuente de datos amplia, que pasa a través de un conjunto de procesos o algoritmos de analítica y se consiguen datos útiles, este subconjunto de los datos originales, básicamente es el resultado de una etapa que se conoce como preprocesamiento. Posteriormente, se concatena este esquema de

procesamiento para conseguir una reducción del dato original con la finalidad de quitar redundancia, realizar limpieza, etc., estos datos pasan a una etapa de procesamiento conocida como analítica, que puede ser agrupamiento o clasificación para encontrar patrones. Finalmente, del patrón se tienen datos útiles para apoyar un proceso de toma de decisiones.

Actualmente, este tipo de procesamiento mencionado en la fase de tratamiento, se lleva a cabo mediante sistemas de flujos de trabajo en ambientes científicos, como por ejemplo aplicar compresión, corrección y codificación en imágenes satelitales [28].

Con respecto al tratamiento administrativo de los datos se realiza el manejo de los meta-datos de los datos, tales como dónde se encuentran los datos, de dónde provienen, de quién son, tamaño, etc.

En el tratamiento para establecer control de los datos se aplican dos tareas, la primera es básicamente el registro de trazabilidad de las operaciones sobre los datos necesario para evitar problemas tales como el no repudio y la segunda tarea es el aseguramiento (por ejemplo, confidencialidad e integridad) que se requiere en los datos sensibles en el momento que son trasmitidos durante las diferentes etapas del flujo de trabajo y primordialmente en entornos de la nube.

En este contexto, los datos recolectados, el tratamiento que se realiza a dichos datos, así como la información obtenida a partir de la transformación de los mismos debería ser veraz y confiable. Lo anterior resulta clave para que los procesos de toma de decisiones sean exitosos. Cualquier alteración en cualesquiera de estos eventos resultaría en información errónea que obstaculizaría la labor de los tomadores de decisiones. Es por eso que en este trabajo de investigación se abordará el manejo de datos (ver Fig. 1.2) que tiene que ver con el tratamiento para la toma de decisiones en el cual se aplican diferentes procesos de preprocesamiento y procesamientos a lo largo de las etapas de un flujo de trabajo, y que a su vez dichos procesos generan un registro transaccional en una red de verificabilidad.

1.1.2 Integración de IoT y bloques encadenados

Para conseguir un procesamiento veraz y confiable, en el estado del arte se ha comenzado a estudiar soluciones que realicen la integración de IoT y bloques encadenados (Blockchain por su acepción en Ingles) [56, 59].

Estas soluciones permiten llevar un registro de las transacciones o procesos realizados en cada etapa de los flujos de trabajo donde se procesan datos provenientes de escenarios del IoT.

Es importante tomar en cuenta que las etapas de procesamiento de un flujo de trabajo pueden ser desplegadas en diferentes tipos de infraestructuras de cómputo, las cuales pueden incluso estar distribuidas geográficamente y bajo la supervisión por diferentes entes de responsabilidad (usuarios, organizaciones, entidades gubernamentales, etc). En este tipo de escenarios, el registro y verificación de transacciones se convierte en una necesidad.

Si bien se espera que la tecnología de bloques encadenados pueda garantizar verificabilidad en un flujo de trabajo de procesamiento de datos IoT, esta integración no es trivial dado que los principios de diseño de los bloques encadenados están orientados a características diferentes a las que se plantean en el Internet de las cosas.

En [59] los autores describen los principales desafíos que implica dicha integración son básicamente tres: el primero es la seguridad de los datos, el segundo tiene que ver con el almacenamiento de información extraída de los dispositivos IoT, mientras que el tercero tiene que ver con la eficiencia en el procesamiento de transacciones y el transporte de los datos entre las etapas de los flujos de procesamiento de dichos datos.

Específicamente en el rubro de seguridad, los autores de [59] observan la confidencialidad de los datos generados por IoT, los dispositivos IoT pueden emitir datos corruptos dado diferentes factores (p.e desconexión, entorno, interceptación, etc) y teniendo en cuenta que los bloques encadenados garantiza la inmutabilidad de los datos en la cadena, cuando los datos registrados ya están dañados en los bloques encadenados, permanecerán corruptos. También se considera un desafío el anonimato

y privacidad de los datos en la integración de bloques encadenados con IoT especialmente para aquellos escenarios donde la información generada es sensible. Asegurar el dispositivo para que los datos se almacenen de forma segura y no sea accedido por personas sin permiso es un desafío, ya que requiere la integración de software criptográfico de seguridad en el dispositivo, entre otros.

En el rubro de almacenamiento, el estudio indicó que la capacidad de almacenamiento resulta en un problema porque los dispositivos IoT pueden llegar a generar rápidamente gigabytes (GB) de datos en tiempo real, lo cual resulta en un incremento en la cantidad de datos a ser registrados en la red de verificabilidad.

Finalmente, el estudio en cuestión indica que la escalabilidad de la red de bloques encadenados resulta un problema en escenarios reales porque diferentes implementaciones de bloques encadenados solo pueden procesar unas pocas transacciones por segundo; que se requiere de un mecanismo de consenso para mantener la consistencia e integridad de las transacciones realizadas, lo cual podría generar un cuello de botella para una aplicación IoT. Este es el tipo de desafío al que se hace frente en el presente trabajo de investigación.

1.2 Planteamiento del problema

Como se mencionó previamente, la integración de sistemas IoT con los bloques encadenados recién se está abordando en el estado del arte [59]. También se ha mencionado que los esquemas de procesamiento de flujos de trabajo tradicionales [6, 15, 27, 66] están siendo recientemente utilizados para desplegar sistemas IoT [61]. La tecnología de bloques encadenados [56] puede ofrecer verificabilidad a los flujos de trabajo desplegados para el procesamiento de datos IoT, pero esta integración no es trivial debido principalmente a tres problemáticas: la primera se presenta porque no se puede delegar la tarea de verificación a los responsables de ejecutar los aplicativos de un flujo trabajo. La segunda tiene que ver, con la eficiencia de procesamiento de datos del IoT, afectada principalmente por la sobrecarga producida por la red de verificabilidad. Finalmente, el manejo de la

información requerido para realizar el despliegue de dicha red, el cual involucra desafíos tales como el manejo de activos criptográficos (certificados, llaves privadas).

A continuación se describen, en forma específica, los problemas que se presentan al delegar la verificabilidad a los responsables de ejecutar las etapas de un flujo de trabajo:

- Control de secuencia de procesos: esta problemática se presenta porque un flujo de trabajo debe garantizar entrega continua de datos a través de sus etapas y, aunque los motores de construcción de estos esquemas de procesamiento son los encargados de desplegar las aplicaciones de cada etapa y en algunos casos de hacer la entrega de los datos a las mismas, éste no puede garantizar que las etapas realicen tres tareas esenciales típicamente definidas en el **modelo ETC** (extracción, transformación y carga, o ETL por sus siglas en ingles) que típicamente se ha utilizado en la literatura en sistemas *Data Warehouse* o *Business Intelligence* para la extracción de una base de datos, procesarla y cargar los resultados en una nueva base de datos (ver Fig 1.3).

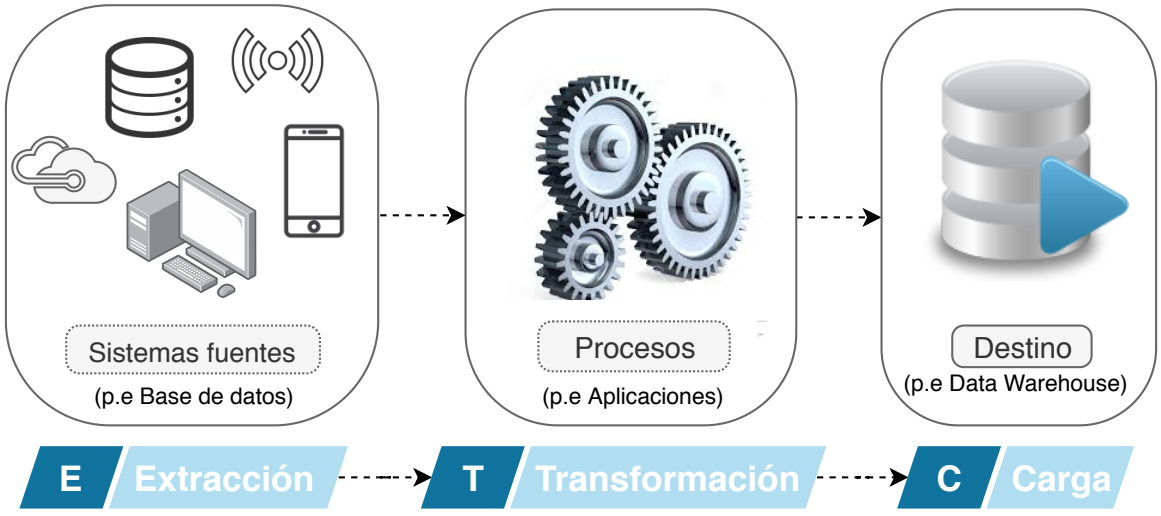


Figura 1.3: Modelo de extracción, transformación y carga (ETC o ETL por sus siglas en inglés).

En este trabajo de investigación, las tareas del modelo ETC corresponden a:

- **Extracción**: los datos deberían provenir de una fuente de datos que no debería cambiar

para evitar que los datos entrantes de cada etapa introduzcan errores que se pueden acarrear hasta la etapa de visualización obstaculizando el proceso de toma de decisiones.

- **Transformación**: Los aplicativos que se utilizan en cada etapa de un flujo de trabajo no deberían alterarse para brindar certidumbre sobre la transformación realizada a los datos en cada etapa del flujo de trabajo.
- **Carga**: Los datos procesados por cada etapa deberían ser los requeridos por las siguientes etapas en un flujo para evitar errores que afecten a la información final resultante y éstos deberían ser verificables en cada etapa.

- Identificación de omisiones o alteraciones: esta problemática se presenta cuando cualquiera de las tres fases del modelo ETC es alterada por los participantes en cualesquiera de los siguientes escenarios:
  - Omisión y/o alteración en la ejecución de los aplicativos asociados a una(s) etapa(s) del flujo de trabajo.
  - Omisión y/o alteración en la entrega continua de datos.

Con respecto al problema de eficiencia, específicamente se tiene:

- Volumen de datos: Cuando se procesan grandes volúmenes de datos, se incrementan los registros de transacciones, lo cual afecta directamente a la experiencia del usuario final del flujo de trabajo.

Los registros de transacciones en un flujo de trabajo que se consideran en esta investigación, básicamente son: 1) registro del servicio (o participante) que realizara el procesamiento en una determinada etapa del flujo de trabajo, 2) registro de asignación que especifica cual carga de trabajo (dato a procesar) ha sido asignada a que servicio o participante, y 3) el registro del procesamiento efectuado por un determinado servicio a la carga que le ha sido asignada.

A continuación se describen, en forma específica, los problemas de manejo de los datos:

- La heterogeneidad de los aplicativos: la heterogeneidad de los aplicativos ejecutados en las etapas de los flujos de trabajo producen problemas de portabilidad que terminan obstaculizando el despliegue de las soluciones en escenarios del mundo real.

- Manejo de material criptográfico: Resulta crítica la gestión de certificados y llaves privadas así como el intercambio de las llaves privadas entre las etapas que componen al flujo de trabajo.

1.3 Hipótesis

Con el fin de hacer frente a la problemática de realizar una verificación continua de las transacciones realizadas en flujos de trabajo ejecutados para el procesamiento de datos/meta-datos provenientes de escenarios del IoT[59, 61], se definen las siguientes premisas:

1) La información que describe las acciones o procesos realizados por cada etapa de un flujo de trabajo a los datos que son procesados, se puede manejar mediante un modelo de Extracción, Transformación y Carga o ETC (ETL por sus siglas en ingles: Extract, Transform and Load).

2) Un marco de gestión realizando las funciones de intermediario entre el flujo de trabajo y la red de verificabilidad inmutable (basada en bloques encadenados) podría capturar la información ETC de cada etapa para definir las transacciones realizadas por dichas etapas y automáticamente registrarlas en la red de verificabilidad inmutable. Esto garantizaría que los registros de transacciones sean en efecto realizados y que los usuarios o aplicaciones involucradas en el flujo de trabajo no intervienen en la realización de dichos registros.

3) La aplicación de patrones de paralelismo orientados al procesamiento concurrente de tareas en las etapas de los flujos de trabajo puede mejorar el rendimiento de los mismos, lo cual compensaría, dependiendo de los recursos disponibles, los costos de las tareas de verificabilidad de transacciones[3].

[3]Estudios muestran que "Las transacciones tradicionales requieren una institución de confianza centralizada. La confirmación y el registro de las transacciones dependen totalmente de la institución de confianza"[46]. Esto, en práctica, incrementa el costo de las transacciones afectando directamente la eficiencia y factibilidad de la verificación de transacciones en escenarios reales

4) La utilización de contenedores virtuales resolverá la problemática asociada a la heterogeneidad y portabilidad de los aplicaciones de las etapas del flujo de trabajo [17, 29].

Dado el problema mencionado anteriormente y las premisas previamente establecidas se plantea la siguiente hipótesis:

Un gestor intermediario entre un flujo de trabajo para el procesamiento de datos IoT basado en un modelo de extracción, transformación y carga (ETC) permite realizar verificabilidad continua a las transacciones realizadas en flujos de trabajo, mientras que el uso de patrones de paralelismo encapsulados en contenedores virtuales permite reducir el impacto de dicha verificabilidad en el rendimiento de dichos flujos de trabajo.

1.4 Objetivos

El objetivo general del trabajo de investigación es el siguiente:

Obtener un marco de gestión de verificabilidad continua para flujos de trabajo aplicables a escenarios de IoT.

Los objetivos particulares son los siguientes:

- Definir un esquema de entrega continua de datos para el manejo eficiente de grandes volúmenes de datos IoT sensibles mediante paralelismo basado en tareas.
- Definir mecanismos de verificabilidad continua de transacciones realizadas con datos IoT sensibles en flujos de trabajo mediante bloques encadenados.

1.5 Metodología de investigación

A fin de alcanzar los objetivos de la investigación, a continuación se describen las etapas de la metodología del presente trabajo de investigación.

Etapa 1. Revisión del estado del arte. En esta etapa se analizó el estado del arte relacionado con flujos de trabajo y conceptos fundamentales de bloques encadenados.

Etapa 2. Diseño y desarrollo del marco de gestión. La meta principal de esta etapa consistió en obtener un marco de gestión con verificabilidad continua en flujos de trabajo de entrega continua. Para ello fue necesaria la realización de las siguientes actividades:

- Definición conceptual de la arquitectura del marco de gestión.
- Diseño y desarrollo de los componentes del marco de gestión: entrega continua (CD) de los datos y verificabilidad continua (CV).
- Diseño de mecanismos de interconexión, gestión, orquestación y coreografía (proceso de ejecución y sincronización) entre los componentes del marco de gestión propuesto.
- Diseño y desarrollo de un esquema de verificación continua para flujos de manejo de datos en IoT. Este esquema involucra una red de verificabilidad de bloques encadenados y mecanismos de aseguramiento de verificabilidad de transacciones o procesos realizados sobre los datos.

Etapa 3. Evaluación experimental del marco de gestión. La tercera etapa de la metodología de la investigación consistió en definir como se iba a evaluar nuestra propuesta. Para ello se realizarán las siguientes tareas:

- Implementación de prototipo del marco de gestión que incluya un componente de verificabilidad continua para la entrega continua de los flujos de trabajo en dos estudios de caso (movilidad de usuarios y medioambiental).
- Definición de solución afín a nuestra propuesta para realizar un análisis comparativo.
- Diseño de experimentos (posibles configuraciones de parámetros) y definir métricas para la evaluación experimental.
- Evaluar y analizar resultados preliminares con el fin de ajustar el diseño (en caso de ser requerido).

Etapa 4. Análisis de resultados y redacción de tesis. En esta etapa el objetivo es analizar los resultados obtenidos de nuestra solución frente a una solución del estado del arte y finalizar la

redacción del documento de tesis. Específicamente, se realizaron las siguientes tareas:

- Analizar resultados finales y verificar la hipótesis.
- Presentar dificultades en la realización del trabajo de investigación y establecer trabajo futuro.

1.6 Organización de la tesis

La organización de la tesis es la siguiente. En el Capítulo 2 se presenta el marco teórico sobre el cual se desarrolla el trabajo de investigación. Posteriormente en el Capítulo 3 se describe el estado del arte con trabajos relacionados en la creación de flujos de trabajo y el uso de los bloques encadenados en el campo del IoT. En el Capítulo 4 se describe el diseño y cada uno de los componentes del marco de gestión propuesto. En el Capítulo 5 se describen los estudios de caso donde se validó la solución propuesta, junto con los experimentos y resultados. Finalmente, en el Capítulo 6 se presentan las conclusiones y trabajo futuro.

2

Marco teórico

En este capítulo se presentan los conceptos fundamentales relacionados con la presente investigación, con énfasis en flujos de trabajo y bloques encadenados.

2.1 Flujos de trabajo

Los flujos de trabajo se definen como el conjunto de etapas definidas por el usuario a través de las cuales los datos serán procesados o almacenados con el fin de obtener información útil de ellos y posteriormente, ser enviados y presentados a las entidades interesadas, con el fin de contribuir al proceso de toma de decisiones de su proceso de negocio [37]. En IoT estas etapas son estructuras configurables que ejecutan operaciones de procesamiento sobre los datos sensibles y se encuentran encadenadas entre sí.

En este sentido, el proceso de ejecución de las etapas de procesamiento se le denomina *entrega continua* de datos. La implementación y ejecución de cada una de las etapas de un flujo de trabajo (procesamientos) puede ser responsabilidad de diferentes organizaciones. Es decir, cada organización

procesa el volumen de datos recibido y lo entrega a una siguiente organización para el procesamiento en una etapa posterior. Por tanto, los componentes de un flujo de trabajo particular comúnmente se despliegan en un entorno totalmente distribuido (por ejemplo, la infraestructura de nube).

A modo de ejemplo, una etapa en el flujo de trabajo puede corresponder a la compresión de datos con el fin de reducir los costos de almacenamiento, otra etapa subsecuente puede ser la encargada del cifrado de los datos sensibles con el fin de brindar confidencialidad, otra etapa puede estar definida para garantizar la confiabilidad de los datos, en la cual los datos son dispersos en múltiples ubicaciones de almacenamiento, entre otras.

De manera general, el conjunto de estas etapas o actividades encadenadas, establecen un flujo de trabajo que básicamente es la especificación formal de un proceso científico, el cual representa, optimiza y automatiza los pasos analíticos y computacionales que se deben de seguir desde la adquisición e integración de datos, el cálculo y el análisis, hasta la presentación y visualización de los datos finales [67].

2.1.1 Componentes de los flujos de trabajo

En [75] hacen una descripción de los componentes de un flujo de trabajo, la cual se describe a continuación y se adapta en el presente trabajo de investigación:

Unidades de procesamiento. Una unidad de procesamiento (PU, por sus siglas en inglés) es una entidad abstracta que ejecuta una tarea o proceso atómico el cual se define al crear la PU. Una PU contiene tres interfaces de entrada/salida (I/O) a través de las cuales se recibe información desde una fuente de datos (data source) u otras PUs; y se envía información hacia un almacén de datos u otras PUs. Estas interfaces son: 1) El Sistema de archivos (FS), 2) Red(N) y 3) Memoria (M). Las interfaces de Sistema de Archivos y de Memoria permiten transportar flujos de información entre PUs locales, mientras que la interfaz de Red permite que el flujo sea transportado desde/hacia PUs remotas.

Etapas. Una etapa en el flujo de trabajo es un componente de integración abstracto que alberga

una o varias PUs interconectadas por las interfaces I/O para formar una cadena de procesamiento, ejecutar en conjunto una tarea específica y concentrar los resultados obtenidos por cada PU. La interconexión a su vez de diferentes etapas, permite un flujo de datos desde la adquisición de los mismos hacia una o múltiples ubicaciones de procesamiento/almacenamiento en la nube donde la datos sean manejados y posteriormente entregados a una etapa final de visualización al usuario final u organización.

Patrones de organización de unidades de procesamiento. La forma como se conectan u organizan cada una de las PUs o etapas del flujo de trabajo y las configuraciones de interfaces de I/O para cada PU, determinan la flexibilidad, desempeño y eficiencia de las operaciones realizadas a lo largo del flujo de trabajo. Es por ello, que un flujo de trabajo según los requerimientos del usuario puede ser desplegado con diferentes patrones para la organización de los PUs. En [69] son descritos en detalle 20 patrones de flujos de trabajo diferentes, alguno de ellos son el patrón secuencial (Fig. 2.1 a.), el patrón de división paralela (Fig. 2.1 b.) y el patrón de sincronización (Fig. 2.1 c.).

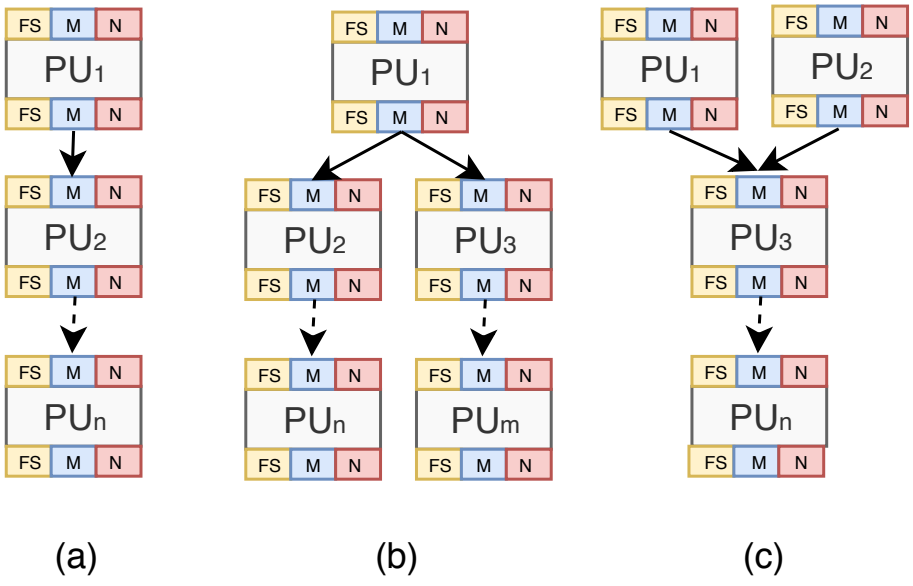


Figura 2.1: Patrones de flujos de trabajo.

Por ejemplo, las PUs pueden ser organizadas en forma paralela para mejorar el desempeño o en forma secuencial para incrementar la funcionalidad. Un ejemplo completo de un flujo de trabajo configurado por el usuario se puede observar en la Fig. 2.2.

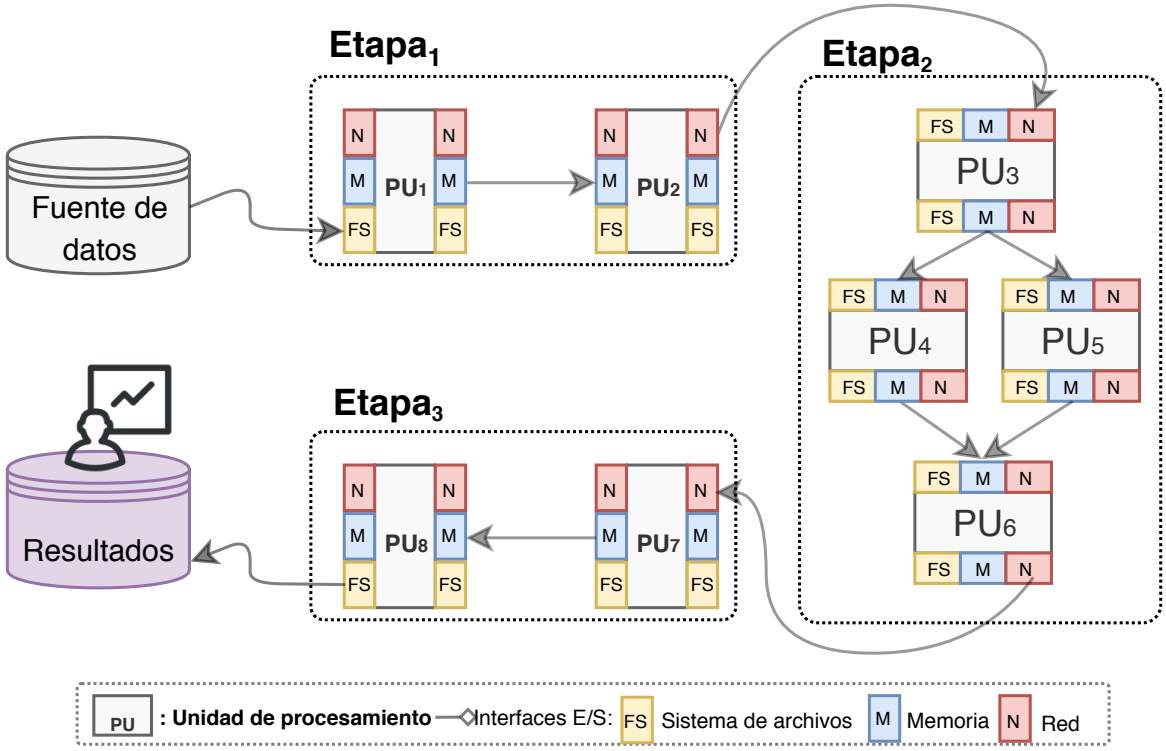


Figura 2.2: Ejemplo de flujo de trabajo involucrando patrones.

En este ejemplo el usuario ha configurado un flujo de trabajo con tres etapas, en la primera de ellas se tienen dos unidades de procesamiento (PU_1 y PU_2) interconectadas a través de la interfaz de Memoria (M) aplicando un patrón secuencial. La primera unidad de procesamiento (PU_1) es la encargada de recibir o adquirir los datos sensibles IoT provenientes de una fuente de datos (data source) a través de la interfaz de Sistema de Archivos (FS). La segunda unidad de procesamiento (PU_2) recibe los resultados de PU_1 por la interfaz de Memoria, aplica el procesamiento a los datos según lo haya definido el usuario y entrega los resultados a través de la interfaz de red (N) a la tercera unidad de procesamiento PU_3, que hace parte de una segunda etapa del flujo de trabajo. Esta segunda etapa consta básicamente de cuatro PUs (PU_3,PU_4,PU_5 y PU_6) que están interconectadas a través

de la interfaz de memoria y están organizadas mediante un patrón de división paralela aplicado en PU_3 y el patrón de sincronización en PU_6. En este ejemplo los PUs que interconectan las etapas del flujo de trabajo se comunican a través de la interfaz de red, esto significa que las etapas pueden estar dispersas geográficamente o ubicadas en diferentes nodos de cómputo en los cuales ha sido desplegado el flujo de trabajo.

Dado que las PUs que conforman una etapa del flujo de trabajo pueden requerir una gran cantidad de bibliotecas para su correcto funcionamiento y un entorno previamente configurado, se hace necesario hacer uso de la Virtualización para lidiar con este problema.

2.2 Contenedores virtuales: una estrategia de virtualización para garantizar portabilidad

La virtualización es una tecnología que permite una asignación flexible de recursos físicos a aplicaciones virtualizadas de una forma dinámica y cuantificable, que permite ajustar las cargas de trabajo de la aplicación.

Adicionalmente, la virtualización permite que múltiples instancias de aplicaciones virtualizadas ("tenants") compartan un servidor físico, a esto se le conoce como *multi-tenancy*. Esto hace posible que las aplicaciones puedan ser consolidadas en un conjunto más pequeño de servidores y con ello reducir los costos de operación. Adicionalmente, al reducir los espacios se simplifica la replicación y escalado de las aplicaciones. Estos beneficios y otros sin mencionar, han producido que cada vez más que los centros de datos estén más virtualizados. Los centros de datos son aquellos servidores que brindan recursos de cómputo, almacenamiento y red para aquellas aplicaciones TI empresariales de las cuales las empresas modernas dependen cada vez más. En [63] describen dos tecnologías de virtualización de servidores: virtualización de hardware y la virtualización a nivel de sistema operativo. Estas tecnologías son comúnmente utilizadas en los centros de datos y entornos de nube con el fin de desacoplar las aplicaciones del hardware en el cual se ejecutan.

2.2.1 Virtualización de recursos a nivel de hardware

En este tipo de virtualización lo que se hace es ejecutar un *hipervisor* que se encarga de virtualizar los recursos del servidor en múltiples maquinas virtuales (VM). Cada VM ejecuta su propio sistema operativo y aplicaciones.

2.2.2 Virtualización de recursos a nivel de sistema operativo

En este caso se virtualizan los recursos a nivel del sistema operativo y se encapsulan los procesos estándar del sistema operativo y sus dependencias para crear contenedores, que son administrados colectivamente por el núcleo del sistema operativo subyacente.

En la Figura 2.3 se ilustran algunos ejemplos de cada una de las tecnologías de virtualización de servidor.

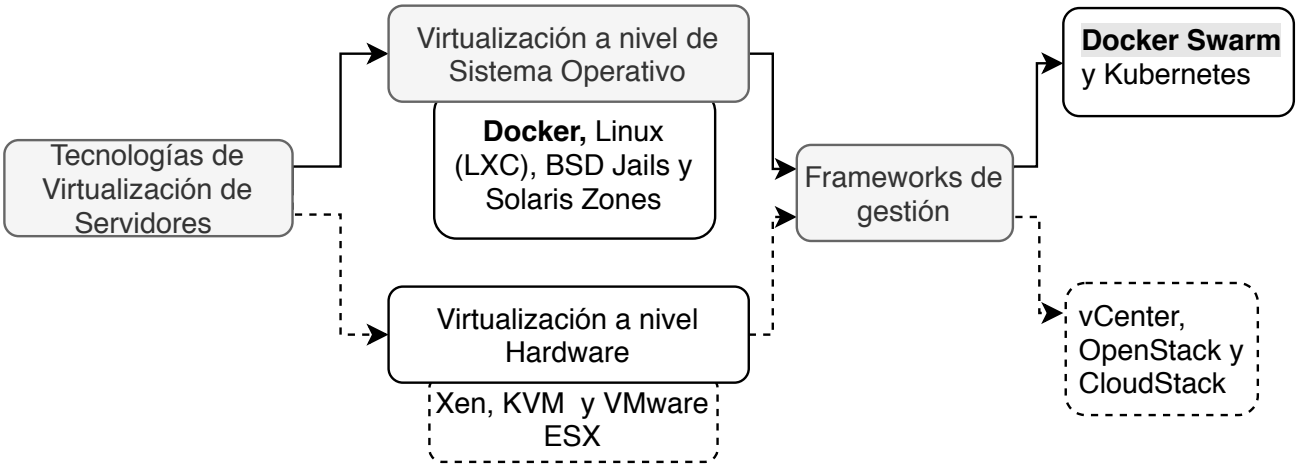


Figura 2.3: Ejemplos de tecnologías de virtualización.

Se puede observar en la figura anterior que ambas tecnologías de virtualización tienen marcos de gestión (Management Frameworks) que permiten que las máquinas virtuales y las aplicaciones puedan ser implementadas y administradas a gran escala, como por ejemplo, similar a un centro de datos (o *datacenter*).

A pesar de que la virtualización de hardware ha sido la tecnología de virtualización predominante para implementar, empaquetar y administrar aplicaciones, los contenedores están cumpliendo cada vez más esa función debido a la popularidad de sistemas como Docker [7], incluso asumen virtualización de bajo costo y rendimiento mejorado en comparación con máquinas virtuales [63].

En la Fig. 2.4 se observa la diferencia existente entre las máquinas virtuales (VMs) y los contenedores virtuales (VCs), la cual radica en los elementos adicionales que requiere una máquina virtual (Hipervisor, Hardware virtual y Sistema operativo huésped) para ejecutar una determinada aplicación, en contraste con los contenedores virtuales que comparten el núcleo del sistema operativo subyacente.

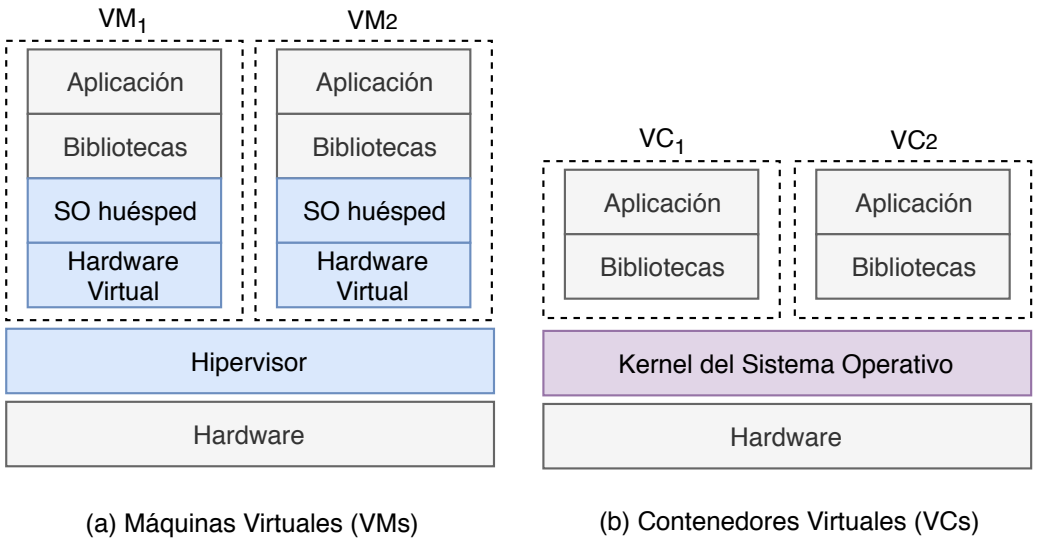


Figura 2.4: Máquina virtual y Contenedor Virtual.

Diferentes investigaciones [63, 77] han realizado estudios de comparación de rendimiento entre las maquinas virtuales y los contenedores. Como resultado de estas investigaciones se ha observado que los contenedores tienen un rendimiento simple, a pesar de que en escenarios multitenant pueden sufrir interferencias en el rendimiento. Sin embargo, una gran ventaja de los contenedores sobre las máquinas virtuales radica en que los contenedores permiten límites de recursos flexibles, que son útiles en escenarios donde hay mucha carga de trabajo y puede haber un sobre-compromiso de los

recursos, ya que pueden usar recursos sub-utilizados asignados a otros contenedores. Adicionalmente, la popularidad de los contenedores se ha visto impulsada por su integración en el proceso de desarrollo de software. Particularmente, el uso de Docker en los sistemas de archivos en capas de copia en escritura y el control de versiones ha permitido una entrega e integración continua más fácil y un proceso de desarrollo de software más ágil.

2.2.2.1. Plataforma Docker

Docker[1] es un proyecto de código abierto que permite la creación de unidades estandarizadas de software conocidas como contenedores. Estos contenedores se caracterizan por ser ligeros y portables para las aplicaciones software que pueden ser ejecutas y desplegadas en cualquier máquina física que tenga Docker instalado sin importar el sistema operativo subyacente.

La plataforma de Docker incorpora una herramienta conocida como *Docker Compose* que permite definir y ejecutar aplicaciones Docker de múltiples contenedores. Esto permite configurar, crear e iniciar los servicios de una aplicación que defina el usuario. Adicionalmente, es posible desplegar varios *clusters* de contenedores mediante un subcomponente de Docker conocido como *Docker Swarm*. Este componente permite administrar un clúster de contenedores Docker denominado enjambre (Swarm), a través del cual es posible implementar servicios de aplicaciones y administrar el comportamiento del enjambre. Básicamente es un orquestador de contenedores.

Los conceptos anteriormente descritos son fundamentales para el desarrollo de nuestro trabajo de investigación, particularmente, en el componente de entrega continua en los flujos de trabajo. Para el componente de verificabilidad continua se consideran nuevos conceptos.

[1] https://www.docker.com/

2.3 Bloques encadenados

El concepto de bloques encadenados (o *Blockchain* por su acepción en Ingles) se asemeja a un libro de contabilidad en el cual se registran transacciones, pero con la particularidad que este libro se encuentra distribuido en una red *peer-to-peer* o *P2P* (por sus siglas en inglés). Esta red básicamente es una arquitectura de aplicación distribuida que distribuye la carga de trabajo (tareas) entre *peers* o nodos que tienen los mismos privilegios. Adicionalmente, el contenido de la información registrada en la red *P2P* es establecida mediante un mecanismo de consenso con la finalidad de que la información registrada en los bloques encadenados sea inmutable. Esto permite que múltiples partes diferentes interactúen de manera segura con la misma fuente universal de verdad.

2.3.1 Tecnologías de bloques encadenados

Los bloques encadenados incorporan algunas tecnologías para lograr la inmutabilidad de datos, entre ellas, un mecanismo de confianza o de consenso.

2.3.1.1. Mecanismo de consenso.

Dado que el sistema de bloques encadenados es descentralizado, este no requiere de una autoridad confiable de terceros para garantizar confiabilidad y consistencia de los datos y las transacciones, para ello, los bloques encadenados adoptan el mecanismo de consenso descentralizado. La función de un protocolo o mecanismo de consenso es mantener el orden de las transacciones en una red de bloques encadenados, a pesar de las amenazas a ese orden.

Existen diferentes mecanismos de consenso tales como: 1) Práctica de tolerancia de faltas bizantinas - PBFT, 2) Prueba de Trabajo - PoW, 3) Prueba de estaca -PoS, 4) Prueba de estaca delegada - DPoS, 5) Prueba de ancho de banda - PoB, 6) Prueba de autoridad - PoA, 7) Prueba de tiempo transcurrido - PoET, entre otras.

Entre los mecanismos más conocidos, se encuentra la Prueba de Trabajo (PoW, por sus siglas

e ingles), la cual utiliza la solución de rompecabezas para demostrar la credibilidad de los datos. El rompecabezas suele ser un problema computacionalmente difícil pero fácilmente verificable. Sin embargo, dada la dificultad del problema, la Prueba de Trabajo necesita muchos cálculos, lo que resulta en un desperdicio de poder de cómputo. Este mecanismo es utilizado por dos de los sistemas de bloques encadenados más populares, Bitcoin y Ethereum [73]. Este último utiliza adicionalmente el mecanismo de consenso PoA.

En el mecanismo de Práctica de tolerancia de faltas bizantinas (PBFT, por sus siglas en ingles), una de las amenazas que se puede presentar y alterar el orden de las transacciones es la falla arbitraria simultánea, uno de los tipos de fallas bizantinas, de múltiples nodos de red. Al usar PBFT, una red de nodos de bloques encadenados puede tolerar hasta un número P de nodos defectuosos, donde P es una fracción arbitraria conocida del número total de nodos, con una máquina de estado replicada en diferentes nodos (una réplica se define como primaria). El algoritmo PBFT funciona de la siguiente manera: (a) Un cliente envía una solicitud de servicio a la máquina primaria, (b) El primario replica la solicitud de las copias de seguridad, (c) Las réplicas ejecutan la solicitud y envían respuestas, (d) El cliente espera $P+1$ respuestas idénticas de diferentes réplicas para considerar un resultado correcto. Este mecanismo de consenso se utiliza preferiblemente en sistemas donde se conoce el número total de nodos. Este mecanismo garantiza la consistencia y la integridad de los datos cuando ocurren fallas bizantinas en hasta $1/3$ de los nodos de la red. Por ejemplo, al usar PBFT, la red de nodos N de los bloques encadenados puede admitir un número P de nodos bizantinos, donde $P = (N-1)/3$. En otras palabras, PBFT garantiza que un mínimo de $2xP+1$ nodos alcancen un consenso sobre el orden de las transacciones antes de adjuntarlos al libro mayor compartido. La regla $2xP+1$ tiene la siguiente implicación: Necesitamos un mínimo de $2xP+1$ nodos para llegar a un consenso antes de pasar al siguiente bloque. El registro en cualquier nodo adicional (más allá de $2xP+1$) se retrasará temporalmente.

2.3.1.2. Desarrollo Tecnológico

La tecnología de bloques encadenados ha experimentado dos etapas de desarrollo: *Blockchain 1.0* y *Blockchain 2.0*. En la etapa *Blockchain 1.0*, la tecnología de bloques encadenados se usa principalmente para la criptomoneda. Además de Bitcoin, hay muchos otros tipos de criptomonedas, como Litecoin, Dogecoin, Ether, etc. Actualmente existen más de 700 tipos de criptomonedas [40]. Entres las características más importantes que destacan a una criptomoneda en contraste con la moneda tradicional, se tiene que sus operaciones de transferencia son irreversibles y rastreables, ya que se guardan de forma permanente en la cadena de bloques. Es descentralizada y anónima dado que no hay una organización de terceros involucrada y los comportamientos de usuarios son anónimos. La seguridad de la criptomoneda está garantizada por la criptografía de clave pública y el mecanismo de consenso de la cadena de bloques, que son difíciles de romper por el atacante.

En la etapa Blockchain 2.0, se introduce el contrato inteligente para que los desarrolladores puedan crear varias aplicaciones a través de contratos inteligentes. Un contrato inteligente puede considerarse como un dAPP ligero (aplicación descentralizada). Dado que los contratos inteligentes se pueden llamar entre sí a través de mensajes, los desarrolladores pueden desarrollar más dAPPs basados en los contratos inteligentes disponibles. En comparación con la aplicación tradicional, un dAPP puede ejecutarse de manera autónoma sin la necesidad de la asistencia y participación de terceros, dado que se desarrollan sobre la base de contratos inteligentes, y los contratos inteligentes se implementan y ejecutan en la cadena de bloques. Adicionalmente, estos contratos son estables, ya que los *bytecodes* de los contratos inteligentes se almacenan en el árbol de estado de los bloques encadenados. Cada nodo completo guarda la información de todos los bloques y estado de la base de datos, incluidos los códigos de byte de los contratos inteligentes. Adicionalmente, son trazables y seguros por la criptografía de clave pública y el mecanismo de consenso de los bloques encadenados. Dado que la información de invocación de los contratos inteligentes se almacena en los bloques encadenados como transacciones, todos los comportamientos de los dAPP se registran y se pueden

rastrear.

Es posible realizar una clasificación o categorización de los bloques encadenados de acuerdo a estas dos etapas de desarrollo mencionadas previamente: *Blockchain 1.0* y *Blockchain 2.0*. Sin embargo, en este trabajo de investigación, dado que el enfoque está sobre el procesamiento de los datos y verificabilidad de los mismos, la elección de los bloques encadenados a utilizar en este trabajo se centró en la forma cómo se acceden a los datos.

2.3.2 Categorías de bloques encadenados basadas en el acceso a datos

Los bloques encadenados pueden clasificarse según el acceso a los datos y la participación del mecanismo de consenso en cualquier cambio propuesto en su libro contable [33]. La Fig. 2.5 ilustra la clasificación de las tecnologías de bloques encadenados según el acceso de los datos junto con algunas características importantes de cada clasificación:

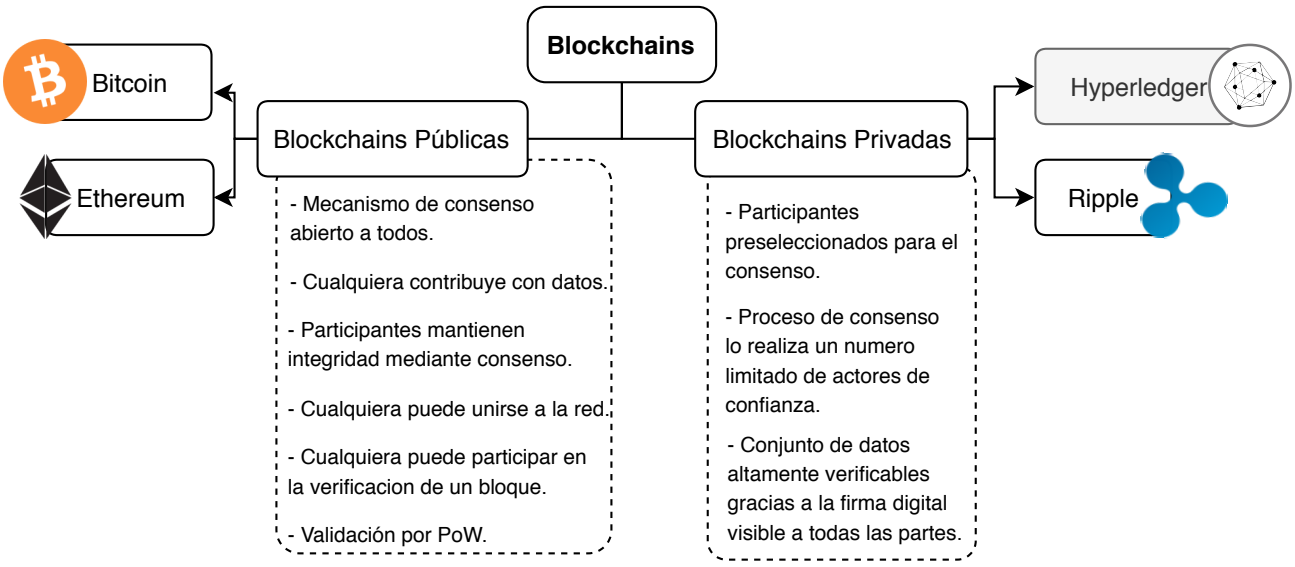


Figura 2.5: Categorías de bloques encadenados basadas en el acceso a los datos.

La primera categoría corresponde a los bloques encadenados públicos o sin permiso, en ellos cualquier usuario o entidad es libre de ingresar y participar en la construcción de los bloques encadenados. Las dos tecnologías más populares de bloques encadenados, Bitcoin [56] perteneciente a la era de *Blockchain 1.0* y Ethereum [73] de la era *Blockchain 2.0*, son ejemplos de bloques encadenados sin permiso.

Sin embargo, para diferentes modelos de negocio y aplicaciones no es adecuado un escenario público, tal como es el caso de este trabajo de investigación. En el cual se busca tener bloques encadenados que permitan establecer control en el procesamiento de los datos que ocurren en un flujo de trabajo, esto requiere tener predefinido previamente cuales son los actores o participantes en los bloques encadenados. Lo cual es una característica inherente de los bloques encadenados privados o con permiso tales como Hyperledger [19], éste es también (al igual que Ethereum) un sistema típico de *Blockchain 2.0*. Una característica importante de los bloques encadenados privados es que al conocer todos los actores que participan, es posible utilizar algoritmos de consenso como PBFT que pueden usarse para lograr un consenso sin la minería de PoW, lo que lleva a un tiempo de procesamiento de bloque mucho menor en comparación con el tiempo de bloques encadenados públicos, prácticamente considerado como realizado en tiempo real.

Una comparativa de los sistemas de bloques encadenados mencionados anteriormente se presentan en la siguiente tabla:

Bloques encadenados	**Bitcoin**	**Ethereum**	**Hyperledger**
Naturaleza	Pública	Pública	Privada
Consenso	PoW / SHA-256	Ethash PoW	PBFT
Propósito	Criptomoneda	Smart Contract	Chaincode
Lenguaje	Scripts basados en pila	Solidity/ Turing completo	Go, Java
Tiempo de procesamiento del bloque	$\sim 600s$	$\sim 15s$	Tiempo real

Tabla 2.1: Comparación entre sistemas de bloques encadenados.

2.3.3 Hyperledger

La Fundación Linux contiene muchos de los proyectos de código abierto más importantes del mundo, entre los cuales se destaca Linux. Estos proyectos son respaldados por más de 1000 compañías tales como Google e IBM para impulsar la innovación de forma rápida y escalable. En el 2016 se lanzó un nuevo proyecto organizado por la fundación Linux llamado Hyperledger [19], fue creado con la finalidad de avanzar en las tecnologías de bloques encadenados en la industria. En este proyecto global participan líderes en finanzas, banca, Internet de las cosas, cadenas de suministro, entre otros.

El proyecto de Hyperledger combina el concepto de bloques encadenados junto con un sistema para contratos inteligentes y otras tecnologías, para construir una nueva generación de aplicaciones transaccionales que tienen entre muchas característica la confianza y transparencia que permitan agilizar los procesos comerciales y las restricciones legales. Esto es en esencia Hyperledger, las comunidades de desarrolladores de software que crean *frameworks* y plataformas de bloques encadenados. Estos frameworks y plataformas hacen parte de una serie de tecnologías de bloques encadenados de negocios que incuba y promueve Hyperledger, entre las cuales se incluyen también los motores de contrato inteligentes, bibliotecas de cliente, interfaces gráficas, bibliotecas de utilidad y aplicaciones de ejemplo. Entre los más destacados, se encuentra el Framework de Hyperledger Fabric [3] y la herramienta de Hyperledger Composer.

2.3.3.1. Hyperledger Fabric

Es uno de los proyectos de Hyperledger, el cual consiste en una implementación de *framework* de bloques encadenados para desarrollar aplicaciones o soluciones con una arquitectura modular, aplicando el concepto de *plug-and-play* a sus componentes, tales como el consenso y los servicios de membresía. Adicionalmente, Hyperledger Fabric utiliza la tecnología de contenedores para almacenar contratos inteligentes llamados *chaincode* que comprenden la lógica de la aplicación del sistema. Los detalles sobre la arquitectura de este *framework* se describen en [9].

2.3.3.2. Hyperledger Composer

Es un conjunto de herramientas de colaboración utilizadas para la creación de redes empresariales de bloques encadenados de una manera simplificada y ágil. Estas herramientas le permiten a los propietarios y desarrolladores de negocios crear contratos inteligentes y aplicaciones de bloques encadenados para resolver problemas de negocios. Composer ofrece abstracciones centradas en la lógica de negocio, así como aplicaciones de muestra fáciles de probar para crear soluciones de bloques encadenados robustas.

3

Estado del Arte

Mediante una taxonomía (ver Fig 3.1) definida para centrar la búsqueda de los trabajos representativos y relacionados a la problemática (descrita en la sección 1.2) que se aborda en este trabajo de investigación, se realiza el presente capitulo del Estado del Arte.

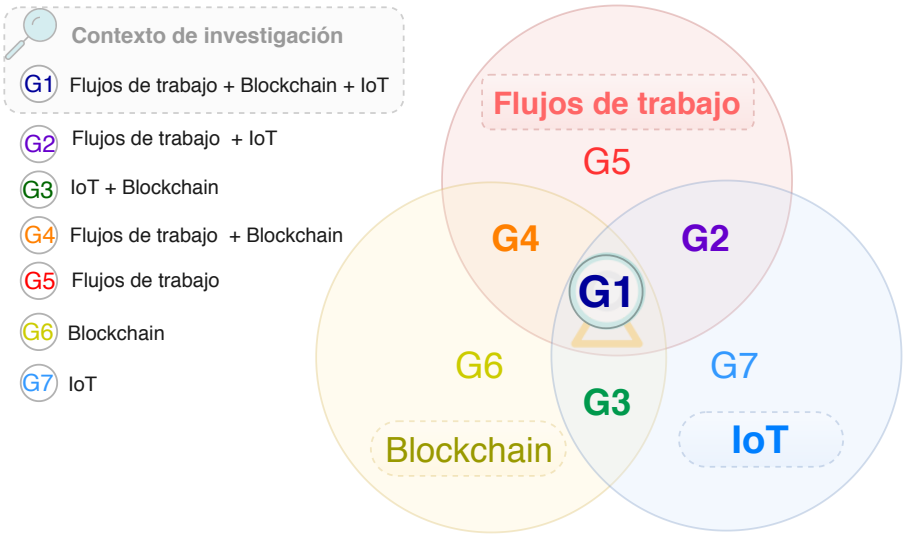


Figura 3.1: Grupos para selección y búsqueda de trabajos relacionados.

En dicha Figura se observa un conjunto de grupos como resultado de considerar las tres entidades (flujos de trabajo, escenarios IoT y bloques encadenados) en las cuales se ubica la problemática que se aborda en este trabajo de investigación. En este sentido, los trabajos descritos a continuación, consideran algunos de los componentes principales de este trabajo de investigación: construcción de flujos de trabajo, implementación de los flujos de trabajo en escenarios IoT, integración de sistemas IoT con los bloques encadenados y esquemas de verificabilidad de procesos realizados en flujos de trabajo. Cabe aclarar que dentro del proceso de búsqueda realizado al momento de la escritura y realización del presente trabajo de investigación, no se encontraron trabajos que abarcaran cada uno de estos componentes de manera conjunta.

3.1 Construcción de flujos de trabajo.

En este apartado se describen los trabajos encontrados en la literatura que permiten la construcción de flujos de trabajo científicos tradicionales, cada uno de ellos se describen a continuación:

DagOn [50] es una librería de código abierto para Python que permite la ejecución de flujos de trabajo mediante la definición explicita de tareas tales como: 1) tareas nativas de Python, 2) tareas web, 3) tareas *batch* ejecutadas mediante el protocolo SSH o mediante un programador local, 4) tareas cloud a partir de una imagen de maquina virtual o 5) tareas Docker representadas mediante un script de contenedor. Estas diferentes tareas permiten que Dagon pueda ejecutar flujos de trabajo masivos en una infraestructura heterogénea, dado que cada una de estas tareas son interoperables. Sin embargo, a pesar de dichas ventajas, el esquema que utiliza DagOn para declarar las tareas que requieren ser procesadas, implica una declaración uno a uno por cada dato a ser procesado. Para un escenario en el cual se tienen millones de datos a ser procesados, esto podría representar un problema. Por otro lado, no se tiene un registro que permita establecer un control en los procesos realizados en el flujo de trabajo por cada una de las tareas declaradas.

Parls [4] (o Scalable Parallel Scripting) proporciona un modelo para la administración y ejecución de flujos de trabajo compuestos por funciones de Python y aplicaciones externas en recursos computacionales arbitrarios. Entre las características principales de Parls esta la elasticidad automatizada, soporte para ejecución en múltiples sitios, tolerancia a fallas y administración de datos automatizada. Adicionalmente, brinda una manera transparente de escalar el procesamiento computacional desde cualquier lugar. Parsl está diseñado específicamente para abordar nuevas modalidades de flujo de trabajo, como la computación interactiva en las computadoras portátiles Jupyter [38]. Jupyter es una aplicación web de código abierto que tiene la capacidad de realizar limpieza y transformación de datos, simulación numérica, modelado estadístico, visualización de datos, aprendizaje automático, entre otras, a partir de la creación y compartición de documentos que contienen ecuaciones, visualizaciones, texto narrativo y código interactivo de un lenguaje de programación en particular (Jupyter soporta 40 lenguajes de programación, entre ellos python, java, scala).

Entre los flujos de trabajo tradicionales y comúnmente utilizados en el área científica, se destacan los trabajos de Pegasus [18], Triana [43] y Sacbe [27].

Pegasus es un marco de trabajo capaz de desplegar flujos de trabajo científicos complejos en recursos distribuidos y permite a los usuarios representar los flujos de trabajo en un nivel abstracto sin tener que preocuparse por los detalles de los sistemas de ejecución de destino. Por otro lado, Triana [43] es un servicio que permite a los usuarios construir flujos de trabajo complejos de manera transparente. Para lograr dicha transparencia, Triana incorpora una interfaz gráfica de usuario (o GUI por sus siglas en inglés) que facilita al usuario la agregación de servicios para construir aplicaciones complejas. Triana permite al usuario diferentes funcionalidades: 1) compartir y exponer servicios, 2) análisis de lo que podría pasar si el flujo de trabajo es alterado, 3) registrar datos de procedencia para el flujo de trabajo y 4) desplegar dicho flujo de trabajo en una red de Grid o P2P. Diferentes aplicaciones del servicio de Triana son presentadas en [13, 66]. Con respecto a Sacbe [27], esta es una arquitectura modular conformada por unidades de software encapsuladas, conocidas como Building

Blocks (BBs) que se comunican a través de la memoria, del sistema de archivos o de la red, para el almacenamiento, procesamiento, compartición y seguridad de los datos. Esta arquitectura permite a los usuarios construir flujos de trabajo para el procesamiento de los datos a través de los BBs. Su principal aportación es su diseño modular y flexible para generar un flujo de trabajo y de datos.

En general, existen muchos sistemas de flujo de trabajo, incluso con funcionalidades y propósitos similares. Es por ello que en [16] realizan una taxonomía de características de la forma en que los científicos utilizan los sistemas de flujo de trabajo existentes. Esta taxonomía permite evaluar la idoneidad de aplicar un determinado flujo de trabajo en un caso en particular.

3.1.1 Grupo 2: Flujos de trabajo en IoT.

En esta sección se presentan los trabajos encontrados en la literatura que consideran la incorporación de los flujos de trabajo en los entornos IoT. Los esquemas de procesamiento de flujos de trabajo tradicionales [6, 15, 27, 66] están siendo recientemente utilizados para desplegar sistemas IoT.

En [61] los autores presentan la orquestación de un entorno IoT mediante el motor de flujos de trabajo DagOn* [50]. En este estudio, proponen un esquema para la integración entre los componentes de adquisición de datos distribuidos basados en IoT y la computación distribuida orquestada mediante el motor de flujos de trabajo DagOn*. Básicamente, los autores incorporan una nueva tarea (al trabajo original DagoOn*) que permite la interacción con dispositivos conectados de IoT como tareas de un flujo de trabajo. El estudio es evaluado en un escenario real en el cual consideran datos medioambientales recabados por estaciones meteorológicas, radares y cámaras web que son proporcionados por un Servicio de observación y pronostico del clima marino en Italia. Las etapas en el flujo de trabajo fueron encapsuladas y desplegadas en contenedores virtuales mediante la plataforma de Docker. Su principal funcionalidad es su integración con otros sistemas en la nube (tales como Amazon,Google Cloud, entre otros). Sin embargo, este trabajo carece de un registro verificable e inmutable, de los procesos realizados en cada una de las etapas del flujo de trabajo.

En [28] presentan la arquitectura FedIDS pensada para construir plataformas geoespaciales confiables. FedIDS es un conjunto de componentes basados en la nube que incluye una arquitectura federada en la nube (Fed) y un servicio de entrega de imágenes por satélite (IDS, por sus siglas en inglés). El componente IDS en particular permite a los usuarios de la plataforma tener garantizada internamente la disponibilidad, integridad y confidencialidad de sus productos en escenarios colaborativos. En contraste con la idea de este trabajo de investigación, en FedIDS no se procesan datos provenientes del IoT sino grandes cantidades de imágenes/datos que son generados por diferentes satélites que observan la tierra, estos datos generados no solo son procesados y almacenados en plataformas geoespaciales confiables, sino también se realiza una distribución eficientemente entre socios/investigadores para crear productos derivados a través de flujos de trabajo colaborativos. La evaluación realizada de la arquitectura reveló su viabilidad, fiabilidad y eficiencia, en comparación con soluciones como SkyCDS [26] y Phoenix [25], en términos de rendimiento, consumo de almacenamiento e integridad/confidencialidad cuando se comparten imágenes/datos en escenarios colaborativos. Sin embargo, la arquitectura que se despliega en un flujo de trabajo colaborativo (es decir, diferentes organizaciones interactúan y cooperan en la generación de los productos), no cuenta con un control confiable y descentralizado que registre las operaciones o transacciones que se realizan dentro de la arquitectura para fines de trazabilidad de los productos que se generan por las diferentes organizaciones. Adicionalmente, no cuenta con un diseño genérico que permita desplegar la arquitectura en otras áreas.

FairWind [51] es un sistema de navegación inteligente para obtener mapas 3-D de alta resolución del fondo del mar. Estos mapas son necesarios para diferentes procesos: 1) modelar el impacto de las olas de tormenta en las actividades humanas costeras, 2) la difusión y la dispersión de contaminantes, 3) la deriva de objetos flotantes (seguridad en el mar – batimetría). Estos procesos son llevados acabo mediante flujos de trabajo, computación en la nube y aceleración CUDA de la unidad de procesamiento gráfico general (GPGPU) e Internet de las cosas para recopilar y usar datos capturados por los sensores de las embarcaciones. Este sistema es implementado en Android y basada

en un Motor de flujos de trabajo conocido como FACE-IT Galaxy [52], diseñada para gestionar datos de múltiples fuentes (GPS, brújula, velocímetro, viento y otros sensores ambientales y ecosondas) recopilados automáticamente. Estos datos capturados, posteriormente son enviados a un servidor alojado en la nube de una manera segura y confiable en una base de datos noSQL para su posterior procesamiento. Este procesamiento incluye extraer los datos de la base de datos, preparar los datos para la interpolación, aplicar algoritmos de interpolación y finalmente producir un nuevo conjunto de datos de profundidad actualizado. El objetivo principal del proyecto es utilizar la batimetría como un caso de prueba para estudiar los mecanismos de recopilación de datos de múltiples fuentes de sensores intermitentes y evaluar problemas de incertidumbre. Este trabajo carece de verificabilidad y trazabilidad en las etapas de flujo de trabajo desplegados por FACE-IT y la infraestructura de cómputo depende de un tercero (Amazon Web Services). Posteriormente, el mismo motor de flujos de trabajo descrito anteriormente (FACE-IT), es usado para la predicción y evaluación operativa de la calidad de los alimentos [49]. Los autores describen su aplicabilidad en un escenario para la supervisión de granjas de mejillones en la Bahía de Nápoles, Italia.

En [35] proponen un framework que permite especificar y gestionar los flujos de trabajo en un sistema ciber-físico, usando un lenguaje de especificación de flujo de trabajo de alto nivel conocido como Redes de Petri que permite describir procesos de una actividad orquestada, es decir, flujos de trabajo y por otro lado, utilizando contratos inteligentes como método de aplicación del flujo de trabajo para garantizar que las entidades involucradas obedezcan el flujo de trabajo acordado. Sin embargo, en esta investigación no implementan los contratos inteligentes sobre la tecnología de bloques encadenados, en este caso, ellos vinculan muchos contratos de transición (transacciones) agrupados en una secuencia para imponer un flujo de trabajo especificado en las redes de Petri. La unión de estos dos componentes proporciona flexibilidad para las entidades que interactúan dentro de un flujo de trabajo. Aunque la principal motivación de este trabajo es asegurar la integridad de los flujos de trabajo en las aplicaciones IoT, los autores descartan la confidencialidad de los datos personales en las aplicaciones IoT y solo dan importancia a la disponibilidad e integridad

de los procesos de ciberseguridad. Finalmente, indican como su propuesta puede ser utilizada para implementar dos casos de uso del mundo real de aplicaciones IoT, uno en la fabricación inteligente y el segundo en la automatización de edificaciones.

En [55] realizan un estudio de las ventajas de la tecnología de flujo de trabajo para el mundo de IoT y analizan cómo los flujos de trabajo permiten rastrear el estado de varios procesos, haciendo énfasis en los procesos comerciales de un negocio, con el fin de detectar si una violación del acuerdo de nivel de servicio (SLA, por sus siglas en Inglés) ha sucedido o está a punto de suceder, y de este modo tomar medidas adecuadas. Adicionalmente, dan las pautas de cómo definir formalmente flujos de trabajo de IoT y cómo estos flujos pueden desencadenarse por los mensajes de IoT de bajo nivel. Finalmente, los autores diseñaron e implementaron un sistema de administración de flujo de trabajo, WFMS con el estándar BPMN (Business Process Model and Notation). El diseño incluye dos fases: análisis y ejecución. Durante el análisis, la definición del proceso especificada en formato BPMN XML se analiza y se convierte en objetos Java. Durante la ejecución, los cambios de estado se realizan en función de los mensajes de IoT recibidos. La experimentación realizada tiene como caso de uso una empresa de mensajería donde el flujo de trabajo se describe en la entrega de los paquetes, este flujo es ejecutado en un motor de flujo de trabajo JBPM [55] (un sistema de flujo de trabajo de código abierto) y el WFMS diseñado para realizar la comparativa, en la cual WFMS obtuvo un rendimiento mucho mayor. Este sistema se encuentra disponible como un servicio web. Sin embargo, no cuenta con interacciones a una base de datos, sino que trabaja en memoria, esto indica que no se tiene un almacén de datos persistente para almacenar el estado del flujo de trabajo y la información se encuentra centralizada.

De estos artículos se pudo obtener información importante referente a las ventajas de usar flujos de trabajo en entornos IoT (tales como obtener una vista del estado de los procesos en tiempo real o la toma de medidas y decisiones basadas en los datos disponibles, etc.), así como aspectos a tener en cuenta en el procesamiento de volúmenes de datos y los requerimientos no funcionales a considerar en la solución que se desea proponer. Estas consideraciones son listadas a continuación:

- Interoperabilidad: En los flujos de trabajo, los procesos desplegados en cada una de las etapas deben tener la capacidad de intercambiar información útil entre ellas.

- Portabilidad: Los procesos que conforman las etapas de un flujo de trabajo pueden ser desplegados y ejecutados en diferentes infraestructuras.

- Modularidad: En sistemas a gran escala, es ideal para el mantenimiento de los sistemas que los componentes que conforman el flujo de trabajo se construyan de forma independiente, es decir, que un componente o proceso no dependa del funcionamiento de otro.

- Flexibilidad: En los flujos de trabajo dada la diversidad de procesos que puede desplegar un usuario según un escenario de estudio dado, se hace necesario poder introducir nuevas características, funciones o procesos a un sistema ya establecido, sin que se cambie o afecte en gran medida el sistema ya desplegado.

- Eficiencia: Dada la necesidad de procesar grandes volúmenes de información (del orden de Gigabytes o mayor), se requieren mecanismos eficientes para el procesamiento de dichos datos que permitan reducir los tiempos percibidos por el usuario de un flujo de trabajo.

- Verificabilidad: Dado que en un flujo de trabajo intervienen diferentes entidades u organizaciones, se requiere tener un control verificable de las operaciones realizadas en las etapas de los flujos de trabajo para determinar el cumplimento u omisión de los procesos acordados por las organizaciones participantes de un flujo de trabajo. Este escenario implica que cada una de las organizaciones de dichos flujos estén de acuerdo con los procesos realizados y que los puedan verificar.

3.2 Grupo 3: Integración de sistemas IoT con los bloques encadenados.

En esta sección se exponen los trabajos de la literatura que se encontraron referentes al uso de la tecnología de bloques encadenados en el Internet de las cosas (IoT). De manera general, se incluyen los trabajos que muestran las amplias oportunidades y escenarios de aplicabilidad de los bloques encadenados en el entorno IoT. De hecho, la comunidad científica y la industria ha previsto una amplia gama de aplicaciones de la tecnología de bloques encadenados en el contexto de IoT, entre las cuales destaca gestión, el control, privacidad y, lo más importante, la seguridad de los dispositivos de IoT. Si bien la investigación a realizar en el presente trabajo no está orientada al uso de bloques encadenados en la seguridad de los dispositivos que conforman el IoT, se da un panorama general de cómo está siendo usada la tecnología de bloques encadenados en el entorno IoT.

Por ejemplo, en [36] se da un panorama de cómo las características intrínsecas de los bloques encadenados son de gran utilidad para el IoT en general, entre las cuales destaca la escalabilidad de la tecnología, al ofrecer más espacio de direcciones que las ofrecidas por IPv6, su aplicabilidad en la identidad y gobernabilidad de los objetos, autenticación, integridad de datos, autorización, privacidad y establecimiento de comunicaciones seguras. En esta investigación, los autores evidencian todos los problemas que engloba el IoT y cómo la tecnología de bloques encadenados puede ser un habilitador clave para resolver esos problemas. Adicionalmente, se hace una revisión de la literatura de la tecnología de bloques encadenados en la que resalta el trabajo [14] en el cual se investiga si los enfoques de estos bloques y *Peer-to-Peer* se pueden emplear para fomentar un IoT descentralizado y privado. Adicionalmente, realizan una revisión sistemática en la cual encontraron 18 casos de uso de bloques encadenados, de los cuales 4 de estos casos de uso están diseñados explícitamente para IoT, en esta revisión categorizan los usos actuales de los bloques encadenados. Como resultado de esta revisión se evidencia los pocos trabajos realizados con respecto al manejo o control de las operaciones

de los datos utilizando bloques encadenados en un entorno IoT. Sin embargo, en [12] describen cómo la combinación de bloques encadenados y IoT puede facilitar el intercambio de servicios y recursos que conducen a la creación de un mercado de servicios entre dispositivos y cómo esta combinación nos permite automatizar de manera criptográficamente verificable varios flujos de trabajo existentes y lentos.

En [33] analizan cómo se puede hacer uso de la tecnología de bloques encadenados para proporcionar seguridad y privacidad en IoT. Como ejemplo, un estudio de caso [22] sobre una casa inteligente permite evidenciar dicha seguridad y privacidad en IoT al usar bloques encadenados. La idea consiste en que cada hogar inteligente está equipado con un dispositivo siempre en línea de alto recurso, conocido como "minero", que se encarga de manejar todas las comunicaciones dentro y fuera del hogar. El minero también conserva bloques encadenados privados y seguro, utilizados para controlar y auditar las comunicaciones. Los autores muestran como el hogar inteligente basado en los bloques encadenados es seguro mediante un análisis exhaustivo de seguridad (confidencialidad, integridad y privacidad) y destacan que los gastos generales (tiempo de procesamiento, tráfico, etc.) son insignificantes en relación con las ganancias de seguridad y privacidad al utilizar la tecnología de bloques encadenados.

En [34] presentan el protocolo Vegvisir, unos bloques encadenados estructurados y acíclicos dirigidos (DAG) tolerante a particiones para su uso en entornos de IoT con restricciones de energía con conectividad de red limitada, que se puede usar para crear un repositorio de datos compartido e inviolable que realiza un seguimiento de la procedencia de los datos. Vegvisir está diseñada específicamente para la configuración de IoT de baja potencia y baja conectividad. Tolera bien las particiones de red y usa un mecanismo de consenso de bajo poder. El costo de esta tolerancia de partición es que los tipos de aplicaciones que se pueden implementar con los bloques encadenados están limitados a aquellos que solo requieren un orden parcial de los eventos registrados. Con este fin, Vegvisir admite aplicaciones basadas en tipos de datos replicados sin conflictos. Los autores presentan como se podría utilizar esta tecnología en escenarios como agricultura digital para el

seguimiento de cadena de suministro de alimentos, en escenarios de respuesta a desastres para actuar oportunamente y a la industria marítima. Los autores tienen una implementación de Android de Vegvisir en construcción. El prototipo de Android está diseñado específicamente para el escenario de primera respuesta de emergencia en atención de desastres. Como tal, permite a los usuarios colocar solicitudes de registros de salud en la cadena de bloques.

Dado que los bloques encadenados son computacionalmente costosos y requieren un gran ancho de banda que no son adecuados para la mayoría de los dispositivos IoT, en [21] proponen una arquitectura liviana basada en bloques encadenados para IoT que elimina virtualmente los gastos generales de los bloques encadenados clásicos, a la vez que mantiene la mayoría de sus beneficios de seguridad y privacidad. La idea es que los dispositivos IoT se benefician de un *ledger* (libro contable) privado inmutable, que actúa de forma similar a los bloques encadenados pero que se gestiona de forma centralizada para optimizar el consumo de energía. Los dispositivos de alto recurso crean una red superpuesta para implementar unos bloques encadenados distribuidos de acceso público que garantiza seguridad y privacidad de extremo a extremo. La arquitectura propuesta utiliza confianza distribuida para reducir el tiempo de procesamiento de validación de bloque. El caso de estudio fue en un entorno de hogar inteligente. La arquitectura fue evaluada cualitativamente bajo modelos de amenazas y los resultados de las simulaciones demuestran que el método reduce significativamente los gastos generales en comparación a los bloques encadenados utilizados en Bitcoin.

En [30] investigan cómo explotar la tecnología de los bloques encadenados para crear aplicaciones distribuidas de economía descentralizada y compartida que permita a las persona monetizar de forma segura sus cosas. Como por ejemplo mecanismos de pago automáticos punto a punto, gestión de derechos digitales, etc. Varias aplicaciones son presentadas en el contexto de una arquitectura de Internet de las cosas utilizando la tecnología de bloques encadenados.

En [60] se define cómo los bloques encadenados puede cambiar la cadena de suministros y la industria de la logística. Ejemplo de una implementación, se lleva a cabo en [68] donde desarrollan un sistema de trazabilidad de una cadena de suministro de alimentos para el rastreo de alimentos

en tiempo real basado en *HACCP* (Análisis de Peligros y Puntos de Control Críticos) [54], bloques encadenados e Internet de las cosas. Este sistema proporciona una plataforma de información para todos los miembros de la cadena de suministro, transparencia, neutralidad, confiabilidad y seguridad.

En [80], describen un sistema descentralizado de administración de datos personales que asegura que los usuarios poseen y controlan sus datos. Implementan un protocolo que convierte una cadena de bloques en un administrador de control de acceso automatizado que no requiere confianza en un tercero. A diferencia de Bitcoin, las transacciones en este sistema no son estrictamente financieras, sino que son utilizadas para llevar instrucciones, como almacenar, consultar y compartir datos.

En [5] presentan una plataforma descentralizada *peer-to-peer* llamada BPIIoT basada en la tecnología de bloques encadenados implementando contratos inteligentes para la Industria del Internet de las cosas. Los autores plantean como BPIIoT puede mejorar la funcionalidad de las plataformas de fabricación basadas en la nube (CBM), al proporcionar una red descentralizada, sin confianza, *peer-to-peer* para las aplicaciones de fabricación. CBM es un modelo de fabricación orientado a servicios en el que los consumidores de servicios pueden configurar, seleccionar y utilizar recursos de fabricación configurables. BPIIoT permite la integración de los equipos heredados del taller en el entorno de la nube y permite el desarrollo de aplicaciones descentralizadas y de fabricación punto a punto. El componente habilitador clave para las máquinas industriales en la plataforma BPIIoT propuesta es el dispositivo IoT. El dispositivo IoT es una solución *plug and play* que permite a las máquinas intercambiar datos sobre sus operaciones a la nube, enviar transacciones a los contratos inteligentes asociados y recibir transacciones de los pares en la red de bloques encadenados. El caso práctico de implementación de la plataforma consistió en el mantenimiento de una máquina y una aplicación de diagnóstico inteligente, contratos inteligentes fueron configurados en una red privada de Ethereum para el servicio de la máquina y reemplazos de piezas. El trabajo realizado indica de manera conceptual el funcionamiento de la plataforma, sin embargo, no se hace ningún tipo de evaluación ni análisis de rendimiento que evidencie la factibilidad y viabilidad de implementar una red de bloques encadenados en el escenario planteado.

En [72] describen PlaTIBART, que es una plataforma para realizar pruebas repetibles a las aplicaciones estándar transaccionales de bloques encadenados en IoT aplicando el patrón Actor (que es un modelo de cálculo comúnmente utilizado en IoT) junto con un lenguaje específico de dominio personalizado (DSL) y gestión de red de prueba de herramientas. Se describe un estudio de caso en el Sistema de Energía Transactivo (TES) que opera en un modo conectado a la red, lo que significa que la red eléctrica local está conectada a un Operador de sistema de distribución (DSO) que proporciona electricidad cuando la demanda es mayor que la que puede generar la red local. PlaTIBART ha sido aplicado para desarrollar, probar y analizar aplicaciones de bloques encadenados de IoT tolerantes a fallas.

En [31] proponen el uso de bloques encadenados para construir el sistema de IoT, ya que con esta tecnología pueden controlar y configurar dispositivos de IoT. Realizan la gestión de claves utilizando criptosistemas de clave pública RSA, donde las claves públicas se almacenan en Ethereum y las claves privadas se guardan en dispositivos individuales. Ethereum fue escogido ya que por medio de sus contratos inteligentes es posible escribir su propio código de Turing completo. De este modo, administran fácilmente la configuración de dispositivos IoT y crean un sistema de administración de claves. Para la prueba de un concepto, se utilizó algunos dispositivos IoT en lugar de un sistema completo de sistema IoT, que consta de miles de dispositivos IoT.

Ethereum también fue elegido en [2], en el cual proponen una arquitectura de seguridad llamada IoTChain que es una combinación de bloques encadenados de autorización basada en el marco ACE (Authentication and Authorization for Constrained Environments) [62] y la arquitectura OSCAR (Object Security Architecture for the Internet of Things) [71] junto con un esquema de clave de grupo, para ofrecer una solución *end-to-end* para el acceso autorizado seguro a los recursos de IoT. Los bloques encadenados proporcionan una forma flexible y confiable de manejar la autorización, mientras que OSCAR usa el libro público para configurar grupos de multidifusión para clientes autorizados. Estos bloques encadenados son implementados sobre una red privada de Ethereum para evaluar la viabilidad de la arquitectura. Además, realizan una evaluación de rendimiento de los

diferentes componentes de la arquitectura.

Otras alternativas para la autenticación de los dispositivos IoT, se presentan en [58, 74]. En el primero, proponen una arquitectura basada en bloques encadenados para las autorizaciones de acceso IoT, se caracterizan con respecto a las otras propuestas por ser tolerante a fallas. En el segundo trabajo, proponen un esquema de autenticación de dos factores fuera de banda para dispositivos IoT basados en la infraestructura de bloques encadenados. Implementan el sistema integrado IoT y bloques encadenados con *Eris Blockchain* y dispositivos informáticos equivalentes para emular dispositivos IoT. Adicionalmente, miden los gastos generales para ejecutar bloques encadenados y los servicios de contrato inteligente en los dispositivos emuladores.

3.3 Grupo 4:Verificabilidad en los procesos realizados en flujos de trabajo mediante bloques encadenados.

En esta sección del estado del arte se expone diversos trabajos que reflejan la combinación de flujos de trabajo con la tecnología de bloques encadenados, estos artículos se exponen y estudian con el fin de observar diferentes características relevantes de la tecnología de bloques encadenados que podrían considerarse en el actual trabajo.

En [24] se evidencia el potencial de los bloques encadenados para servir como infraestructura para la gestión del flujo de trabajo entre las organizaciones. El caso de uso en el cual se realizó el prototipo es de las cartas de crédito que se realizan en un banco alemán. Con la investigación realizada, ellos logran determinar que las ventajas de una solución de bloques encadenados para la administración de flujo de trabajo entre organizaciones son la posibilidad de tener un historial inviolable, permitiendo una verificación de los procesos, la automatización de los pasos del proceso manual y la naturaleza descentralizada del sistema. Para la evaluación del prototipo, los autores identificaron unas áreas en las cuales consideraban podían hacer mejora de los procesos implementados en el prototipo de bloques encadenados, la evaluación de la BDW (del inglés *Blockchain Document WorkFlow*) se

realizó por cada área de mejora, las cuales fueron evaluadas mediante entrevistas semi estructuradas a expertos en el área que dan su opinión respecto a los cambios obtenidos en los procesos, las mejoras mencionadas se relacionan directamente con la digitalización del proceso y las propiedades de los bloques encadenados (inmutabilidad, descentralización, etc.). Sin embargo, el estudio no muestra o evidencia cómo la incorporación de la tecnología de bloques encadenados afecta o no el rendimiento del flujo de trabajo, o si por el contrario se compensa con los beneficios de seguridad obtenidos al utilizar la tecnología. Esto teniendo en cuenta que la tecnología de bloques encadenados proporcionan seguridad y privacidad descentralizadas, pero implica una sobrecarga de energía, retraso y computación considerable [20, 21, 22] que no es adecuada en algunos escenarios.

En [42] mediante la implementación de contratos inteligentes en Ethereum, realizan la ejecución de un flujo de trabajo declarativo distribuido, en un escenario en el cual las partes que intervienen en el flujo de trabajo acordado no confían entre sí. Como resultado de la implementación garantizaron la aplicación de la semántica del flujo de trabajo y un registro incontrovertible de la historia de ejecución del flujo de trabajo. Como inconveniente en la implementación se tiene el costo de las operaciones en Ethereum, el cual debe reducirse al mínimo para las partes honestas, y se debe evitar que el adversario inflige un costo adicional a los demás.

En [32] documentan una aplicación de bloques encadenados desarrollada para el sector inmobiliario que permite la administración del flujo de trabajo distribuido en un proceso de transacción que consiste en la venta de una acción en una empresa de vivienda. Esta investigación utiliza contratos inteligentes de Ethereum para facilitar la interacción entre las partes involucradas y el motivo de dicha aplicación fue comprender el proceso de desarrollo de aplicaciones de bloques encadenados con socios industriales y ver la factibilidad de utilizar contratos inteligentes de Ethereum para aplicaciones en la industria y las finanzas. Finalmente evidencian que las estructuras de flujo de trabajo impulsadas por el mercado, pueden aparecer en cadenas de valor en donde el número de partes es elevado y las fuentes de información son numerosas, pero están desconectadas.

Con respecto a la procedencia segura de los datos que es crucial para la rendición de cuentas, la

informática forense y la privacidad de los datos. En [41], diseñan e implementan ProvChain que es una arquitectura descentralizada y fiable para recuperar y verificar la procedencia de datos en la nube utilizando la tecnología de bloques encadenados, la cual proporciona registros inviolables, permiten la transparencia de la responsabilidad de los datos en la nube y ayudan a mejorar la privacidad y disponibilidad de la procedencia de los datos. Esta arquitectura opera en un flujo de trabajo de tres fases que son la recopilación, el almacenamiento y la validación de los datos de procedencia. El escenario utilizado es el almacenamiento en la nube y selección del archivo de la nube como una unidad de datos para detectar operaciones del usuario para recolectar datos de procedencia. Los resultados de la evaluación del desempeño demuestran que ProvChain brinda características de seguridad que incluyen una procedencia a prueba de manipulaciones, privacidad del usuario y confiabilidad con bajos gastos generales para las aplicaciones de almacenamiento en la nube. La implementación de ProvChain que cuenta con preservación de la privacidad del usuario y una mayor disponibilidad. No considera el requisito funcional de tolerancia a fallos y la evaluación realizada indica que entre mayor es el tamaño de los archivos, se tiene una mayor sobrecarga de utilizar ProvChain (6.49 % adicional), a pesar de que el rango de archivos que fue evaluado es bastante pequeño (1KB a 2MB). Los autores utilizaron Apache Jmeter[1] para evaluar el rendimiento de la aplicación en lugar de poner a prueba la implementación en un escenario real o caso de uso.

La revisión de los trabajos previos da una visión más amplia de cómo la tecnología de bloques encadenados está siendo utiliza para administrar y ejecutar flujos de trabajo en diferentes escenarios.

3.4 Resumen del estado del arte

A partir de los artículos expuestos se puede concluir que la tecnología de bloques encadenados brinda múltiples ventajas y beneficios que deben ser considerados en el desarrollo de la investigación. Se evidenció que son escasos los trabajos que realicen la adquisición de los datos de un entorno IoT

[1] https://jmeter.apache.org/

y realicen flujos de trabajo de procesamiento sobre los datos en la nube junto con la implementación de bloques encadenados para el control de las operaciones realizadas. Sin embargo, se muestra cómo se han integrado los flujos de trabajo en el mundo IoT y, por otro lado, cómo se integra con la tecnología de bloques encadenados, en las cuales se presentan diferentes necesidades que para ser suplidas requieren de un minucioso estudio. Un resumen de las principales ventajas/desventajas de los trabajos relacionados en el contexto de la solución propuesta en este trabajo de investigación se presenta en las Tablas 3.1, 3.2, 3.3 y 3.4.

Construcción de flujos de trabajo					
Autor	Titulo	Año	Aportación	Ventaja	Desventaja
Montella et al.[50]	DagOn*.	2018	Despliegue de flujos de trabajo masivos en una infraestructura heterogénea.	Interoperabilidad y paralelización de tareas	Declaración de tareas y generación de dependencias de forma unitaria, no tiene registros verificables.
Babuji et al.[4]	Parls	2018	Despliegue de flujos de trabajo masivos en una infraestructura heterogénea.	Interoperabilidad, Tolerancia a fallos, Administración automatizada	No tiene registros verificables.
Gonzalez-Compean et al.[27]	Sacbe	2018	Arquitectura para el almacenamiento, procesamiento, compartición y seguridad de los datos.	Diseño modular y flexible	No tiene registros verificables.
Deelman et al.[18]	Pegasus	2005	Despliegue de flujos de trabajo científicos en una infraestructura heterogénea.	Interoperabilidad	No tiene registros verificables.
Majithia et al.[43]	Triana	2004	Despliegue de flujos de trabajo complejos en una red Grid o P2P mediante una GUI	Interoperabilidad, Usabilidad	No tiene registros verificables.

Tabla 3.1: Resumen del estado del arte (Parte 1: Construcción de Flujos de trabajo)

Flujos de trabajo en IoT					
Autor	Titulo	Año	Aportación	Ventaja	Desventaja
Sanchez et al.[61]	DagOn* IoT	2019	Orquestación de un entorno IoT mediante el motor de flujos de trabajo DagOn*[50]	Interoperabilidad e interacción con sensores IoT	Declaración de tareas y generación de dependencias de forma unitaria. No tiene registros verificables.
Gonzalez-Compean et al.[28]	FedIDS	2018	Una arquitectura para construir plataformas geoespaciales confiables	Disponibilidad, integridad y confidencialidad de los datos	No tiene registros verificables.
Montella et al.[51]	FairWind	2017	Un sistema para obtener mapas 3-D de alta resolución del fondo del mar	Implementación Móvil	No tiene registros verificables.
Montella et al.[52]	FACE-IT Galaxy	2015	Un motor de flujos de trabajo	Gestiona datos de múltiples fuentes, transmisión segura y confiable de los datos.	No tiene registros verificables.
Montella et al.[49]	FACE-IT Galaxy (Alimentos)	2018	Un sistema para la predicción y evaluación operativa de la calidad de los alimentos.	Permite la integración, procesamiento y simulación de datos.	No tiene registros verificables.
Kasinathan and Cuellar[35]	Redes de Petri	2018	Un marco de gestión para especificar y gestionar los flujos de trabajo en un sistema ciber-físico	Flexibilidad, disponibilidad e integridad de los procesos. Casos de uso: fabricación inteligente y automatización de edificaciones.	No consideran confidencialidad de los datos y no tiene registros verificables.
Mukherjee et al.[55]	WFMS con el estándar BPMN (Business Process Model and Notation)	2017	Un sistema de administración de flujo de trabajo (WFMS)	Implementación en un escenario real (Entrega de paquetes en empresa de mensajería)	Dependiente del lenguaje Java, no tiene registros verificables y no almacena el estado de los procesos

Tabla 3.2: Resumen del estado del arte (Parte 2: Construcción de Flujos de trabajo en IoT)

Integración de sistemas IoT con los bloques encadenados					
Autor	Titulo	Año	Aportación	Ventaja	Desventaja
Karlsson et al.[34]	Vegvisir	2018	Protocolo para crear un repositorio de datos compartido e inviolable	Tolerante a particiones de red, mecanismo de consenso de bajo poder y realiza un seguimiento de la procedencia de los datos.	Implementación en construcción y las aplicaciones que se pueden implementar están limitadas a aquellas que solo requieren un orden parcial de los eventos registrados.
Dorri et al.[21]	Towards an optimized blockchain for IoT	2017	Una arquitectura liviana que elimina virtualmente los gastos generales de los bloques encadenados clásico (Bitcoin)	Optimización del consumo de energía de los dispositivos IoT, seguridad y privacidad *end-to-end*. Caso de estudio: hogar inteligente.	Gestión centralizada del ledger (libro de registros) y acceso público
Tian[68]	Rastreo de alimentos en tiempo real	2017	Un sistema de trazabilidad de una cadena de suministro de alimentos	Confiabilidad, seguridad y visualización de la información	No hay un análisis o estudio que muestre el costo de implementación de los bloques encadenados
Zyskind et al.[80]	Privacidad descentrali-zada	2015	Un sistema de administración de datos personales que asegura que los usuarios poseen y controlan sus datos	Descentralizado y tiene control de acceso automatizado y definición de instrucciones como almacenar, consultar y compartir datos.	No se realiza procesamiento de los datos para la extracción de conocimiento
Bahga and Madisetti[5]	BPIIoT	2016	Una plataforma para mejorar la funcionalidad de las plataformas de fabricación basadas en la nube (CBM)	Descentralización. Implementado en un escenario real: mantenimiento y diagnóstico inteligente de una máquina	Implementación en una red privada de Ethereum, considerando un costo por transacción realizada.

Tabla 3.3: Resumen del estado del arte (Parte 3: Integración de sistemas IoT con los bloques encadenados)

Verificabilidad en los procesos realizados en flujos de trabajo mediante bloques encadenados.					
Autor	Titulo	Año	Aportación	Ventaja	Desventaja
Fridgen et al.[24]	Gestión de flujo de trabajo entre organizaciones utilizando bloques encadenados.	2018	Prototipo de bloques encadenados para la gestión de flujos de trabajo entre organizaciones	Auditabilidad y la automatización de procesos. Así como registros inmutables y descentralizados. Caso de uso: Banco Alemán	Dependiente del caso de uso. No hay generalidad en la implementación del prototipo.
Madsen et al.[42]	Colaboración entre adversarios.	2018	Prototipo de bloques encadenados para la ejecución de flujos de trabajo distribuido donde las partes no confían entre si.	Registros inmutables y ejecución de la semántica del flujo de trabajo.	Costo de operaciones en la red de verificabilidad (Ethereum)
Hukkinen et al.[32]	Gestión de flujos de trabajo distribuidos con contratos inteligentes	2017	Aplicación que muestra la factibilidad de los contratos inteligentes para ejecutar un flujo de trabajo en la venta de una acción en una empresa de vivienda	Registros inmutables y autoridad de confianza descentralizada	Costo de operaciones en la red de verificabilidad (Ethereum) y dependiente del caso de uso. No hay generalidad en la implementación del aplicativo.
Liang et al.[41]	ProvChain	2017	Una arquitectura descentralizada y fiable para recuperar y verificar la procedencia de datos en la nube	Registros inmutables, privacidad de los datos y descentralización de confianza. También realizan evaluación de rendimiento.	Especifico para las aplicaciones de almacenamiento en la nube.

Tabla 3.4: Resumen del estado del arte (Parte 4: Verificabilidad en los procesos realizados en flujos de trabajo mediante bloques encadenados.)

4

Marco de gestión basado en bloques encadenados

En este capítulo se describe la solución propuesta para proveer verificabilidad en las transacciones realizadas en los modelos de procesamiento de datos basados en entrega continua, los cuales son usados en flujos de trabajo. Tal como se ha descrito anteriormente, la solución propuesta integra bloques encadenados con los flujos de trabajo.

4.1 Introducción

El procesamiento de datos en escenarios IoT se realiza mediante flujos de trabajo que involucran en cada etapa aplicaciones de propósito específico. Tales etapas se organizan en la forma de líneas de producción que comúnmente inicia en una fuente de datos representativa de la adquisición de información por parte de sensores IoT. El procesamiento de datos en dicha etapa depende del caso de uso particular. El flujo de trabajo concluye en la etapa procesamiento que se ejecute en un almacén de los datos recolectados. Tal como fue descrito en capítulo 2 del presente trabajo de investigación, los componentes de un flujo de trabajo particular comúnmente se despliegan en un

entorno totalmente distribuido como por ejemplo la infraestructura de nube. En este contexto, debido a los elementos y actores que intervienen en un flujo de trabajo en un entorno de nube, uno de los desafíos tecnológicos consiste en disponer de mecanismos que permitan controlar de forma eficiente las transacciones realizadas en cada una de las entregas de las etapas individuales de un flujo de trabajo. El control de dichas transacciones podría resolver diferendos o eventos de repudio entre los participantes de un flujo de trabajo en el caso de la interrupción o alteración de la entrega continua, la cual debería estar garantizada en este tipo de esquema de procesamiento.

En este trabajo de investigación se propone hacer frente a la problemática de realizar verificación de transacciones en la entrega continua de datos/meta-datos de un flujo de trabajo mediante un gestor de verificabilidad continua para entrega continua llamado CD/CV (**C**ontinuous**D**elivery/**C**ontinuous**V**erifiability). Por verificabilidad continua, nos referimos a la acción de registrar las transacciones realizadas por cada etapa de un flujo de trabajo.

Para poder realizar esta acción de registro de transacciones, el gestor CD/CV propuesto en esta Tesis, modela las etapas de un flujo de trabajo con un modelo de extracción, transformación y carga conocido como ETC[1] (ver Fig. 1.3 en la sección 1.2). Este modelo comúnmente se utiliza para analítica de datos y para el traslado de los mismos entre diferentes bases de datos. Sin embargo, en este trabajo de Tesis se realiza una adaptación del modelo ETC para mantener el control sobre las transacciones de cada etapa de un flujo de trabajo.

El resultado de la aplicación de este modelo a las etapas de un flujo de trabajo es información sobre estas tres tareas (extracción, transformación y carga), las cuales se asume sean realizadas en cada etapa para cada interacción con otras etapas, la fuente de datos o el almacén de datos (ver un ejemplo de un modelo ETC aplicado a un flujo de trabajo en Fig. 4.1). Si esta información se ingresa a un red de verificabilidad en la forma de transacciones, como lo son los bloques encadenados [3, 73] y se asume que cada etapa se basa en el modelo ETC, entonces los bloques encadenados realizarán la difusión de las transacciones a cada organización a cargo de una etapa, las cuales podrán dar

[1]El equivalente en inglés es ETL (extract, transform and load) [70].

seguimiento a las transacciones acordadas en un flujo de trabajo.

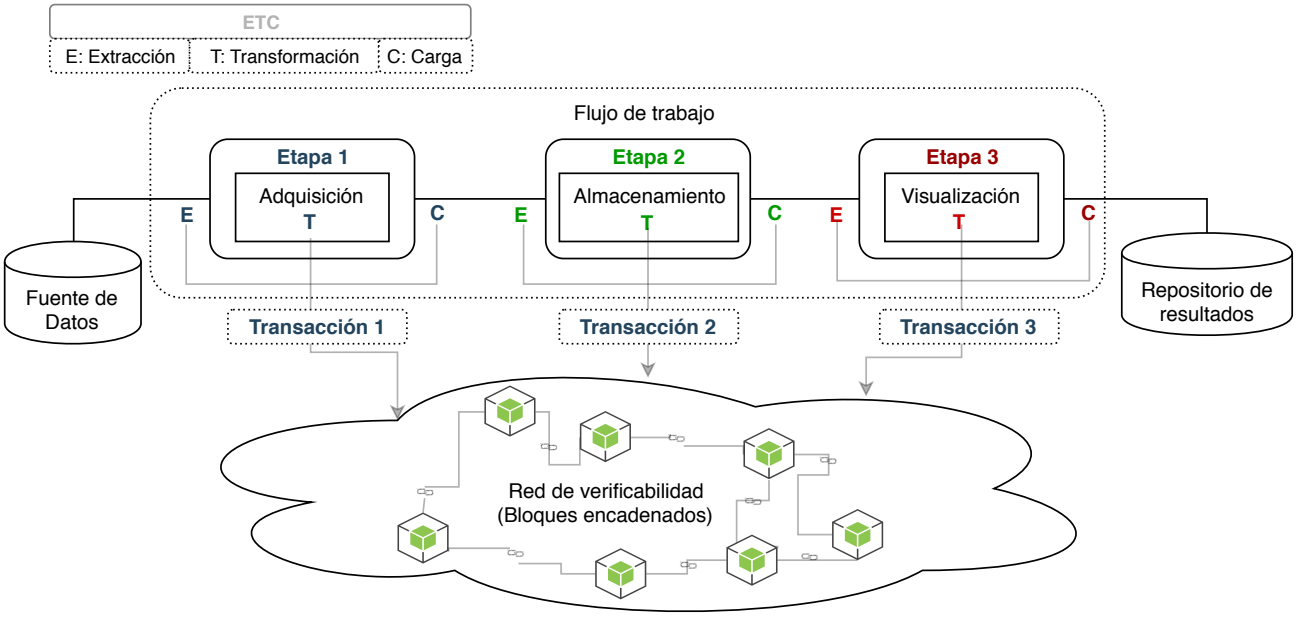


Figura 4.1: Modelo ETC aplicado a un flujo de trabajo.

Aplicando este modelo a flujos de trabajo se obtiene verificabilidad continua a la entrega continua (CD/CV) realizada en un flujo de trabajo. La implementación de el modelo CD/CV propuesto en este trabajo de investigación impone los siguientes desafíos:

- Garantizar que una etapa aplica el modelo ETC a sus transacciones, lo cual es clave para evitar que una etapa intencionadamente o involuntariamente omita el registro de transacciones o altere el contenido de las mismas cuando éstas sean registradas en los bloques encadenados.

- Acoplar el flujo de trabajo a una red de verificabilidad en forma transparente y automática, lo cual resulta clave para la difusión de las transacciones.

- Reducir el costo de la operación de la solución propuesta que ha de ser absorbido por las organizaciones. Estos costos se pueden expresar en términos de la carga de trabajo subyacente producida por la red de verificabilidad, la cual podría afectar a las etapas de procesamiento de

un flujo de trabajo. Este problema específico resulta clave para la factibilidad de la solución propuesta.

4.2 Solución Propuesta

En la Fig. 4.2 se muestra la metodología definida para la solución propuesta en la presente investigación para hacer frente a los desafíos mencionados anteriormente. En la primera etapa de esta metodología se realiza una preparación que incluye un esquema declarativo en el cual todas las etapas que quieran participar en un flujo de trabajo expresan o declaran su ETC.

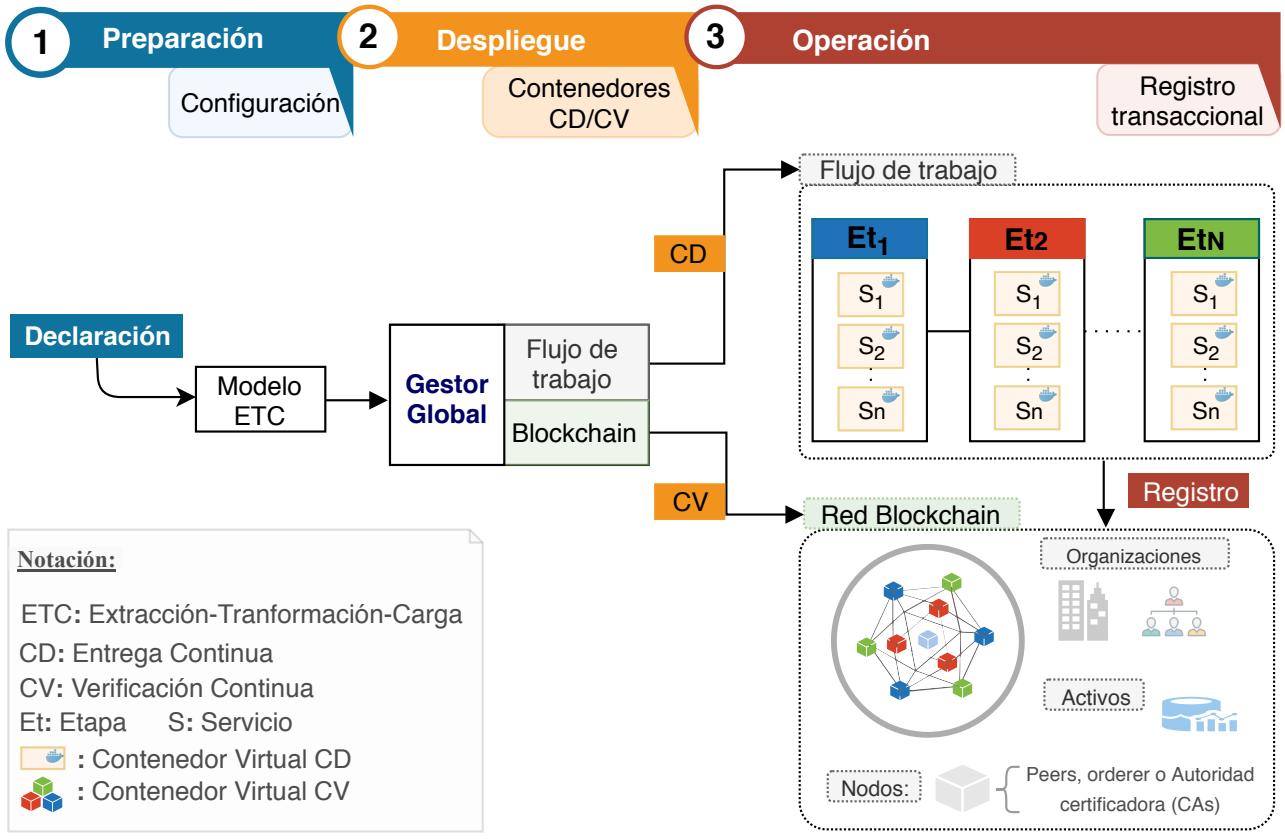


Figura 4.2: Metodología de la solución propuesta.

Para garantizar que este ETC sea efectivamente ejecutado, la declaración se envía a un gestor que ejecuta el flujo de trabajo junto con algunos componentes que garantizarán que el ETC declarado

sea extraído de las etapas e inyectado en la red de verificabilidad, lo cual se realiza en la etapa de despliegue, en el cual se considera la entrega continua (CD) y su espejo de verificación continua (CV).

Finalmente, para hacer frente al desafío número tres se utiliza la virtualización por software, ya que esto permite controlar el lanzamiento tanto del flujo de trabajo como de la red de verificabilidad, se pueden lanzar clones de las etapas y manejarlas como clusters en forma de patrones paralelos (p.e Manager/Worker), los cuales mejoran el rendimiento de la etapa y se piensa que se puede compensar el costo de la red de verificabilidad.

En las siguientes secciones del presente capítulo, se describen cada uno de los componentes y sus respectivas fases de la solución propuesta.

4.3 Gestor global CD/CV

Para construir y operar un marco CD/CV se requiere de un Gestor para cada una de estas fases (construcción y operación). En este sentido, el Gestor Global como se muestra en la Fig. 4.3 se conforma de un Gestor de construcción y un Gestor de operación.

El Gestor Global es básicamente la principal contribución de esta investigación. Este componente incorpora un modelo declarativo de interconexión en el cual se declara el flujo de trabajo a ser desplegado mediante el proceso ETC (Extracción, Transformación, Carga) que se representa como una estructura de datos. Este modelo se describe a continuación.

4.3.1 Modelo declarativo de interconexión (ETC)

El modelo incorporado por el gestor global considera el componente del flujo de trabajo y la red de verificabilidad para poderlos interconectar. Los elementos del modelo que describen el componente del flujo de trabajo son los siguientes:

1. **DS→Data Source:** Representa la fuente original de datos que será procesada a través de un

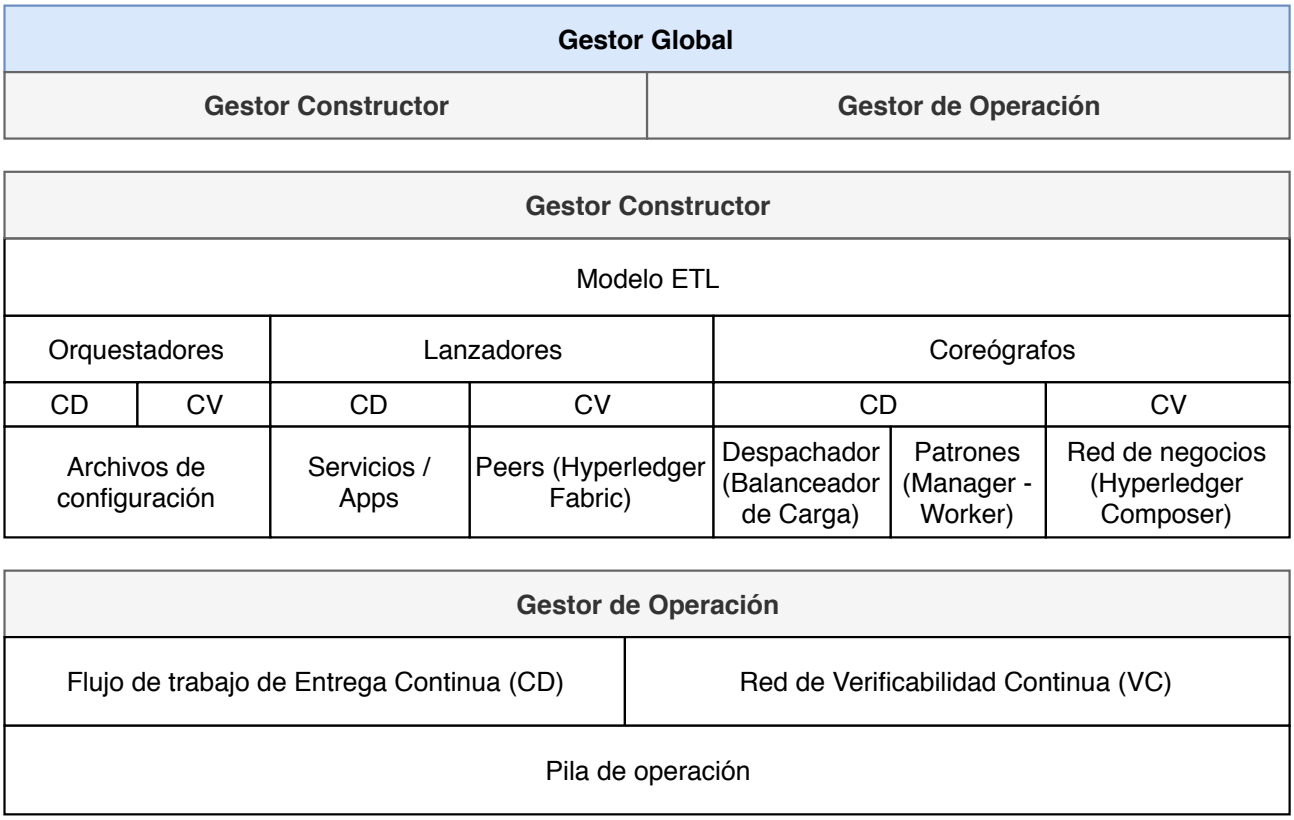


Figura 4.3: Módulos del Gestor Global.

flujo de trabajo (W).

2. **S→Service:** Representa la aplicación del usuario que realiza un determinado procesamiento. (e.g S_1:análisis de datos, S_2 : compresión de datos, S_3 : cifrado,etc.)

3. **St→Stage:** Representa el conjunto denotado como $St = \{S_1, S_2, S_3, ..., S_m\}$ donde m es la cantidad de servicios (S) que conforman la etapa (St). En una etapa cada uno de los S_i son servicios que aplican un determinado procesamiento en común.

4. **W→Workflow:** El flujo de trabajo describe el conjunto de etapas (St) que de manera colaborativa realizan un procesamiento general para transformar la fuente original de datos (DS) en información útil para entidades u organizaciones que requieran tomar decisiones a partir de la información encontrada. Un flujo de trabajo se denota como $W = \{St_1, St_2, ..., St_N\}$

donde N es el número de etapas de procesamiento, las cuales se encuentran interconectadas a través de un proceso ETC. Cada etapa St_i fue descrita en el inciso anterior.

5. **ETL→ <Extract,Transform,Load:>** El proceso ETC (o ETL por sus siglas en inglés) le permite a un servicio (S_i) obtener datos de una fuente determinada (DS) aplicarles un procesamiento dado y cargarlos en un sistema de almacenamiento o espacio determinado, de tal manera que el resultado entregado por S_i sea la entrada de un siguiente servicio (S_{i+1}) con el fin de realizar el mismo proceso, formando de esta manera un flujo de trabajo (W). Este proceso ETC se define por cada $S_i \in St \wedge St \in W$ y consta de las siguientes fases:

   a) **E→Extract:** Indica dónde extraer o adquirir los datos que serán la entrada de un servicio S_i. En esta fase se consideran dos atributos $< type, path >$. Donde $type$ indica el tipo de entrada que recibe S_i: esta puede ser un directorio (folder) o un archivo (file) y $path$ indica la ruta dónde se encuentran los datos a extraer.

   b) **T→Transform:** Indica como S_i procesará los datos. En esta fase se consideran tres atributos $< type, path, parameters >$. Donde $type$ indica el método a través del cual S_i implementa la transformación de los datos, éste puede ser de tipo servicio (service) o un ejecutable (p.e .jar). El segundo atributo $path$ indica la ruta donde se encuentra el método de transformación y $parameters$ son los argumentos que recibe el método de transformación para poder ser ejecutado.

   c) **L→Load:** Indica dónde se entregan los resultados de procesamiento de Si. En esta fase se consideran dos atributos $< type, path >$. Donde $type$ indica el tipo de salida que entrega S_i (folder, file) y $path$ indica la ruta donde se cargan los resultados de procesamiento que se tomarán como entrada del servicio S_{i+1}.

6. **Pa→Patterns:** El modelo considera una serie de patrones para desplegar los flujos de trabajo (W). El patrón considerado en este trabajo de investigación es:

a) **Pa_Ma/Wo → Pattern Manager/Worker:** Este patrón tiene un Manager encargado de repartir la carga de trabajo (datos a procesar) entre N Trabajadores (o Workers). El Manager divide el conjunto de datos original y lo asigna a cada uno de los trabajadores disponibles para que procesen de manera paralela la fuente de datos (DS).

Por otro lado, los elementos del modelo que describen la red de verificabilidad se mencionan a continuación:

1. **O→Organización:** Desde el punto de vista del componente del flujo de trabajo (W), una organización (O) representa una entidad que despliega un conjunto de servicios ($S_1, S_2, ...S_N$). Estos servicios fueron descritos como una Etapa ($Stage \rightarrow St = \{S_1, S_2, ...S_N\}$, ver St) que hace parte del procesamiento de los datos en un flujo de trabajo (W). En este sentido, una Organización es vista como la entidad que realiza una o mas Etapas que pertenecen a un flujo de trabajo en el cual participan múltiples organizaciones.

   Desde el punto de vista de la red de verificabilidad, dicha organización posee un conjunto de Nodos (N) que forman parte de una red de verificabilidad basada en bloques encadenados que contiene un API REST a través del cual una organización registra las acciones que realiza en sus Etapas (St) que pertenecen a un flujo de trabajo (W_i).

2. **N→Nodos**: Representa un nodo de la red de verificabilidad, el cual puede ser uno de los siguientes:

   a) N_{peers}→**Nodos peers**: Son los nodos de la red de verificabilidad que emiten transacciones.

   b) N_{or}→**Nodo orderer**: Es el nodo de la red encargada de establecer un mecanismo de consenso entre los nodos peers al momento de emitir una transacción. El objetivo del nodo ordenador es tener una red de verificabilidad de registros ordenados y consistentes.

c) N_{ca}→**Nodo Autoridad Certificadora**: Es el nodo que emite los certificados y llaves privadas a todos los nodos peers pertenecientes a una determinada organización.

3. **BC→Blockchain:** Es la red de bloques encadenados *peer-to-peer* o P2P en la cual se registran las acciones realizadas por cada servicio S que hace parte del flujo de trabajo verificable.

4. **Tx→Transaction:** Representa una acción realizada por un servicio S en W. La estructura de una transacción (Tx), está en función de la red de negocio (BN) que se haya definido.

5. **BN→Business Network:** Define la lógica de negocio que se traduce en la forma cómo se realizan las transacciones y la abstracción de las entidades que intervienen en dichas transacciones (p.e. activos, participantes).

6. **Ad→Administrator:** Es el ente encargado de gestionar las llaves de acceso de una organización para poder emitir transacciones en nombre de dicha organización (O) a una red de verificabilidad basada en bloques encadenados (BC).

7. **PxO→Peers por Organización:** Este elemento indica cuántos nodos peers (N_{peers}) se van a desplegar por cada organización (O).

Una vez descrito el modelo declarativo de interconexión para el flujo de trabajo y la red de verificabilidad, el Gestor Global, a través de un Gestor Constructor y un Gestor de Operación, puede construir y operar respectivamente un sistema CD/CV.

4.3.2 Gestor constructor CD/CV

El Gestor de Construcción corresponde al primer módulo del marco de gestión propuesto. Este Gestor tiene tres componentes principales: Orquestadores, Lanzadores y Coreógrafos. Estos tres componentes trabajan en conjunto para construir un sistema CD/CV a partir del modelo declarativo ETC descrito en la sección anterior.

4.3.2.1. Construcción del sistema CD/CV

El Gestor Constructor es el encargado de la construcción del sistema CD/CV, la cual se desglosa en dos fases principales: una fase de preparación y una fase de despliegue.

4.3.2.1.1. Fase de preparación: Declaración y configuración. Para preparar el prototipo CD/CV se realizan dos tareas, la primera es una tarea declarativa y la segunda es una tarea de configuración.

Declaración del Modelo ETC .

En esta fase el usuario declara las características de cada etapa del flujo de trabajo. Cada una de las etapas del flujo de trabajo a desplegar, se definen en el modelo de interconexión ETC de la siguiente forma:

$$St = \{St_1, St_2, ..., St_N\}, \text{ donde:}$$
$$N \rightarrow \text{Número de etapas que conforman el flujo de trabajo.}$$

El orden en el cual se definen cada uno de los St_i en St determina el orden a través del cual las etapas se interconectan y los datos son procesados. Es decir, en el ejemplo anterior, la fuente de datos será procesada en primera instancia por la etapa St_1 y entregará sus resultados a la etapa St_2 (ver notación de St). Este proceso se realiza iterativamente hasta llegar a la etapa final del flujo de trabajo.

Una vez que se ha establecido la cantidad de etapas del flujo de trabajo mediante el modelo ETC, el siguiente paso en la fase de preparación consiste en definir el número de servicios que contendrá cada etapa. Se espera que a mayor cantidad de servicios por etapa se mejore el rendimiento del sistema y se compensen la sobrecarga/costos producidos por el componente CV.

La cantidad de servicios y/o aplicaciones se declaran en el modelo de interconexión ETC de la siguiente forma:

El Gestor Constructor CD/CV, a partir de la notación declarada por el usuario mediante la declaración del modelo ETC y la cantidad de servicios desplegados en cada etapa, crea un patrón

$St_1 = Ns_1 \rightarrow$ Número de servicios de la etapa uno.
$St_2 = Ns_2 \rightarrow$ Número de servicios de la etapa dos.
$St_N = Ns_N \rightarrow$ Número de servicios de la n-ésima etapa del flujo de trabajo.

arquitectural para el análisis eficiente de datos provenientes de entornos IoT (ver Fig. 4.4). Este patrón permitirá materializar un flujo de trabajo en la forma de una solución en la nube (análisis como servicio).

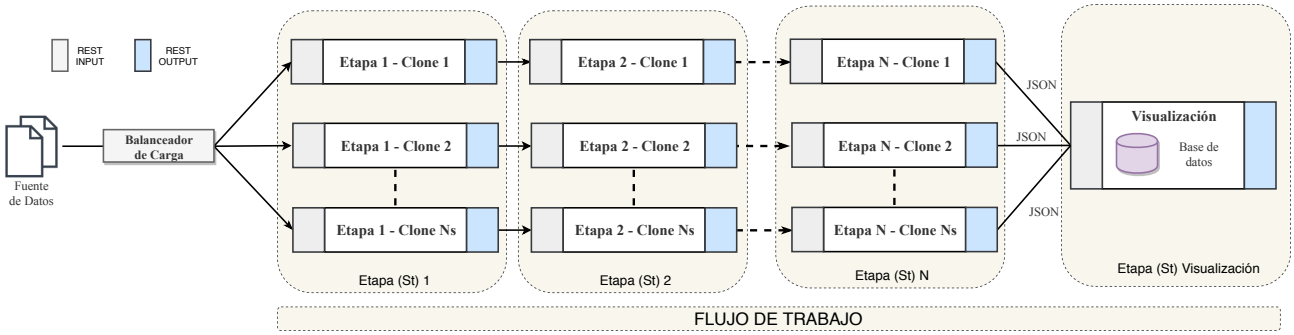


Figura 4.4: Patrón arquitectural del Gestor CD/CV para el análisis/procesamiento de datos provenientes de entornos IoT.

La cantidad de servicios desplegados en cada etapa es el parámetro que determina el número de clones que se muestra en la Fig. 4.4. Dadas las ventajas de encapsular un servicio o aplicación mediante la tecnología de contenedores, es posible desplegar una cantidad variable de servicios por etapa (clones) ya que se asume que cada servicio es encapsulado en un contenedor virtual.

Por otro lado, para el componente de verificabilidad continua (CV), se ha establecido por defecto en este proyecto de investigación, que el número de organizaciones que participarán en la red de verificabilidad, está definido por la cantidad de etapas del flujo de trabajo. Lo anterior se estableció asumiendo que cada etapa del flujo de trabajo realiza un procesamiento diferente. Por ejemplo, en un flujo de trabajo una organización puede encargarse de la adquisición de los datos, una segunda organización realizará el preprocesamiento de los mismos, una tercera analizará los datos y una cuarta organización visualizará los resultados obtenidos. A pesar de lo mencionado, cabe aclarar que es posible asociar múltiples organizaciones a una etapa dada de un flujo de trabajo o múltiples etapas, por ejemplo, a una sola organización y que el componente CV realice, en forma automática,

las acciones de verificabilidad de acuerdo a dicha asociación.

En lo que respecta a los nodos que conformarán la red de verificabilidad, recordemos que existen tres tipos de nodos en la red de verificabilidad (nodo orderer, nodos CAs y nodos peers). En este trabajo de investigación se asume que, por defecto, el esquema CV debe incluir lo siguiente: 1) Un solo nodo orderer, el cual es el único nodo de la red que implementa el mecanismo de consenso necesario para establecer consistencia en las transacciones; 2) Un nodo CA por cada organización, ya que sólo se requiere de una entidad que emita el material criptográfico para cada organización y 3) El número de nodos peers N_{peers} que se determine para cada organización en cada caso. Este parámetro se asume sea variable para propósitos de experimentación.

En función de lo antes mencionado, los parámetros que el usuario final debe declarar son: el número de organizaciones y la cantidad de nodos peers por organización de la red de verificabilidad. Estos parámetros son declarados en el modelo de interconexión ETC de la siguiente forma:

$$PxO = N_{peers}$$
$$Orgs = \{Org1, Org2, ..., OrgN\}$$

Donde:

N_{peers} determina el número de nodos peers para cada organización. En este trabajo de investigación se estableció este número de peers igual para todas las organizaciones con fines prácticos para la experimentación y para manejar un esquema equitativo donde cada organización tiene los mismos privilegios. Sin embargo, cabe aclarar que el número de nodos peers por cada organización puede ser variable.

$Orgs$ define las organizaciones participantes. En este trabajo de investigación se asume que haya una correspondencia entre el numero de etapas del flujo de trabajo y número de organizaciones participantes.

Como resultado final, para el componente de verificabilidad continua, la cantidad total de nodos de la red de verificabilidad, se representa por la siguiente notación:

Por defecto, el marco de gestión configura un $administrador$ para cada organización. Los

$$Nodos = \{Peers_{Org1}, Peers_{Org2}, ..., Peers_{OrgN}, Nodos_{CAs}, N_{or}\}$$
$$Donde:$$
$$Peers_{Org_i} = \{peer0_{Org_i}, peer1_{Org_i}, ..., peerN_{peers_{Org_i}}\} \in Orgi \text{ con } i \in 1, 2, .., N$$
$$Nodos_{CAs} = \{ca_{Org_1}, ca_{Org_2}, ..., ca_{Org_N}\}$$
$$N_{or} = \{orderer\} \rightarrow \text{Nodo para establecer consenso.}$$

administradores son los encargados de agregar nuevos participantes a la sección de la red de verificabilidad asociada a su organización. Este tipo de usuario también se encarga de configurar la red de negocios en cada nodo peer de su organización. Con las llaves privadas y certificados de cada Administrador se despliega posteriormente el API REST que cada organización usará para acceder a los registros transaccionales de la red de verificabilidad.

Configuración del Gestor Constructor CD/CV. En esta sección se describen los pasos realizados para configurar el despliegue tanto del sistema de entrega continua como de la red de verificabilidad (sistema CD/CV).

Para realizar esta tarea, el Gestor Constructor crea archivos de configuración para el despliegue del flujo de trabajo en una infraestructura dada. Estos archivos son colocados por el gestor en un conjunto de rutas que serán utilizadas para desplegar y poner en operación un flujo de trabajo.

En el componente de entrega continua (CD) se definieron las siguientes rutas:

- Fuente de datos: Esta ruta apunta a la base de datos que sera procesada en el flujo de trabajo con verificabilidad continua.
- ETC de cada servicio: Tres rutas son definidas en este apartado (descritas en la sección 4.3.1, como E:ruta de extracción, T:ruta de transformación y C:ruta de carga de resultados) para cada servicio desplegado en cada etapa. Estas rutas automáticamente permitirán a los contenedores virtuales de dichos servicios extraer datos, transformarlos y cargar los datos procesados para su posterior utilización.
- Configuración de contenedores virtuales: Esta ruta apunta a la configuración de Docker para cada imagen de contenedor virtual de los servicios considerados en cada etapa del flujo de

trabajo. Esta ruta es usada por el gestor para crear instancias de contenedor de las imágenes de los servicios. Por lo antes mencionado, es posible clonar los servicios dependiendo de un parámetro dado en la configuración de cada servicio (descrito anteriormente como *número de servicios por etapa*) y permite distribuir carga de trabajo a dichos servicios una vez desplegados.

El gestor hace que todas las rutas antes mencionadas apunten a un volumen compartido al cual tienen acceso cada una de las instancias de los servicios desplegados. Este volumen es desplegado sobre el sistema de archivos del servidor o infraestructura definida para desplegar el Gestor CD/CV.

En el componente de verificabilidad continua (CV) se definieron dos rutas en el archivo de configuración del gestor:

- Hiperledger Fabric: Esta ruta apunta a los binarios del Fabric requeridos para la creación del material criptográfico (certificado, llaves públicas/privadas) así como binarios para crear los artefactos de construcción de los bloques encadenados (canal de comunicación de los pares, bloque génesis y gestor de membresías).
- Hiperledger Composer: Esta ruta apunta al archivo de configuración que contiene el modelo de negocio asociado al modelo ETC del flujo de trabajo.

4.3.2.2. Componentes del Gestor Constructor CD/CV.

En esta sección se describe cada uno de los sub-componentes del Gestor Constructor CD/CV encargadas de ejecutar la fase de preparación previamente descrita, junto con las fases de despliegue y operación del sistema CD/CV.

De manera general, el Gestor Constructor recibe la declaración realizada por el usuario y lanza *Orquestadores* que se encargarán de generar a partir de dicha declaración, nuevos archivos de configuración necesarios para que los *Lanzadores* desplieguen el Sistema CD/CV y posteriormente los *Coreógrafos* puedan aplicar la carga de trabajo (datos y transacciones) a dicho sistema. Cada uno de los componentes del Gestor Constructor se describen a continuación.

4.3.2.2.1. Orquestadores Los Orquestadores son los encargados de crear los archivos de configuración y artefactos necesarios para desplegar el Sistema CD/CV. En este sentido, los orquestadores realizan la etapa de preparación en el Gestor Constructor.

Se consideran dos Orquestadores, un Orquestador CD encargado de la entrega continua y un Orquestador CV designado para la verificación continua.

Orquestador CD. Este componente crea la configuración necesaria que permita sincronizar las aplicaciones para que se comuniquen entre si. El Orquestador CD recibe la representación del grafo por parte del Gestor y ordena a un Lanzador CD (descrito en el siguiente componente) que lo despliegue con unos puertos de entrada y salida en una infraestructura determinada.

Orquestador CV. Este componente se encarga de generar dos tipos de artefactos. El primer tipo corresponde a los artefactos lógicos necesarios para desplegar la red de verificabilidad. Un ejemplo de estos artefactos incluye los archivos o registros donde se almacenan las transacciones que conforman los bloques encadenados. El segundo tipo de artefactos corresponden a la lógica de negocio. Estos artefactos definen el modelo de negocio que se va implementar en la red de verificabilidad. Por ejemplo, el modelo de negocia en una red podría estar definido por participantes, activos e intercambios de esos activos entre los participantes.

4.3.2.2.2. Lanzadores. Los Lanzadores leen los archivos de configuración, que son una notación particular creada por los Orquestadores y despliegan en una determinada infraestructura (ej: cluster) unidades de servicio, a las cuales llamaremos Contenedores Virtuales CD/CV.

Un *Contenedor Virtual* CD/CV básicamente se compone de una Unidad de procesamiento, una unidad de control e interfaces de entrada y salida (Sistema de archivos, memoria o red) a través de las cuales se reciben las solicitudes de procesamiento y se emiten los resultados. La estructura base de un Contenedor Virtual CD/CV se ilustra en la Fig. 4.5.

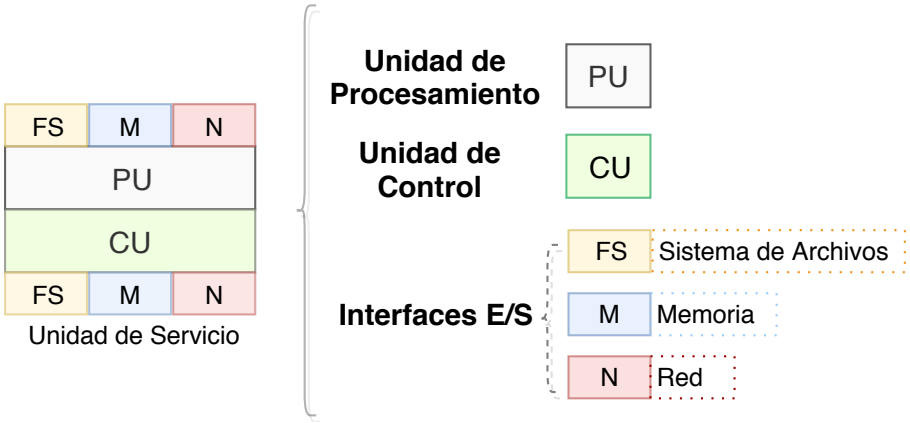


Figura 4.5: Estructura base de un Contenedor Virtual CD/CV (Unidad de Servicio)

Lanzador CD. Despliega Contenedores Virtuales CD. Un Contenedor Virtual CD corresponde a una aplicación o servicio que equivale a una etapa del flujo de trabajo descrito en el modelo ETC.

Lanzador CV. Despliega Contenedores Virtuales CV. Un Contenedor Virtual CV corresponde a un Nodo en la red de verificabilidad.

4.3.2.2.3. Coreógrafos. Una vez que los Contenedores Virtuales CD/CV son desplegados se requiere tener un componente que los controle, a este componente lo denominamos el Coreógrafo.

Un Coreógrafo en primera instancia debe registrar los servicios desplegados mediante un Descubridor de servicios. Creando de esta forma una lista de servicios disponibles que pueden ser consultados mediante un *API*.

Posteriormente, el Coreógrafo debe inyectar la carga al Sistema CD/CV, la cual puede ser una carga de datos a procesar en un flujo de trabajo o una carga de transacciones a registrar en la red de verificabilidad.

De igual forma que los componentes anteriores, se tiene un Coreógrafo CD para controlar los Contenedores Virtuales CD y un Coreógrafo CV para controlar los Contenedores Virtuales CV.

Coreógrafo CD. Un Coreógrafo CD es el despachador de la carga de datos en un flujo de trabajo. Para realizar esta distribución de los datos, el Coreógrafo incorpora un Balanceador de Carga y un patrón Manager/Worker los cuales se describen en la sección de implementación.

Coreógrafo CV. Un Coreógrafo CV controla los Contenedores Virtuales CV e inyecta la carga, la cual corresponde a las transacciones que se requieren registrar en la red de verificabilidad.

Una vez que el Gestor de Construcción tiene todos estos componentes creados y desplegados se tiene un Sistema CD/CV que básicamente es un flujo de trabajo de entrega continua (CD) y una red de verificación continua (CV) en operación, estado en el cual interviene el Gestor de Operación.

4.3.3 Gestor de operación

El Gestor de Operación corresponde al segundo módulo del marco de gestión propuesto. Este módulo se encarga de gestionar el funcionamiento correcto del flujo de trabajo y de la red de verificabilidad. Es decir, que el Gestor de Operación debe garantizar que se realice la entrega continua de los datos en el flujo de trabajo y también de la verificación continua de las transacciones realizadas. La Fig. 4.6 ilustra de manera general el Marco de Gestión una vez que está en operación.

4.3.3.1. Flujo de trabajo de entrega continua (CD)

Una vez que el Marco de Gestión esta en operación (ver Fig. 4.6) se tiene un flujo de trabajo en el cual se ha desplegado un número N de etapas que contiene n Contenedores Virtuales CD (unidades de servicio y/o aplicaciones) que procesarán una fuente de datos y extraerán, a partir de ella, conocimiento o información útil para la toma de decisiones en organizaciones o entidades interesadas. Dicha información relevante se visualiza en un módulo de exhibición (por ejemplo, un servicio web que muestre gráficas estadísticas de la información obtenida).

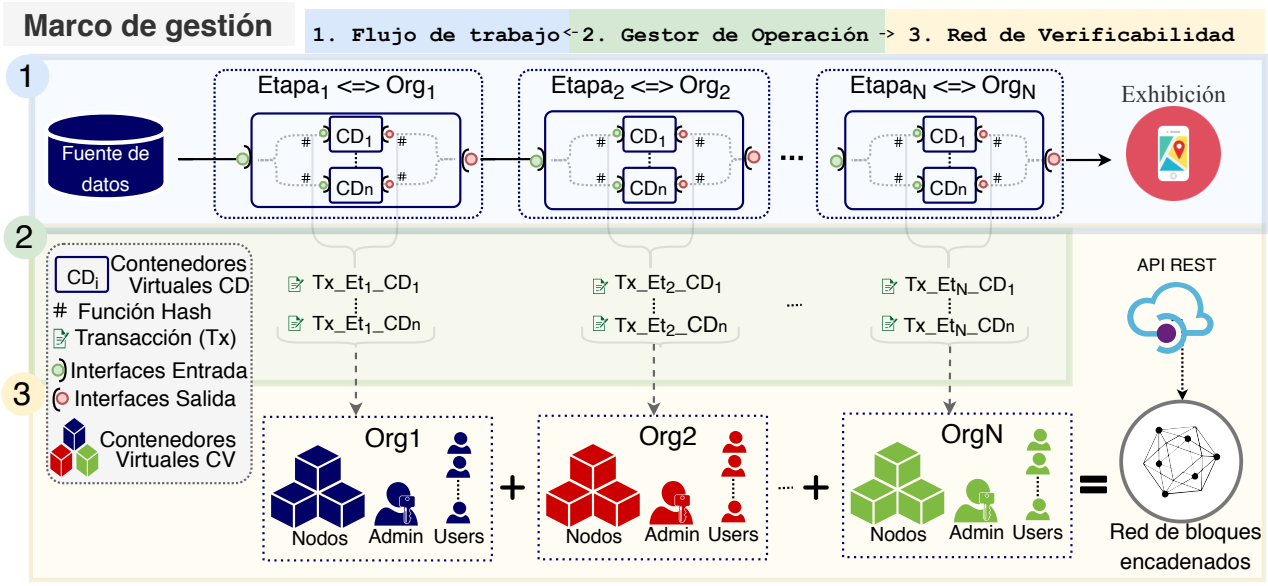


Figura 4.6: Marco de gestión CD/CV en estado de operación.

4.3.3.2. Red de verificabilidad continua (CV)

Cuando el Marco de Gestión se encuentra en estado de operación el componente de verificabilidad continua puede registrar transacciones (Tx) provenientes del flujo de trabajo en la red de verificabilidad.

Una transacción proveniente del flujo de trabajo indica que un conjunto de datos ha sido procesado por una etapa a través de un Contenedor Virtual (CD). Para realizar dichas transacciones el Marco de Gestión incorpora un módulo con funciones resumen (o comúnmente conocidas como funciones *hash*). Dichas funciones generan una serie de caracteres para una entrada de datos dada (en este caso, el contenido de datos a procesar o que fue procesado por una etapa en el flujo de trabajo) que permite la identificación de dicho contenido en un momento dato. Este mecanismo garantiza que cualquier alteración en la entrada de una transacción, incluso el cambio más mínimo, dé como resultado el cálculo de un valor diferente del *hash*, lo que indica la entrada de una transacción potencialmente en riesgo. En este sentido, mediante estas funciones se tiene integridad de las transacciones que son emitidas durante la ejecución del flujo de trabajo.

Posteriormente, con el *hash* de los datos a procesar o que han sido procesados, los meta-datos que describen el Contenedor Virtual (CD) que realiza el procesamiento y los meta-datos del contenido procesado; se construye la estructura de una transacción y se registra en la red de verificabilidad basada en bloques encadenados.

El contenido de datos que es procesado en el flujo de trabajo y las transacciones registradas en la red de verificabilidad fluyen a través de una Pila de operación que utilizan los Coreógrafos CD/CV para proporcionar entrega continua y verificación continua. Esta Pila de operación se describe a continuación:

4.3.3.3. Pila de operación de los Coreógrafos CD/CV

La Pila de operación (Fig. 4.7) describe cómo fluye la carga (datos o transacciones) a través de los Coreógrafos en el Marco de Gestión.

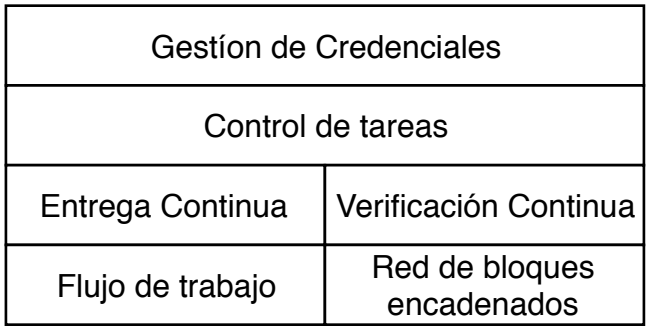


Figura 4.7: Pila de operación del Gestor de Operación.

Cada usuario del marco de gestión debe tener credenciales con el fin de determinar quién realiza la configuración del Marco de Gestión, a qué Organización o entidad pertenece, qué rol tiene: administrador de la red, usuario, etc.

En este módulo se determina qué es lo que se va a procesar, esto puede ser un dato que va dirigido al flujo de trabajo o una transacción que va dirigido a la red de verificación. La Gestión de Credenciales da el acceso al usuario en dos niveles. El primer nivel corresponde al de Construcción. En este nivel el usuario puede construir y desplegar el Sistema CD/CV mediante el Gestor Global.

El segundo nivel corresponde al de Operación. En este nivel el usuario puede emitir transacciones al Sistema CD/CV mediante un *API REST*, aplicativo móvil, etc.

Una vez que el usuario accede al Marco de Gestión puede emitir tareas o instrucciones que pueden ir destinadas a un flujo de trabajo o a la red de verificabilidad. Las tareas en el flujo de trabajo son los datos que requieren ser procesados. Las tareas en la red de verificabilidad corresponden a operaciones tales como consultar registros de los bloques encadenados, crear transacciones, eliminar transacciones, etc., que representan las operaciones aplicadas sobre los datos del flujo de trabajo.

Las tareas descritas anteriormente son controladas por los Coreógrafos (Control de tareas) y éstos determinan qué es lo que se va a procesar (dato o transacción) y a qué parte del Sistema CD/CV va dirigida la tarea. Esta puede ir dirigida al módulo de entrega continua (CD) o al módulo de verificación continua (CV). Si una tarea es asignada al módulo de entrega continua, ésta será procesada por los aplicativos o servicios que conforman las etapas de un Flujo de trabajo. En caso de ser asignada al módulo de verificación continua, ésta será procesada por la red de bloques encadenados o descrita anteriormente como la red de verificabilidad.

4.4 Resumen

En este capítulo fue presentado el Marco de gestión de flujos de trabajo basado en bloques encadenados para la verificación continua de datos en escenarios del Internet de las Cosas. Se describió la principal contribución de esta investigación, la cual se le denomina Gestor CD/CV (ContinuousDelivery/ContinuousVerifiability) o Gestor Global. Este Gestor es el encargado de la construcción y operación del Marco de Gestión, para ello incorpora un modelo declarativo de interconexión en el cual se declara el flujo de trabajo a ser desplegado mediante el proceso ETC (Extracción, Transformación, Carga) que se representa como una estructura de datos. Adicionalmente, fueron descritos cada uno de los módulos del Gestor Global (Gestor Constructor y Gestor de Operación) con sus respectivos componentes (orquestadores, lanzadores y coreógrafos)

y su funcionamiento en las diferentes fases (preparación, despliegue y operación). Se espera que el diseño de la solución propuesta permita generar flujos de trabajo con entrega continua de datos IoT entre las etapas de dichos flujos y que permita adicionalmente, de forma eficiente, realizar la verificabilidad de los procesos o transacciones realizadas en dichas etapas, mediante la incorporación de los bloques encadenados para apoyar el proceso de toma de decisiones.

5

Resultados Experimentales

En esta sección se describe la metodología de evaluación de la solución propuesta para dos escenarios IoT que involucran datos sensibles: movilidad de usuarios y gestión de datos medioambientales.

5.1 Metodología de evaluación

Una evaluación experimental fue conducida en la forma de estudios de caso. Para este trabajo de investigación se consideraron dos escenarios de procesamiento de datos provenientes de entornos del IoT: el primer caso basado en información de movilidad de usuarios y el segundo basado en datos medioambientales capturados por sensores.

En ambos escenarios el manejo de los datos es realizado mediante un procesamiento de entrega continua y se requiere que las transacciones realizadas por las etapas de dichos flujos sean verificadas. A continuación se describe cada uno de los escenarios y la metodología de evaluación que se realizó en cada uno de ellos.

Note que los flujos de trabajo creados en ambos escenarios fueron implementados usando el prototipo descrito en el Capítulo 4.

5.2 Estudio de caso uno: Movilidad de usuarios

En esta sección se describen las actividades que permitieron conducir el estudio de caso sobre la Movilidad de usuarios usando el gestor CD/CV y que se listan a continuación: 1) Las características del estudio de caso; 2) La preparación, despliegue y operación del flujo de trabajo, para el procesamiento de información del estudio de caso usando el Marco de Gestión CD/CV; 3) La variación experimental que permitirá evaluar diferentes configuraciones del Gestor CD/CV y la descripción de la solución del estado del arte que se utilizó para realizar una comparativa de rendimiento; 4) Las métricas para evaluar el rendimiento de dichas soluciones y 5) El análisis y discusión de los resultados obtenidos.

5.2.1 Descripción del estudio de caso: Análisis de Movilidad de usuarios

El estudio de caso se basa en datos sobre movilidad de usuarios capturados a partir de dispositivos GPS. Para conducir este estudio de caso se creó un flujo de trabajo implementando un algoritmo propuesto por Montoliu et. al. [53] que permite la extracción de puntos de interés (o POIs[1], por sus siglas en inglés) de un usuario, a partir de una lista de puntos de ubicación. Este algoritmo fue implementado en la forma de un flujo de trabajo comprendiendo etapas tales como: preprocesamiento, procesamiento, almacenamiento y exhibición (ver Anexo I para detalles sobre la implementación del algoritmo).

Fuente y almacén de datos.

La base de datos Geolife [78] contiene información de trayectorias de 182 usuarios recolectadas

[1]Significa punto de interés de un usuario. Por ejemplo, restaurantes, hoteles, o cualquier lugar que un usuario frecuente.

durante cinco años en la ciudad de Beijing, China. En total cuenta con 18,670 trayectorias correspondientes a 1,292,951 kilómetros. Dentro de los datos recolectados se encuentran la latitud, longitud y estampa de tiempo de cada punto recabado mediante diferentes dispositivos GPS.

La distribución de estas trayectorias por usuario (ver Fig. 5.1) indican que se tiene una distribución de datos heterogénea. Por ejemplo, el usuario con mayor trayectorias de la base de datos (2024) representa el 10.84 % de la base de datos y el usuario con menor cantidad (1) representa el 0.0053 %.

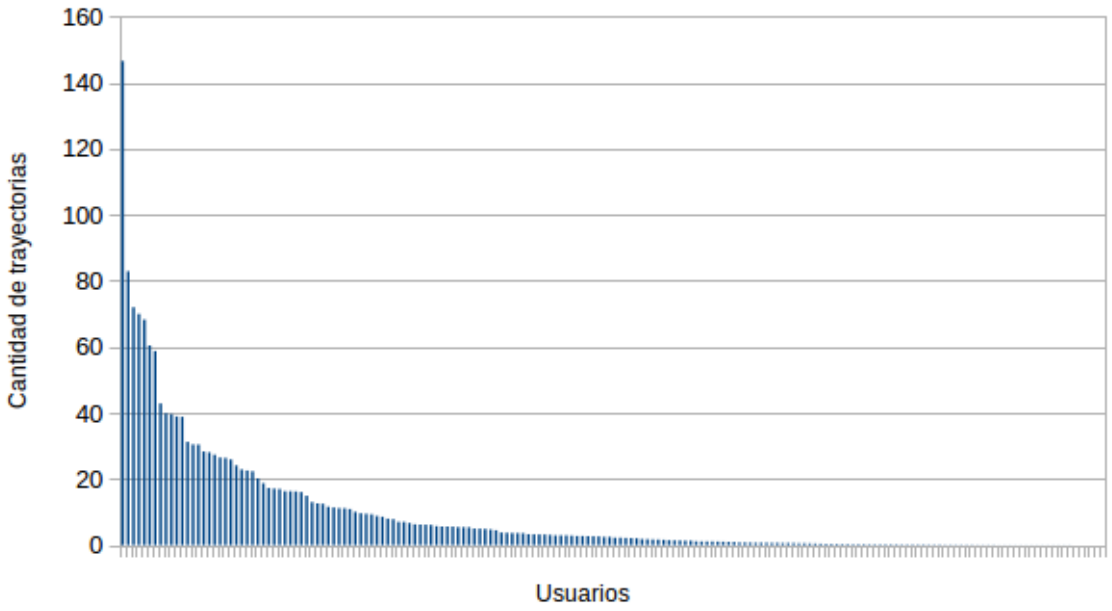


Figura 5.1: Distribución de las trayectorias de los 182 usuarios.

5.2.2 Preparación, despliegue y operación del flujo de trabajo usando el Marco de Gestión CD/CV

En esta sección se describe la preparación del prototipo CD/CV para crear, desplegar y poner en operación el flujo de trabajo que es objeto de estudio.

5.2.2.1. *Preparación del prototipo Gestor CD/CV*

Tal como se mencionó en la sección 4.3.2.1, para preparar el prototipo CD/CV se realizan dos tareas, la primera es una tarea declarativa y la segunda es una tarea de configuración.

5.2.2.1.1. Declaración del Modelo ETC. Las etapas (St_i) del flujo de trabajo (W) para este estudio de caso se definen en el modelo de interconexión ETC de la siguiente forma:

$$W = \{St_1, St_2, St_3\}, \text{ donde:}$$
$$St_1 \rightarrow \text{Etapa de Preprocesamiento}$$
$$St_2 \rightarrow \text{Etapa de Procesamiento}$$
$$St_3 \rightarrow \text{Etapa de Exhibición}$$

El siguiente paso en la declaración del modelo ETC es definir los servicios a ser desplegados. La cantidad de servicios y/o aplicaciones para este estudio de caso se declaran en el modelo de interconexión ETC de la siguiente forma:

$$St_{1,NuSer} = N_1 \rightarrow \text{Número de servicios de Preprocesamiento}$$
$$St_{2,NuSer} = N_2 \rightarrow \text{Número de servicios de Procesamiento}$$
$$St_{3,NuSer} = N_3 \rightarrow \text{Número de servicios de Exhibición}$$

El gestor CD/CV, a partir de la notación declarada por el usuario mediante la declaración del modelo ETC y la cantidad de servicios desplegados en cada etapa, crea un patrón arquitectural para el análisis eficiente de datos GPS (ver Fig. 5.2).

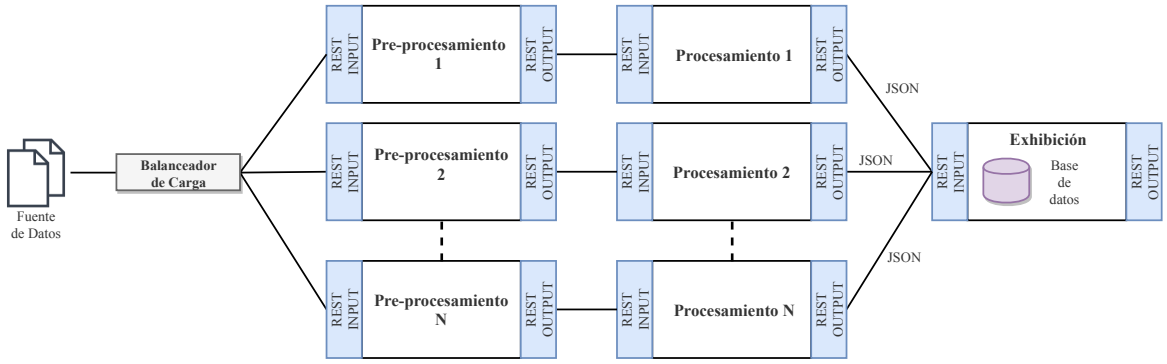


Figura 5.2: Patrón arquitectural propuesto para el análisis de datos GPS.

En este sentido, para desplegar el patrón arquitectural que se muestra en la Fig. 5.2, se define el valor de N_1 y N_2 en la variación experimental (sección 5.2.4) y el valor de N_3 se establece en 1.

Por otro lado, para el componente de verificabilidad continua, tal como se mencionó en la descripción de la solución propuesta, se asume que el número de organizaciones está definido por la cantidad de etapas del flujo de trabajo. Es decir, que el número de organizaciones para este estudio de caso corresponde a tres organizaciones tomando en cuenta las etapas consideradas en el flujo de trabajo que es objeto de estudio. En este sentido, para este estudio de caso se asume que una organización realiza el preprocesamiento, otra realiza el procesamiento y una tercera se encarga de la exhibición de los resultados producidos en el flujo de trabajo.

El segundo parámetro a declarar para el componente de verificabilidad continua corresponde al número de peers N_{peers} para cada organización previamente declarada. Recordemos que el resto de nodos peers para la red de verificabilidad está establecido por defecto, donde se tiene un solo nodo orderer para el mecanismo de consenso y un nodo de entidad certificadora para cada organización, que para este estudio de caso serían tres entidades certificadoras.

Para este estudio de caso, los parámetros mencionados (número de organizaciones y cantidad de nodos peers por organización) se declaran en el modelo de interconexión ETC de la siguiente forma:

$$PxO = N_{peers}$$
$$Orgs = \{Org1, Org2, Org3\}$$

donde:

N_{peers} determina el número de peers para cada organización. La variación de este parámetro se describe en la sección 5.2.4. $Orgs$ define las organizaciones participantes. Dado que en el flujo de trabajo que es objeto de estudio considera tres etapas, la red de verificabilidad incluye tres organizaciones.

El resultado de las definiciones establecidas previamente quedan representadas en el modelo ETC por la notación que se muestra a continuación, junto con una representación visual de la red de verificabilidad conformada para este estudio de caso.

$$Nodos = \{Peers_{Org1}, Peers_{Org2}, Peers_{Org3}, Nodos_{CAs}, N_{or}\}$$
$$Donde:$$
$$Peers_{Org1} = \{peer0_{Org_1}, peer1_{Org_1}, ..., peerN_{peers_{Org_1}}\} \in Org1$$
$$Peers_{Org2} = \{peer0_{Org_2}, peer1_{Org_2}, ..., peerN_{peers_{Org_2}}\} \in Org2$$
$$Peers_{Org3} = \{peer0_{Org_3}, peer1_{Org_3}, ..., peerN_{peers_{Org_3}}\} \in Org3$$
$$Nodos_{CAs} = \{ca_{Org_1}, ca_{Org_2}, ca_{Org_3}\}$$
$$N_{or} = \{orderer\} \rightarrow \text{Nodo para establecer consenso.}$$

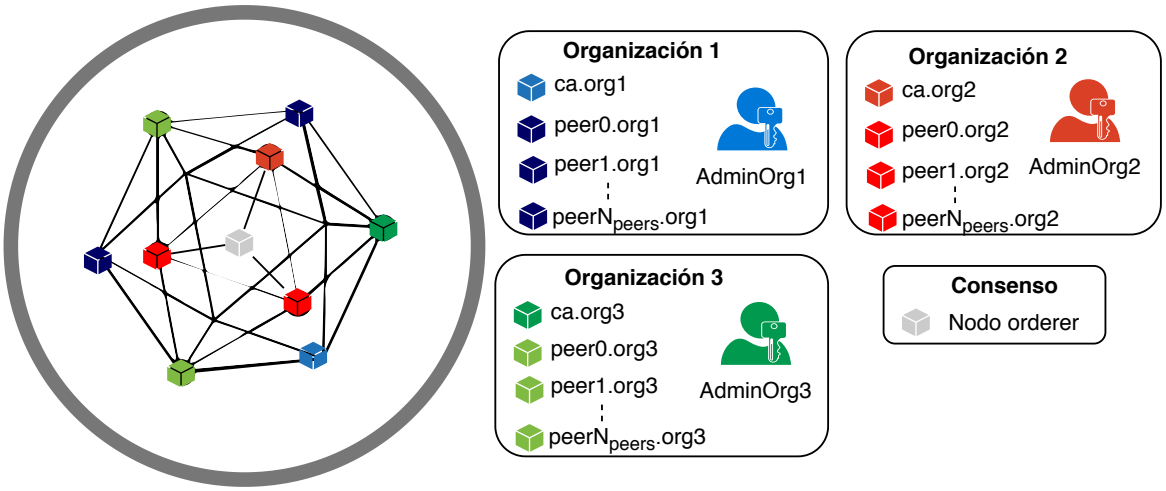


Figura 5.3: Red de verificabilidad para el estudio de caso de movilidad.

Por defecto el marco de gestión configura un *administrador* para cada organización, en este sentido, para este estudio de caso se definieron tres administradores: AdminOrg1, AdminOrg2 y AdminOrg3.

5.2.2.1.2. Configuración del gestor CD/CV. La tarea de configuración en la fase de preparación del sistema CD/CV para este estudio de caso, consistió en definir las siguientes rutas para el componente de entrega continua (CD):

- Fuente de datos: Esta ruta apunta a la base de datos Geolife [78] con la cual iniciará el flujo de trabajo de este estudio de caso.
- ETC de cada servicio: Tres rutas son definidas en este apartado para cada servicio (preprocesamiento, procesamiento y exhibición) desplegado en cada etapa. Por medio de un

servicio web o una configuración manual el usuario define por cada servicio las rutas para extraer los datos, transformarlos y cargar los datos procesados para su posterior utilización.

- Configuración de contenedores virtuales: Esta ruta apunta a la configuración de Docker para cada imagen de contenedor virtual de los servicios considerados en cada etapa del flujo de trabajo de este estudio de caso. Esta ruta es usada por el gestor para crear instancias de contenedor de las imágenes de los servicios de preprocesamiento, procesamiento y exhibición, dado este estudio de caso.

Como se mencionó previamente, el gestor hace que todas las rutas antes mencionadas apunten a un volumen compartido. Este volumen compartido se desplegó sobre el sistema de archivos del servidor descrito en la Tabla 5.1. Con respecto a las rutas del componente de verificabilidad continua (CV) las rutas de Hyperledger Fabric e Hyperledger Composer son establecidas por defecto por parte del Gestor CD/CV y no son dependientes del estudio de caso.

5.2.2.2. Despliegue del flujo de trabajo creado por CD/CV

Las etapas del flujo de trabajo descrito en la sección 5.2.1 fueron encapsuladas en contenedores virtuales usando la plataforma Docker. En este estudio de caso, el prototipo del gestor CD/CV fue desplegado usando la infraestructura hardware y software descrita en las Tablas 5.1 y 5.2 respectivamente.

	Infraestructura Hardware
Servidor	Compute11
Descripción	Asignados a la nube. Cuenta con Docker CE
Procesador	(1) Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz
Memoria RAM	64 GB
Almacenamiento	1 x Disco duro de 2.7 TB 1 x Disco duro de 2.7 TB
Sockets	1
Cores por Socket	12
Threads por core	1
IP	10.0.0.41 148.247.201.223

Tabla 5.1: Infraestructura hardware del estudio de caso uno

Software	
Descripción	**Versiones**
Sistema Operativo con instalación tipo "Compute Node"	Linux CentOS 7 x64
Cloudera	Manager
Docker y Docker Compose	>= 17.06.2-ce
cURL	Latest
Go	1.11.x
Python	2.7.x
Node.js Runtime y NPM	8.9.x
Hyperledger Fabric	1.2
Hyperledger Composer	0.20

Tabla 5.2: Infraestructura software del estudio de caso uno.

La cantidad de servicios (contenedores virtuales) utilizados se define en la sección 5.2.4 en función de la infraestructura hardware descrita en la Tabla 5.1. En este punto el prototipo se encuentra preparado para ser desplegado con diferentes configuraciones sobre una infraestructura dada.

5.2.2.3. Operación del flujo de trabajo creado por CD/CV

Una vez que el flujo de trabajo ha sido desplegado, las etapas comienzan a realizar la extracción de puntos de interés - POIs (ver Fig. 5.4) aplicando los procesos siguientes:

La primer etapa (St_1) consiste en la carga de los datos de movilidad en la plataforma, para ello se propone la siguiente tarea:

- **Obtención y preprocesamiento de los datos:** La fuente de datos descrita en la sección

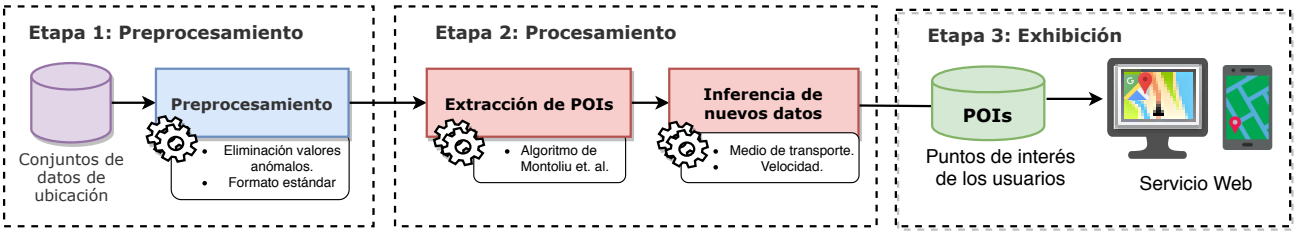


Figura 5.4: Diseño del flujo de trabajo para el estudio de caso uno.

5.2.1 es preprocesada en primera instancia. El preprocesamiento de los datos incluyó un proceso de eliminación de valores anómalos, así como la unificación de formatos y la transformación a una forma estructurada de los datos, como son los archivos JSON.

En la segunda etapa (St_2) se presentan las tareas de procesamiento de los datos. Dentro de las tareas a realizar en esta etapa se encuentran las siguientes:

- **Extracción de POIs.** Para la extracción de POIs se utilizó el algoritmo propuesto por Montoliu et. al. [53], el cual se encuentra descrito en el Anexo II: Implementación Estudio de Caso - Movilidad de usuarios.

- **Inferencia de nuevos datos.** A partir de los POIs extraídos, se infiere nueva información de movilidad del usuario como el medio de transporte entre POIs y la velocidad a la que se movió.

Los POIs calculados son almacenados en una base de datos desplegada en la tercera etapa (St_3). Finalmente, un módulo de exhibición o visualización basado en un servicio web muestra los POIs obtenidos por usuario que solicite el usuario final del flujo de trabajo. Este flujo de trabajo descrito compone el sistema CD/CV desplegado por el Gestor, el cual se puede apreciar en la Fig.5.5.

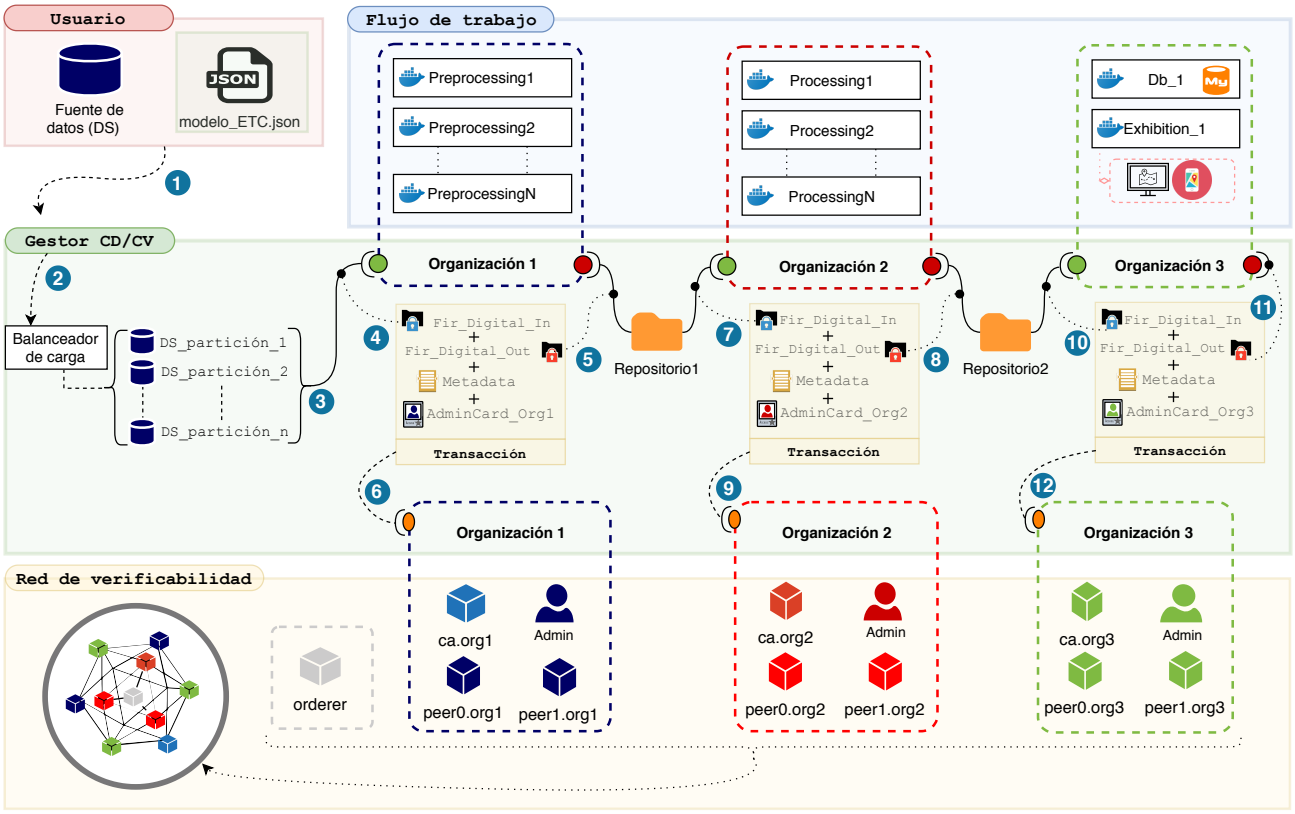


Figura 5.5: Operación del sistema CD/CV para el estudio de caso de movilidad de usuarios.

5.2.3 Soluciones estudiadas

Para evaluar el rendimiento del gestor CD/CV se estudiaron soluciones afines encontradas en el estado del arte descritas en la Tabla 5.3:

En esta evaluación se compara el rendimiento de MG-CD/CV con el producido por MG-CD para observar la sobrecarga generada por el gestor CD/CV. Específicamente se estudia aquella producida por el componente de verificabilidad (CV). En esta misma comparativa se pretende observar el impacto de los patrones de paralelismo en la reducción e incluso eliminación de dicha sobrecarga. La comparación de MG-CD/CV con DagOn tiene como finalidad analizar la sobrecarga del registro de las transacciones en los bloques encadenados cuando ambas soluciones crean flujos que utilizan patrones de paralelismo.

Id Solución Estudiada	Nombre	Descripción
MG-CD	Marco de Gestión con Entrega Continua (CD)	- En esta solución se construye, despliega y opera el flujo de trabajo para la extracción de puntos de interés.
MG-CD/CV	Marco de Gestión con Entrega Continua (CD) y Verificación Continua (CV)	- Se construye, despliega y opera el mismo flujo de trabajo creado por MG-CD pero incluyendo el componente CV que registra cada transacción en la red de verificabilidad.
[50] DagOn*	Engine DagOn	- Es un motor para la construcción de flujos de trabajo (Workflow Engine) dirigido a procesamiento científico y medioambiental. En esta solución se despliega y opera el mismo flujo de trabajo de las soluciones anteriores.

Tabla 5.3: Soluciones estudiadas para la evaluación experimental del estudio de caso uno.

5.2.4 Variación experimental.

En la experimentación del estudio de caso de movilidad de usuarios se realizó una variación sistemática de los parámetros tanto del componente de entrega continua (CD) como de verificabilidad continua (CV). Esta variación se realizó en dos etapas, una exploratoria y otra de comparación.

En la primera etapa de variación de parámetros el objetivo fue explorar el rendimiento del marco de gestión para diferentes cargas de trabajo. Esto considerando que se tiene una distribución de datos heterogénea respecto a la cantidad de trayectorias que tiene cada usuario.

Con los resultados obtenidos en la primera etapa se realizaron ajustes a los parámetros dados con el fin de realizar, en una segunda etapa, una comparativa entre MG-CD, MG-CD/CV y la solución del estado del arte (DagOn).

La variación experimental realizada para cada componente en cada etapa se describe a continuación.

5.2.4.1. Entrega Continua - Etapa Exploratoria.

Para el componente de entrega continua se determinó evaluar los siguientes parámetros en la etapa exploratoria:

$$CD = \begin{cases} \text{Variedad} \rightarrow \text{Servicios por Etapa: } \{1,2,4,6,8,10,12,24\} \\ \text{Volumen} \rightarrow \text{Procesamiento de Datos: } \{1,10,100\} \\ \text{Semilla} \rightarrow \text{Generador de números aleatorios: } \{1,2\} \end{cases}$$

Para la solución MG-CD, el total de configuraciones a evaluar viene dado por los parámetros de *Variedad* en los servicios que se despliegan en cada etapa del flujo de trabajo (S_1,S_2,S_4,S_6,S_8,S_{10},S_{12},S_{24}) y *Volumen* con respecto a la cantidad de datos o subconjunto de datos a procesar en cada flujo de trabajo (D_1,D_{10},D_{100}).

Para este estudio de caso y dada la fuente de datos descrita en la sección 5.2.1, hacemos referencia a la cantidad de datos a procesar como la cantidad de usuarios que se procesarán en el flujo de trabajo. Por ejemplo, si se fija la variable D_{10}, esto indica que se seleccionarán 10 usuarios con sus respectivas trayectorias y mediante un balanceador de carga se asignarán los 10 usuarios entre los S_i servicios definidos para cada etapa del flujo de trabajo con el parámetro de *Variedad*.

La selección de este subconjunto de datos o usuarios se realiza de forma aleatoria uniforme, para ello se utilizaron dos valores de semillas diferentes (1,2) en el generador de números aleatorios, para seleccionar diferentes cargas de trabajo para D_1, D_{10} y D_{100}. En este sentido, considerando todas las posibles combinaciones de los parámetros de *Variedad* y *Volumen* se tiene un total de 24 configuraciones para evaluar el componente de entrega continua para la solución MG-CD con el mismo subconjunto de datos seleccionado para cada configuración de D_i por cada servicio S_i. Por ejemplo, las configuraciones D_1 con S_1,S_2,..,S_{24} toman el mismo subconjunto de datos. Esto se lleva a cabo con el objetivo de realizar una comparativa justa entre las soluciones estudiadas. Por comparativa justa se entiende que cada solución es comparada procesando el mismo subconjunto de

datos de la fuente original la cual es seleccionada fijando un valor de semilla para el generador de números aleatorios. Las posibles configuraciones se muestran en la siguiente Tabla 5.4.

Carga de 1 Usuario	Carga de 10 Usuarios	Carga de 100 Usuarios
$C_1 = (D_1, S_1)$	$C_9 = (D_{10}, S_1)$	$C_{17} = (D_{100}, S_1)$
$C_2 = (D_1, S_2)$	$C_{10} = (D_{10}, S_2)$	$C_{18} = (D_{100}, S_2)$
$C_3 = (D_1, S_4)$	$C_{11} = (D_{10}, S_4)$	$C_{19} = (D_{100}, S_4)$
$C_4 = (D_1, S_6)$	$C_{12} = (D_{10}, S_6)$	$C_{20} = (D_{100}, S_6)$
$C_5 = (D_1, S_8)$	$C_{13} = (D_{10}, S_8)$	$C_{21} = (D_{100}, S_8)$
$C_6 = (D_1, S_{10})$	$C_{14} = (D_{10}, S_{10})$	$C_{22} = (D_{100}, S_{10})$
$C_7 = (D_1, S_{12})$	$C_{15} = (D_{10}, S_{12})$	$C_{23} = (D_{100}, S_{12})$
$C_8 = (D_1, S_{24})$	$C_{16} = (D_{10}, S_{24})$	$C_{24} = (D_{100}, S_{24})$

Tabla 5.4: Configuración de parámetros para evaluar la solución MG-CD con una semilla seleccionada

Con respecto a la selección del subconjunto de datos, las configuraciones C_1 a C_8 procesan el mismo subconjunto de datos, que en este caso es un usuario (D_1). De igual forma, las configuraciones C_9 a C_{16} y C_{17} a C_{24} seleccionan los mismos 10 (D_{10}) y 100 (D_{100}) usuarios respectivamente.

Las configuraciones que se muestran en la Tabla 5.4 representan un experimento E_i en el cual el valor de i es la semilla utilizada en la selección aleatoria del subconjunto de datos. Este valor se configura con el fin de reproducir los experimentos.

Para aplicar diferentes cargas (subconjunto de datos) al flujo de trabajo se realizaron dos experimentos E_1 y E_2, cada uno de ellos con las 24 configuraciones que se muestran en la tabla 5.4.

Como resultado se tiene un total de 48 evaluaciones ($\#experimentos$ x $\#configuraciones$) para evaluar la solución MG-CD para el componente de entrega continua en la fase exploratoria.

5.2.4.2. Verificabilidad Continua - Etapa Exploratoria.

Para el componente de verificación continua se determinó evaluar los siguientes parámetros en la etapa exploratoria:

$$CV = \begin{cases} \text{Variedad} \rightarrow \text{Peers por organización: } \{1,2,4,6\} \end{cases}$$

La solución MG-CD/CV es la única de las soluciones estudiadas que considera el componente de verificabilidad continua. Para esta solución el total de configuraciones a evaluar viene dado por los parámetros de *Variedad*, *Volumen*, *Semilla* (definidos en el componente de entrega continua) y un nuevo parámetro dado por la cantidad de Peers por cada organización, el cual es variado de la siguiente forma: P_1, P_2, P_4 y P_6. Por ejemplo, el valor P_2 indica que las tres organizaciones que fueron definidas para este estudio de caso tendrán cada una dos nodos peers que representarán a la organización en la red de verificabilidad.

Considerando estos cuatro parámetros para la verificabilidad continua se tienen un total de 96 configuraciones que se muestran en la Tabla 5.5 con una misma carga de trabajo para todas las configuraciones D_1, D_{10} y D_{100} dada una semilla. En estas configuraciones, D_i representa la cantidad (i) de datos a procesar en el flujo de trabajo, S_j la cantidad (j) de servicios por etapa y P_k la cantidad (k) de nodos peers para cada organización.

Al igual que la solución MG-CD, para MG-CD/CV se realizaron dos experimentos E_1 y E_2, cada uno de ellos con las 96 configuraciones que se muestran en la Tabla 5.5 utilizando las mismas semillas (1,2) seleccionadas para la solución MG-CD con el fin de procesar los mismos datos pero esta vez considerando registros en una red de verificabilidad.

Como resultado se tiene un total de 192 evaluaciones ($\#experimentos$ x $\#configuraciones$) para evaluar la solución MG-CD/CV considerando el componente de entrega continua y verificabilidad continua en la etapa exploratoria de la variación experimental.

Con respecto a la etapa comparativa con la solución DagOn, la variación de los parámetros para MG-CD y MG-CD/CV se ajusta en función de los resultados presentados en la etapa exploratoria. Esta etapa comparativa se describe en la sección 5.2.7.4.

Configuraciones con un nodo peer por organización		
$C_1 = (D_1, S_1, P_1)$	$C_2 = (D_{10}, S_1, P_1)$	$C_3 = (D_{100}, S_1, P_1)$
$C_4 = (D_1, S_2, P_1)$	$C_5 = (D_{10}, S_2, P_1)$	$C_6 = (D_{100}, S_2, P_1)$
$C_7 = (D_1, S_4, P_1)$	$C_8 = (D_{10}, S_4, P_1)$	$C_9 = (D_{100}, S_4, P_1)$
$C_{10} = (D_1, S_6, P_1)$	$C_{11} = (D_{10}, S_6, P_1)$	$C_{12} = (D_{100}, S_6, P_1)$
$C_{13} = (D_1, S_8, P_1)$	$C_{14} = (D_{10}, S_8, P_1)$	$C_{15} = (D_{100}, S_8, P_1)$
$C_{16} = (D_1, S_{10}, P_1)$	$C_{17} = (D_{10}, S_{10}, P_1)$	$C_{18} = (D_{100}, S_{10}, P_1)$
$C_{19} = (D_1, S_{12}, P_1)$	$C_{20} = (D_{10}, S_{12}, P_1)$	$C_{21} = (D_{100}, S_{12}, P_1)$
$C_{22} = (D_1, S_{24}, P_1)$	$C_{23} = (D_{10}, S_{24}, P_1)$	$C_{24} = (D_{100}, S_{24}, P_1)$
Configuraciones con dos peers por organización		
$C_{25} = (D_1, S_1, P_2)$	$C_{26} = (D_{10}, S_1, P_2)$	$C_{27} = (D_{100}, S_1, P_2)$
$C_{28} = (D_1, S_2, P_2)$	$C_{29} = (D_{10}, S_2, P_2)$	$C_{30} = (D_{100}, S_2, P_2)$
$C_{31} = (D_1, S_4, P_2)$	$C_{32} = (D_{10}, S_4, P_2)$	$C_{33} = (D_{100}, S_4, P_2)$
$C_{34} = (D_1, S_6, P_2)$	$C_{35} = (D_{10}, S_6, P_2)$	$C_{36} = (D_{100}, S_6, P_2)$
$C_{37} = (D_1, S_8, P_2)$	$C_{38} = (D_{10}, S_8, P_2)$	$C_{39} = (D_{100}, S_8, P_2)$
$C_{40} = (D_1, S_{10}, P_2)$	$C_{41} = (D_{10}, S_{10}, P_2)$	$C_{42} = (D_{100}, S_{10}, P_2)$
$C_{43} = (D_1, S_{12}, P_2)$	$C_{44} = (D_{10}, S_{12}, P_2)$	$C_{45} = (D_{100}, S_{12}, P_2)$
$C_{46} = (D_1, S_{24}, P_2)$	$C_{47} = (D_{10}, S_{24}, P_2)$	$C_{48} = (D_{100}, S_{24}, P_2)$
Configuraciones con cuatro peers por organización		
$C_{49} = (D_1, S_1, P_4)$	$C_{50} = (D_{10}, S_1, P_4)$	$C_{51} = (D_{100}, S_1, P_4)$
$C_{52} = (D_1, S_2, P_4)$	$C_{53} = (D_{10}, S_2, P_4)$	$C_{54} = (D_{100}, S_2, P_4)$
$C_{55} = (D_1, S_4, P_4)$	$C_{56} = (D_{10}, S_4, P_4)$	$C_{57} = (D_{100}, S_4, P_4)$
$C_{58} = (D_1, S_6, P_4)$	$C_{59} = (D_{10}, S_6, P_4)$	$C_{60} = (D_{100}, S_6, P_4)$
$C_{61} = (D_1, S_8, P_4)$	$C_{62} = (D_{10}, S_8, P_4)$	$C_{63} = (D_{100}, S_8, P_4)$
$C_{64} = (D_1, S_{10}, P_4)$	$C_{65} = (D_{10}, S_{10}, P_4)$	$C_{66} = (D_{100}, S_{10}, P_4)$
$C_{67} = (D_1, S_{12}, P_4)$	$C_{68} = (D_{10}, S_{12}, P_4)$	$C_{69} = (D_{100}, S_{12}, P_4)$
$C_{70} = (D_1, S_{24}, P_4)$	$C_{71} = (D_{10}, S_{24}, P_4)$	$C_{72} = (D_{100}, S_{24}, P_4)$
Configuraciones con seis peers por organización		
$C_{73} = (D_1, S_1, P_6)$	$C_{74} = (D_{10}, S_1, P_6)$	$C_{75} = (D_{100}, S_1, P_6)$
$C_{76} = (D_1, S_2, P_6)$	$C_{77} = (D_{10}, S_2, P_6)$	$C_{78} = (D_{100}, S_2, P_6)$
$C_{79} = (D_1, S_4, P_6)$	$C_{80} = (D_{10}, S_4, P_6)$	$C_{81} = (D_{100}, S_4, P_6)$
$C_{82} = (D_1, S_6, P_6)$	$C_{83} = (D_{10}, S_6, P_6)$	$C_{84} = (D_{100}, S_6, P_6)$
$C_{85} = (D_1, S_8, P_6)$	$C_{86} = (D_{10}, S_8, P_6)$	$C_{87} = (D_{100}, S_8, P_6)$
$C_{88} = (D_1, S_{10}, P_6)$	$C_{89} = (D_{10}, S_{10}, P_6)$	$C_{90} = (D_{100}, S_{10}, P_6)$
$C_{91} = (D_1, S_{12}, P_6)$	$C_{92} = (D_{10}, S_{12}, P_6)$	$C_{93} = (D_{100}, S_{12}, P_6)$
$C_{94} = (D_1, S_{24}, P_6)$	$C_{95} = (D_{10}, S_{24}, P_6)$	$C_{96} = (D_{100}, S_{24}, P_6)$

Tabla 5.5: Configuración de parámetros para evaluar la solución MG-CD/CV con una semilla

5.2.5 Métricas

Se definieron las siguientes métricas que fueron capturadas en las fases de ejecución (preparación, despliegue y operación) del flujo de trabajo por las tres soluciones estudiadas:

1. RT: Tiempo de respuesta (Response Time): Esta métrica representa el tiempo (en segundos) que transcurre desde que el usuario realiza una petición hasta que obtiene alguna respuesta. Para este trabajo de investigación, el tiempo de respuesta equivale desde que el usuario aplica una carga al flujo de trabajo hasta que obtiene la visualización de los resultados.

2. DT: Tiempo de despliegue (Deployment Time): Representa el tiempo requerido para desplegar y tener en servicio todas las unidades de procesamiento o contenedores virtuales. Este tiempo también incluye Tiempo de ejecución (Ru o Runtime), el cual representa el intervalo de tiempo (seg) en el que un script de despliegue se ejecuta en el sistema operativo.

3. Ov_d: Sobrecarga (Overhead): Esta métrica representa el tiempo (seg) extra que una determinada operación o proceso puede requerir en caso de añadir una funcionalidad. En este caso, se busca medir la sobrecarga producida al agregar la funcionalidad de verificación continua a un flujo de trabajo tradicional como DagOn o sobre nuestra propuesta MG-CD.

4. Th: Rendimiento (Throughput): Corresponde al volumen de datos (Mb/seg) que son procesados a través del flujo de trabajo.

Las métricas anteriormente descritas fueron evaluadas de la siguiente forma.

5.2.5.1. Fase de preparación y despliegue

Para el componente de entrega continua se aplicó la métrica de Tiempo de ejecución (Ru) de los siguientes procesos de la fase de preparación, tanto para MG-CD como MG-CD/CV:

1. **Tiempos de ejecución (Ru)**:
   - Generación de archivos de configuración de construcción del flujo de trabajo.
   - Cálculo de la firma digital del archivo que contiene el modelo ETC.
   - Cargar configuración del modelo ETC al Marco de Gestión.

Con respecto al componente de verificabilidad continua se calculó la métrica Ru de los siguientes procesos para la solución MG-CD/CV:

a) Construcción de la red de verificabilidad (Red de Hyperledger Fabric - HF):

- Generación archivos de configuración.
- Generación de material criptográfico (llaves privadas, certificados, etc).
- Generación de artefactos del canal.

b) Construcción de la red de negocios (red de Hyperledger Composer - HC):

- Construir perfiles de conexión a la red HF
- Crear tarjetas de conexión a HF para los administradores de la red
- Importar tarjetas de conexión al Wallet de Hyperledger Composer
- Instalar el archivo de red de negocio (BNA) en HF
- Obtener identidad de los administradores
- Iniciar red BNA en HF.
- Crear tarjetas de acceso a HF para los participantes.
- Importar tarjetas de acceso
- Ping a la Red BNA en HF
- Crear API REST

En la fase de despliegue se calculó la métrica Tiempo de Despliegue (DT) para la solución de la literatura DagOn que sólo considera entrega continua y para nuestra solución propuesta MG-CD/CV con el fin de realizar una comparativa y medir el sobrecosto (Ov_d) de la verificación continua en el despliegue.

1. **Tiempos de despliegue (DT) y Sobrecarga (Ov_d)**:

a) DT_{DagOn}: Tiempo de despliegue de DagOn.

$$DT_{DagOn} = \sum_{i=1}^{n} ST_{D_{S_i}} \tag{5.1}$$

donde:

$ST_{D_{S_i}}$ es el tiempo de servicio del despliegue del contenedor virtual S_i

b) $DT_{CD/CV}$: Tiempo de despliegue del Gestor CD/CV.

$$DT_{CD/CV} = \sum_{i=1}^{n} ST_{D_{CD_i}} + \sum_{i=1}^{n} ST_{D_{CV_i}} \tag{5.2}$$

donde:

$ST_{D_{CD_i}}$ es el tiempo de servicio del despliegue (ST_D) del contenedor virtual de entrega continua CD_i y $ST_{D_{CV_i}}$ es el tiempo de servicio del despliegue del contenedor virtual de verificación continua CV_i.

c) Ov_d: Sobrecarga en la etapa de despliegue.

$$Ov_d = (DT_{CD/CV} - DT_{DagOn})/DT_{DagOn} \tag{5.3}$$

Adicionalmente para el componente de verificabilidad continua específicamente para la solución MG-CD/CV se calculó la métrica Ru de los siguientes procesos en la fase de despliegue:

2. **Tiempos de ejecución (Ru)**:

   - Registro de los participantes.
   - Creación del canal.
   - Unir todos los nodos peers al canal.
   - Fijar todos los anchors peers de la organización.

5.2.5.2. Fase de Operación:

En la fase de operación se calcularon las métricas de Tiempo de Respuesta (RT) y el Rendimiento (Th) para todas las soluciones propuestas descritas en la tabla 5.3.

1. **Tiempos de respuesta (RT)**:

   a) RT_{DagOn}: Tiempo de respuesta de DagOn.

$$RT_{DagOn} = \sum_{i=1}^{n} ST_{S_i} \tag{5.4}$$

   donde:

   ST_{S_i} es el tiempo de servicio del contenedor virtual o servicio S_i

   b) $RT_{CD/CV}$: Tiempo de respuesta del Gestor CD/CV.

$$RT_{CD/CV} = \sum_{i=1}^{n} ST_{S_i} + ST_{BC_{ETC}} \tag{5.5}$$

   donde:

   $ST_{BC_{ETC}}$: Tiempo de servicio de las transacciones generadas con el modelo ETC en los bloques encadenados.

$$ST_{BC_{ETC}} = T_{HashETC} + ST_{BC} \tag{5.6}$$

   ST_{BC}: Tiempo de servicio de registrar todas las transacciones (Tx) de cada servicio (S_i) en los bloques encadenados.

$$ST_{BC} = \sum_{i=1}^{n} ST_{Tx_{S_i}} \tag{5.7}$$

   c) Ov_o: Sobrecarga en la etapa de operación (MG-CD/CV vs MG-CD).

$$Ov_o = (RT_{CD/CV} - RT_{CD})/RT_{CD} \tag{5.8}$$

donde:

RT_{CD} es el Tiempo de respuesta del Marco de Gestión MG-CD

$$RT_{CD} = \sum_{i=1}^{n} ST_{S_i} \tag{5.9}$$

d) Ov_o: Sobrecarga en la etapa de operación (MG-CD/CV vs DagOn)

$$Ov_o = (RT_{CD/CV} - RT_{DagOn})/RT_{DagOn} \tag{5.10}$$

2. **Rendimiento (Th)**:

a) $Th_{CD/CV}$: Rendimiento del Gestor CD/CV.

$$Th_{CD/CV} = DataSize/RT_{CD/CV} \tag{5.11}$$

donde:

$DataSize$ es el tamaño en Megabytes (MB) de la carga de datos o subconjunto de datos seleccionados para procesar en el flujo de trabajo.

b) Th_{CD}: Rendimiento del Gestor CD.

$$Th_{CD} = DataSize/RT_{CD} \tag{5.12}$$

c) Th_{DagOn}: Rendimiento de DagOn.

$$Th_{DagOn} = DataSize/RT_{DagOn} \tag{5.13}$$

5.2.6 Resultados y discusión de la etapa exploratoria

En esta sección se presentan los resultados obtenidos en la etapa exploratoria de las soluciones propuestas en este trabajo de investigación con las configuraciones definidas en la variación experimental.

El primer análisis realizado en la etapa exploratoria consistió en determinar la cantidad de contenedores virtuales CD/CV que se despliegan para el escenario de movilidad en función de los parámetros *cantidad de Servicios por etapa* y *Nodos peers por organización*.

En la Figura 5.6 se muestra la cantidad de los diferentes tipos de contenedores desplegados por cada notación $S_i - P_j$, donde i es la cantidad de servicios por etapa y j la cantidad de nodos peers por organización de una determinada configuración definida en la etapa exploratoria.

Para este estudio de caso de movilidad los contenedores CD que corresponden al flujo de trabajo que realiza entrega continua son: *Base de Datos*, *Exhibición*, *Etapa Preprocesamiento* y *Etapa Procesamiento*. Los contenedores CV que conforman la red de verificabilidad que realiza la verificación continua son: *Nodos Ca*, *Nodo Orderer*, *cliente*, *Nodos Peers* y *dev-peer*. El contenedor *Gestor CD/CV* es la solución propuesta en este trabajo de investigación encapsulada.

A modo de ejemplo, en la Fig. 5.6 se muestra que la notación que más contenedores virtuales despliega es la S_8-P_6 con un total de 60 contenedores. De los cuales 8 corresponden a la etapa de preprocesamiento y 8 de la etapa procesamiento dada la notación S_8. Para efectos prácticos del desarrollo de este trabajo de investigación, cualquier configuración desplegada tiene 1 contenedor virtual para la Base de Datos y 1 para el módulo de exhibición donde se visualizan los resultados. Estos 18 contenedores mencionados previamente conforman el componente de entrega continua CD. La notación P_6 en la configuración tomada como ejemplo (S_8-P_6) indica que cada organización (tres para este estudio de caso) tiene 6 nodos peers en la red de verificabilidad, es decir, dado que son 3 organizaciones se tendrá un total de 18 contenedores virtuales que representan los 18 Nodos peers de las tres organizaciones. Como se estableció previamente, la red de verificabilidad desplegada

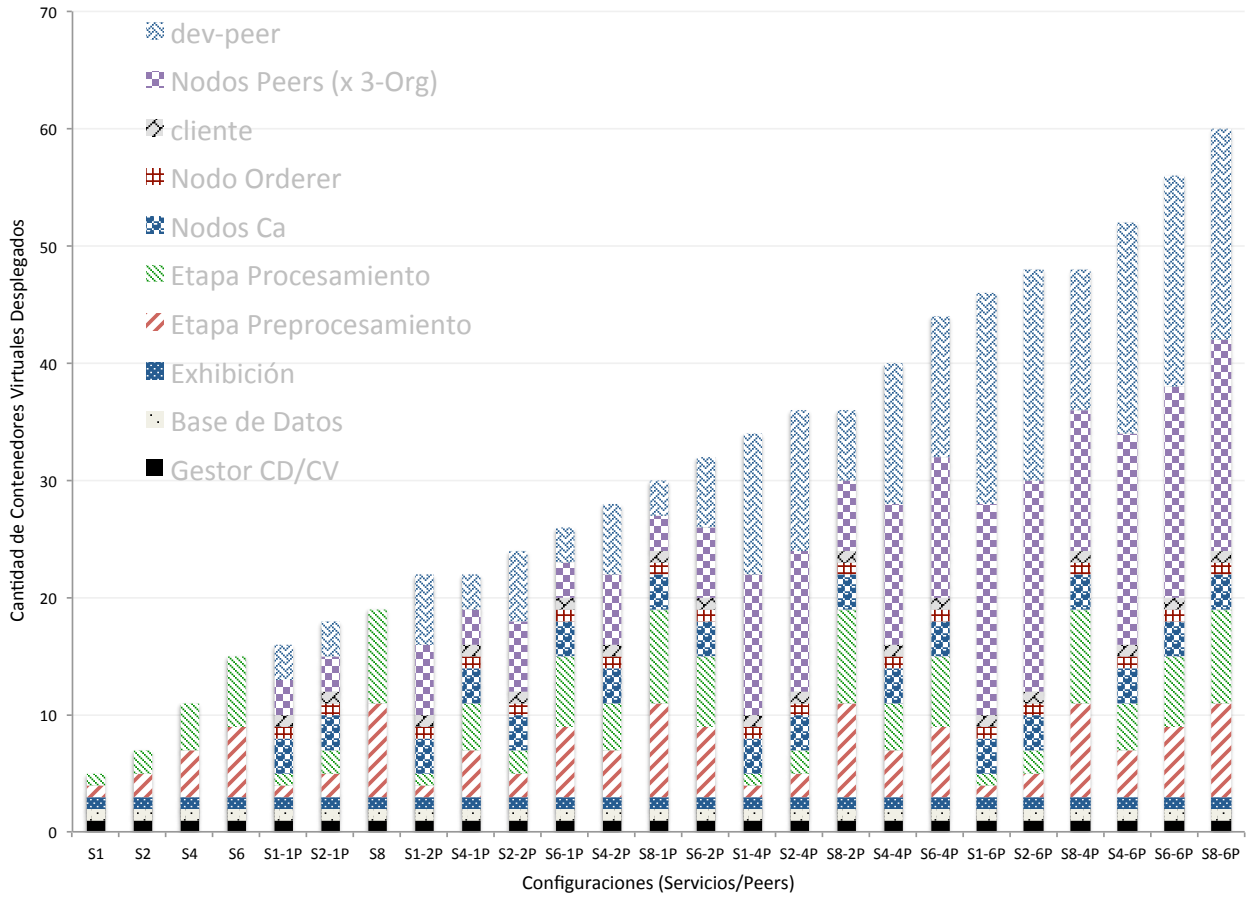
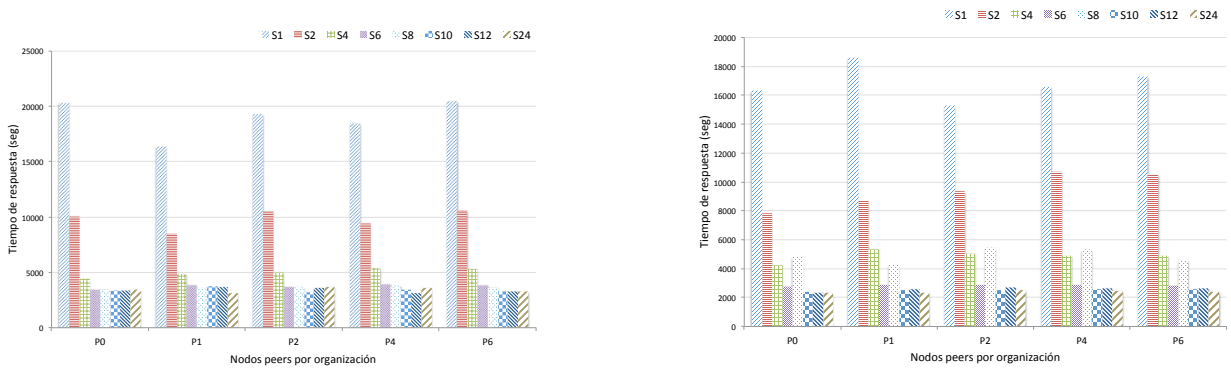


Figura 5.6: Cantidad de Contenedores Virtuales CD/CV desplegados por cada configuración para el escenario de movilidad - Etapa exploratoria.

tendrá 1 solo Nodo orderer y un nodo de autoridad certificadora (CA) para cada organización. Un contenedor virtual llamado *cliente* es desplegado para realizar operaciones tales como la creación del canal a través del cual se comunican los nodos peers, la unión de los nodos peer al canal, entre otros. Adicionalmente, por cada Nodo Peer desplegado se despliega otro contenedor virtual (dev-peer) para configurar la red de negocios en el Nodo Peer. En este sentido, para la red de verificabilidad se tendrían 41 contenedores virtuales CV. Por último, el contenedor *Gestor CD/CV* es la solución propuesta de este trabajo de investigación encapsulada que, junto a los contenedores virtuales CD/CV mencionados recientemente, conforman el total de 60 contenedores desplegados para la configuración S_8-P_6 tomada como ejemplo.

El despliegue de cada contenedor virtual ya sea CD o CV implica un mayor uso de memoria, espacio en disco y demás recursos que se deben compartir entre todos los contenedores virtuales desplegados en la infraestructura que se definió previamente. En este sentido, en la Figura 5.6 se puede observar que aquellas configuraciones con mayor número de peers (P_4 y P_6) implican un mayor uso de los recursos.

Para observar el rendimiento de las configuraciones previas en la etapa exploratoria se muestran a continuación los tiempos de respuesta obtenidos en las soluciones propuestas donde la notación P_0 equivale a la solución *MG-CD* dado que no considera nodos peers por organización y las configuraciones con los parámetros P_1, P_2, P_4 y P_6 pertenecientes a solución *MG-CD/CV* variando la cantidad de nodos peer por cada organización.

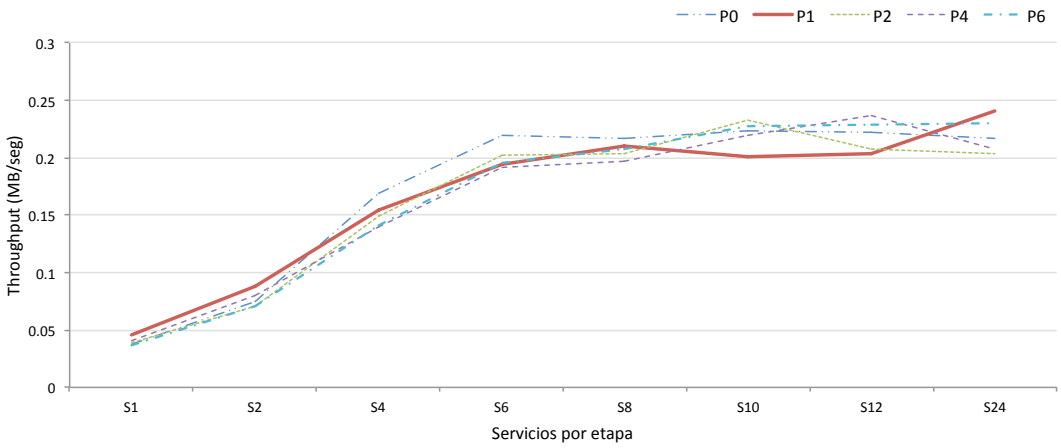


(a) (b)

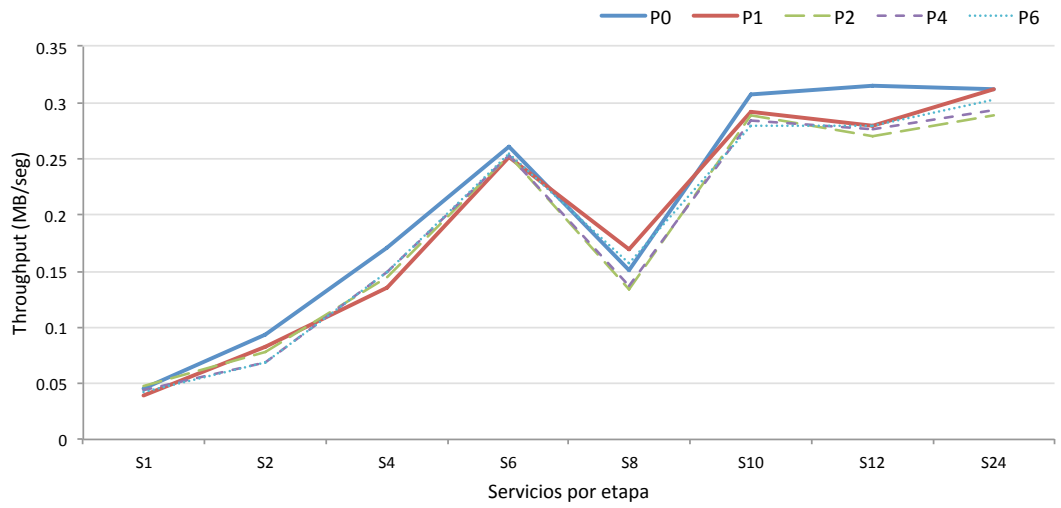
Figura 5.7: Tiempos de respuesta de las configuraciones definidas en la etapa exploratoria. (a) Carga de 100 datos seleccionados con semilla 1 (b) Carga de datos 100 datos seleccionados con semilla 2.

Se puede observar en las Figuras 5.7(a) y 5.7(b) que, independientemente de la carga de trabajo o el número de nodos peers por organización, las configuraciones con los parámetros S_{10}, S_{12} y S_{24} son las que obtienen los tiempos de respuesta más reducidos en comparación con las otras configuraciones. Adicionalmente, se puede observar que no existe una gran diferencia cuando el número de nodos peers por organización es variado (P_1, P_2, P_4 y P_6). Esto se debe a que todos los nodos peers de la red de verificabilidad se encuentran desplegados en la misma infraestructura independientemente de la organización a la que pertenezcan.

El rendimiento de cada configuración de las soluciones propuestas *MG-CD* y *MG-CD/CV* para las cargas de 100 datos seleccionados con semilla 1 y 2 se muestran en la Fig. 5.8(a) y 5.8(b) respectivamente.



(a)



(b)

Figura 5.8: Rendimiento (Throughput) de las configuraciones definidas en la etapa exploratoria. (a) Carga de 100 datos seleccionados con semilla 1 (b) Carga de datos 100 datos seleccionados con semilla 2.

Como se puede observar en las figuras 5.8(a) y 5.8(b), es posible obtener un mayor rendimiento cuando se aplican esquemas de paralelismo (configuraciones S_2 a S_{24}) que permitan procesar mayor

cantidad de trabajo en menor unidad de tiempo. Por otro lado, cuando se considera el componente de verificabilidad continua y se considera diferentes números de nodos por organización, el valor seleccionado para dicho parámetro no afecta en gran medida el rendimiento de la solución propuesta dado que todos los nodos peers para esta investigación se encuentran geográficamente en el mismo lugar. Sin embargo, la utilización de recursos en el despliegue de la solución y durante la operación del sistema, aumenta notablemente cuando se considera una mayor cantidad de nodos peers por organización tal como se vio en la Fig. 5.6.

Cabe aclarar que los resultados presentados hasta este momento fueron evaluados en una sola iteración con el objetivo de conocer una aproximación de los parámetros con los cuales las soluciones propuestas obtienen un mejor comportamiento. Sin importar la carga, los resultados en esta etapa exploratoria indican que es recomendable tener la cantidad de servicios por etapa en función de la cantidad de cores disponibles en la infraestructura que se despliega la solución. Del mismo modo, utilizar 4 y 6 nodos peers para cada organización implicaba una sobre utilización de los recursos (memoria, disco, etc) solo para el despliegue de la solución del gestor CD/CV, lo cual afectaba directamente el tiempo de respuesta que percibía el usuario en tiempo de operación al tener menos recursos disponibles para realizar el procesamiento. Adicionalmente, una configuración de un nodo peer por organización podría no ser adecuada, dado que en un eventual fallo del nodo, la organización quedaría sin posibilidad de acceder a los registros de la red de verificabilidad.

En este contexto, una segunda etapa a la cual denominamos *etapa comparativa*, fue llevada a cabo para realizar una comparativa directa entre una solución del estado del arte y las soluciones propuestas en este trabajo de investigación con los parámetros del Gestor que permiten mayores prestaciones de rendimiento.

5.2.7 Resultados y discusión de la etapa comparativa

Dados los resultados obtenidos en la etapa exploratoria con las soluciones propuestas en este trabajo de investigación, para el componente de entrega continua se determinó aplicar los mejores tres parámetros de servicios por etapa $S = \{10, 12, 24\}$ con tres cargas de trabajo diferentes $D = \{1, 10, 100\}$ seleccionadas con un valor de semilla igual a 1 para realizar la evaluación experimental. Adicionalmente, se consideró el parámetro $S = 1$ para determinar el comportamiento de las soluciones estudiadas sin aplicar paralelismo, es decir, realizando el procesamiento en el flujo de trabajo de manera secuencial.

Cada una de las configuraciones formadas a partir de la combinación de los parámetros anteriores, fueron ejecutadas en 15 iteraciones cada una con el objetivo de tener una mejor aproximación de los resultados respecto a la fase exploratoria. A continuación se describen los resultados obtenidos por cada solución estudiada y posteriormente se presenta un análisis comparativo de las tres soluciones.

5.2.7.1. Resultados solución MG-CD.

El total de configuraciones evaluadas para la solución MG-CD se muestran en la Tabla 5.6.

Carga de 1 Usuario	Carga de 10 Usuarios	Carga de 100 Usuarios
$C_1 = (D_1, S_1)$	$C_5 = (D_{10}, S_1)$	$C_9 = (D_{100}, S_1)$
$C_2 = (D_1, S_{10})$	$C_6 = (D_{10}, S_{10})$	$C_{10} = (D_{100}, S_{10})$
$C_3 = (D_1, S_{12})$	$C_7 = (D_{10}, S_{12})$	$C_{11} = (D_{100}, S_{12})$
$C_4 = (D_1, S_{24})$	$C_8 = (D_{10}, S_{24})$	$C_{12} = (D_{100}, S_{24})$

Tabla 5.6: Configuración de parámetros para evaluar la solución MG-CD en la etapa de comparación

Las configuraciones C_1, C_5 y C_9 cuyo parámetro en común es S_1 (procesamiento lineal) permiten ver los tiempos de servicio de una solución secuencial en función de la carga de trabajo (ver Fig. 5.9).

Se puede ver en la Fig. 5.9 que existe una correlación directamente proporcional entre la carga de trabajo y el tiempo de respuesta en una solución secuencial. Dado que se tiene un crecimiento de un orden de magnitud entre la configuración con D_1 y D_{10}, así como también entre D_{10} y D_{100}

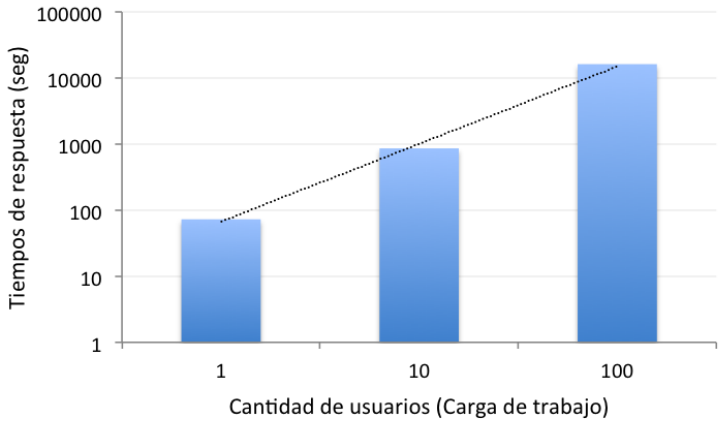
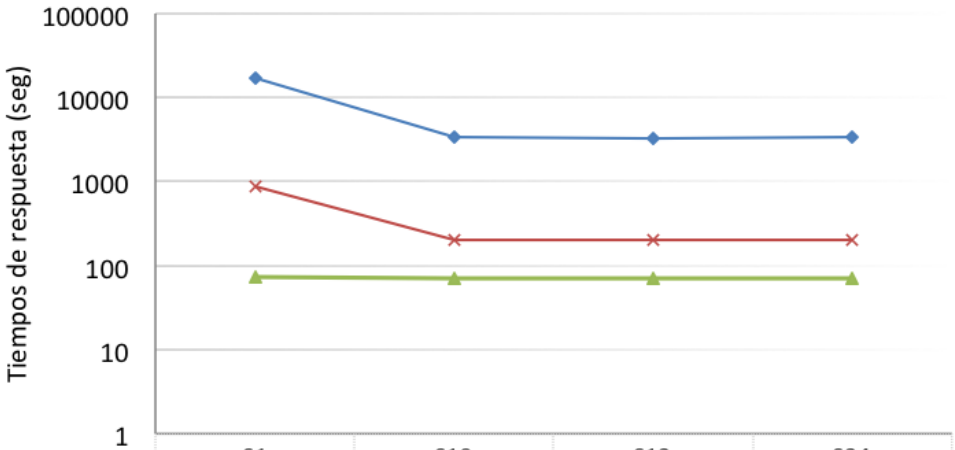


Figura 5.9: Aumento de la carga de trabajo en el procesamiento secuencial.

se hace necesario el uso de paralelismo (configuraciones que incluyen S_{10}, S_{12}, S_{24}) para procesar cargas de trabajo mayores a las mencionadas.

Los resultados promedio de las 15 iteraciones para el total de configuraciones de MG-CD que incluye el procesamiento secuencial y paralelo se muestran a continuación:



	S1	S10	S12	S24
100	16903.8468	3357.32585	3229.942755	3341.727177
10	862.437708	203.225618	201.0027328	203.0039779
1	72.7779436	70.3220512	70.49304833	71.6804314

Figura 5.10: Tiempos de respuesta promedio de la solución MG-CD con las 12 configuraciones de la etapa de comparación.

Los resultados que se muestran en la Fig. 5.10 indican que la mejor configuración de servicios por etapa para el flujo de trabajo del escenario de movilidad es de S_{12}. Este resultado indica que se obtienen mejores tiempos de respuesta al utilizar todos los cores disponibles de la infraestructura (12 cores) en la cual se despliega la solución. En este caso, cuando se despliegan más hilos o threads como el caso de las configuraciones con el parámetro S_{24}, se empiezan a reutilizar los cores disponibles de la infraestructura y no se obtiene una mejora considerable en los tiempos de respuesta.

5.2.7.2. Resultados solución MG-CD/CV.

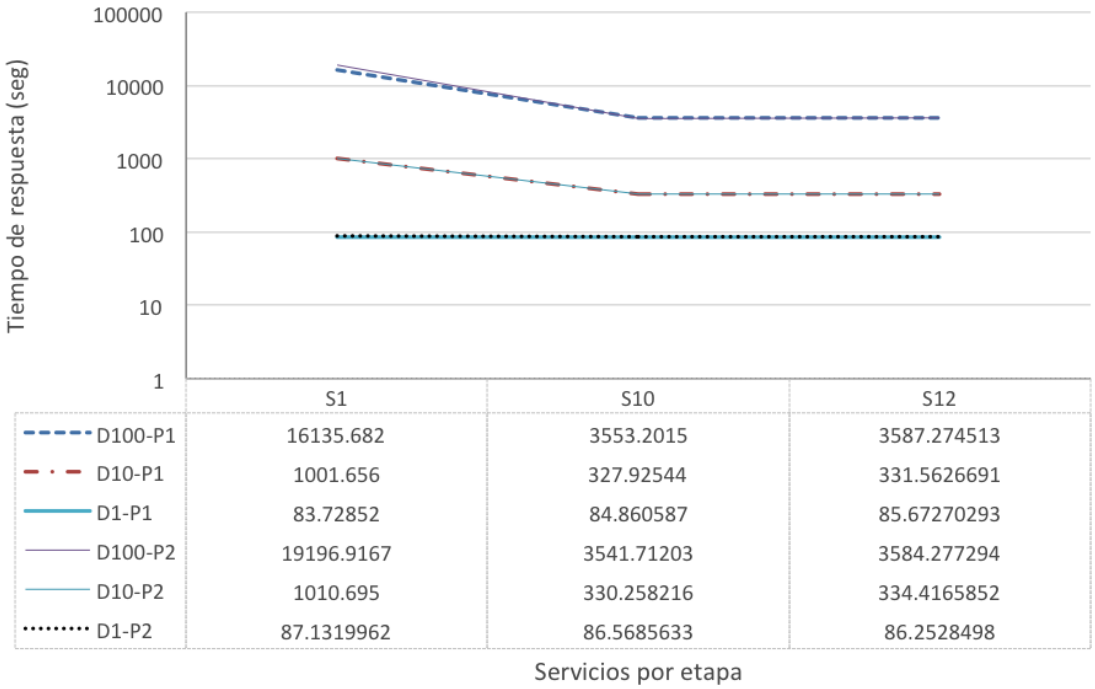
Con base a los resultados anteriores se evaluó la solución MG-CD/CV con las mismas cargas de trabajo y cantidad de servicios que fue evaluada la solución MG-CD pero excluyendo el parámetro S_{24} dado que no representa una mejora respecto al uso de los recursos disponibles y por ende al procesamiento y rendimiento del flujo de trabajo. Las configuraciones que fueron evaluadas para la solución MG-CD/CV se muestran en la tabla 5.7.

Configuraciones con un nodo peer por organización		
$C_1 = (D_1, S_1, P_1)$	$C_2 = (D_{10}, S_1, P_1)$	$C_3 = (D_{100}, S_1, P_1)$
$C_4 = (D_1, S_{10}, P_1)$	$C_5 = (D_{10}, S_{10}, P_1)$	$C_6 = (D_{100}, S_{10}, P_1)$
$C_7 = (D_1, S_{12}, P_1)$	$C_8 = (D_{10}, S_{12}, P_1)$	$C_9 = (D_{100}, S_{12}, P_1)$
Configuraciones con un nodo peer por organización		
$C_{10} = (D_1, S_1, P_2)$	$C_{11} = (D_{10}, S_1, P_2)$	$C_{12} = (D_{100}, S_1, P_2)$
$C_{13} = (D_1, S_{10}, P_2)$	$C_{14} = (D_{10}, S_{10}, P_2)$	$C_{15} = (D_{100}, S_{10}, P_2)$
$C_{16} = (D_1, S_{12}, P_2)$	$C_{17} = (D_{10}, S_{12}, P_2)$	$C_{18} = (D_{100}, S_{12}, P_2)$

Tabla 5.7: Configuración de parámetros para evaluar la solución MG-CD/CV en la etapa comparativa

Los resultados promedio de las 15 iteraciones para las 18 configuraciones de MG-CD/CV se muestran a continuación:

Los resultados que se muestran en la Fig. 5.11 indican que sin importar la cantidad de peers que se seleccione (P_1 o P_2) los tiempos de respuesta entre las configuraciones que aplican la misma carga de trabajo y tienen la misma cantidad de servicios S_i son muy semejantes. Por ejemplo, para las configuraciones con S_{10} con una carga de 100 datos (D_{100}), la diferencia en los tiempos de respuesta



	S1	S10	S12
D100-P1	16135.682	3553.2015	3587.274513
D10-P1	1001.656	327.92544	331.5626691
D1-P1	83.72852	84.860587	85.67270293
D100-P2	19196.9167	3541.71203	3584.277294
D10-P2	1010.695	330.258216	334.4165852
D1-P2	87.1319962	86.5685633	86.2528498

Figura 5.11: Tiempos de respuesta promedio de la solución MG-CD/CV con las 18 configuraciones en la etapa de comparación.

promedio, de utilizar 1 Nodo peer o 2 Nodos peers por organización en la red de verificabilidad es tan solo de $0.32\,\%$. Adicionalmente, se observa que a partir de 10 servicios los tiempos de respuesta tienden a acotarse y no se obtiene una mejora sustancial al incrementar dicho parámetro respecto a las configuraciones menores a S_{10}.

5.2.7.3. Resultados solución DagOn.

Respecto a la solución de DagOn, este motor de flujos de trabajos también realiza paralelismo de tareas utilizando hilos (o threads). En este sentido se evaluó esta solución variando la cantidad de hilos que podía lanzar ($T = 1, 10, 12, 100, *$) con el fin de poder comparar dicha solución con nuestras soluciones propuestas. El parámetro $T = *$, significa que DagOn utiliza todos los recursos disponibles de la infraestructura (ver Tabla 5.1) en la cual se despliega la solución. Para ello, DagOn incorpora un componente conocido como SLURM [76] que es un administrador de *clusters* capaz de

escalar a miles de procesadores que le permite a DagOn gestionar todos los recursos disponibles de la infraestructura donde es desplegado.

El total de configuraciones evaluadas para la solución DagOn se muestran en la tabla 5.8.

Carga de 1 Usuario	Carga de 10 Usuarios	Carga de 100 Usuarios
$C_1 = (D_1, T_1)$	$C_6 = (D_{10}, T_1)$	$C_{11} = (D_{100}, T_1)$
$C_2 = (D_1, T_{10})$	$C_7 = (D_{10}, T_{10})$	$C_{12} = (D_{100}, T_{10})$
$C_3 = (D_1, T_{12})$	$C_8 = (D_{10}, T_{12})$	$C_{13} = (D_{100}, T_{12})$
$C_4 = (D_1, T_{100})$	$C_9 = (D_{10}, T_{100})$	$C_{14} = (D_{100}, T_{100})$
$C_5 = (D_1, T_*)$	$C_{10} = (D_{10}, T_*)$	$C_{15} = (D_{100}, T_*)$

Tabla 5.8: Configuración de parámetros para evaluar la solución DagOn en la etapa de comparación

Los resultados promedio de las 15 iteraciones para las 15 configuraciones de DagOn se muestran en la Fig. 5.12, donde se puede observar los tiempos de respuesta obtenidos considerando una versión secuencial de DagOn y cómo mejora en función de la cantidad de hilos que se le habilitan para el procesamiento. La configuración de DagOn que permite utilizar todos los recursos disponibles ($DagOn*$) con el componente SLURM fue la que mejor resultados obtuvo reduciendo en un $96.43\,\%$ los tiempos de respuesta respecto a la versión secuencial de DagOn para una carga de 10 datos y un $94.09\,\%$ para una carga de 1 dato.

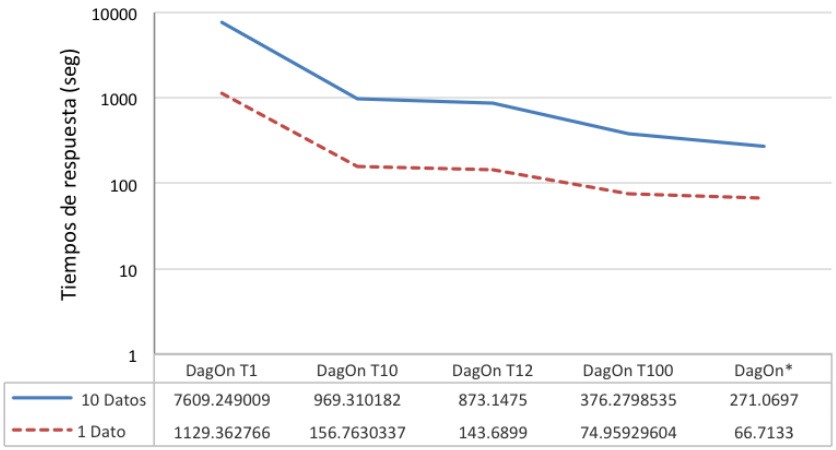


Figura 5.12: Tiempos de respuesta promedio de la solución DagOn con las 10 configuraciones ejecutadas correctamente. (Configuraciones con D_{100} sin éxito.)

Para la solución de DagOn se puede observar en la Fig. 5.12 que sólo se muestran datos para una y diez cargas de trabajo (D_1 y D_{10}). Esto se debe a que las pruebas cuyas configuraciones eran con una carga de 100 datos (C_{11} a C_{15} de la tabla 5.8) no pudieron ser evaluadas correctamente debido a un problema de memoria que presentaba la solución. Al tener una carga de trabajo considerable (del orden de Gbytes), en este caso de 100 datos, el mecanismo que utiliza DagOn para generar la dependencia entre las tareas generaba por cada dato a procesar cerca de 10 subtareas que debían ser ejecutas para garantizar una entrega continua en el flujo de trabajo. Esto generaba más de 10000 tareas que debían ser ejecutadas en el flujo de trabajo y que eran cargadas en memoria durante la ejecución de la solución. Teniendo en cuenta la infraestructura en la cual se desplegó DagOn, la cantidad de memoria disponible en este trabajo de investigación fue insuficiente para ejecutar dichas tareas que permitieran procesar una carga de 100 datos. Es por ello que para la solución de DagOn sólo fue posible realizar la experimentación con 1 y 10 cargas de trabajo para realizar una comparativa con nuestras soluciones propuestas.

5.2.7.4. Análisis comparativo y discusión entre las soluciones estudiadas.

En esta sección se presenta un análisis comparativo entre las soluciones estudiadas con los resultados obtenidos y descritos anteriormente. El primer análisis que se presenta consiste en describir el sobrecosto (Overhead) que se genera al agregar la funcionalidad de verificabilidad en la solución *MG-CD/CV* respecto a las soluciones estudiadas de flujos de trabajo tradicionales como *MG-CD* y *DagOn*. Este análisis se llevó a cabo en primera instancia comparando las versiones secuenciales (parámetro S_1 en la configuración) de las soluciones estudiadas.

La Fig. 5.13 muestra el sobrecosto de la solución MG-CD/CV y DagOn en su versión secuencial, respecto a la solución secuencial de MG-CD. En primer lugar se puede ver que DagOn cuando opera de manera secuencial tiene un gran sobrecosto (1451.79 % para 1 dato y 782.29 % para 10 datos) respecto a la solución *MG-CD* e incluso con la solución MG-CD/CV que considera el componente de verificabilidad. Este porcentaje tan alto de la solución del estado del arte se debe a que esta solución

está pensada para el procesamiento paralelo de tareas y no para operar de manera secuencial.

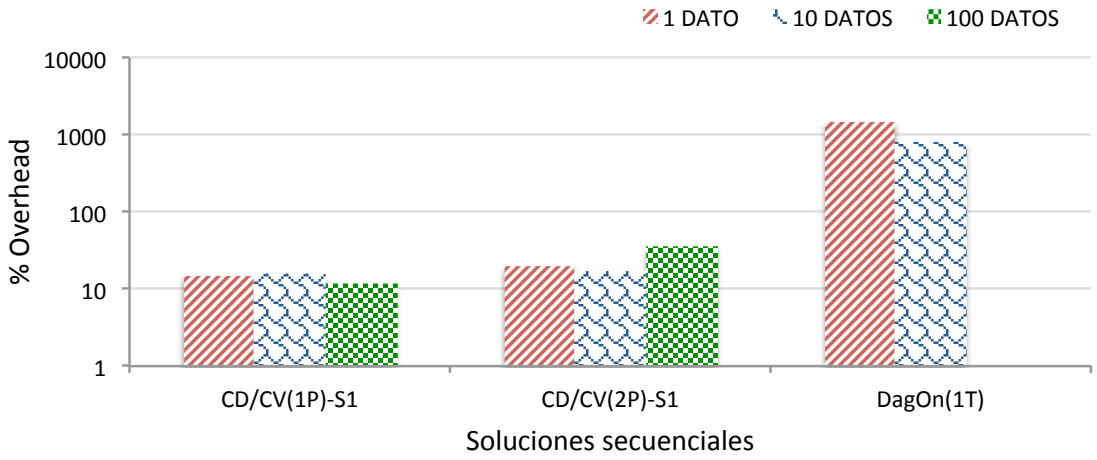


Figura 5.13: Comparativa del sobrecosto de las soluciones MG-CD/CV y DagOn respecto a MG-CD sin considerar paralelismo de tareas.

Con respecto a las configuraciones de MG-CD/CV, sin importar la carga de trabajo, generan un sobrecosto entre el 12 % y 35 % respecto a MG-CD. Adicionalmente, cuando se considera una configuración con mayor número de nodos peer y la carga de trabajo aumenta, se tendrá un mayor sobrecosto. Esto se debe a que por tener un número mayor de nodos peers en la red de verificabilidad, mayor es el tiempo en aceptar una transacción en el protocolo de consenso, lo cual impacta directamente en los tiempos de respuesta que percibe el usuario en el flujo de trabajo.

Para enfrentar estos sobrecostos del componente de verificación en el flujo de trabajo tradicional de MG-CD se aplicaron las soluciones paralelizadas de MG-CD/CV y DagOn con 10 y 12 hilos de ejecución cada uno (notación S_{10} y S_{12} en las configuraciones de cada solución).

En la Fig. 5.14 se puede ver que para las configuraciones de MG-CD/CV con 10 y 12 hilos sin importar la cantidad de nodos peers, ya no existe un sobrecosto respecto al flujo de trabajo tradicional de MG-CD secuencial para cargas de trabajo de 10 y 100 usuarios. Para 1 dato se mantiene el sobrecosto dado que no importa la cantidad de hilos de procesamiento que se desplieguen, sólo un hilo será ocupado para el procesamiento de dicho dato. El porcentaje de ganancia de las diferentes

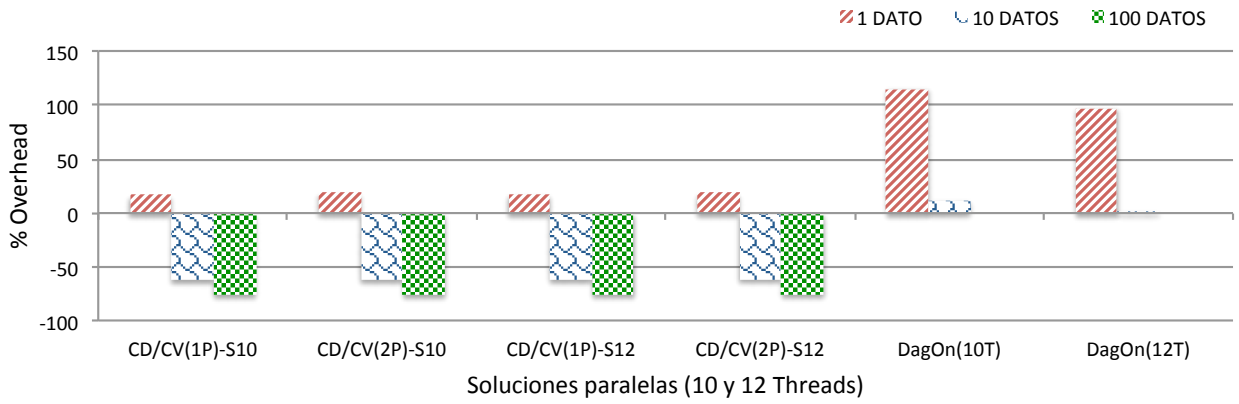


Figura 5.14: Comparativa del sobrecosto entre las soluciones estudiadas considerando paralelismo de tareas.

configuraciones de MG-CD/CV con 10 y 12 hilos respecto a MG-CD fue aproximadamente del 61 % para 10 datos y del 75 % para 100 datos.

Con respecto a DagOn, cuando aplica 10 y 12 hilos sigue teniendo un sobrecosto considerable respecto a la solución MG-CD secuencial. Sin embargo el porcentaje de este sobrecosto de DagOn fue reducido considerablemente de 1451.79 % a 97.43 % para 1 dato y de 782.29 % a 1.24 % con 10 datos cuando utiliza 12 hilos.

Los tiempos de respuesta para dichas configuraciones paralelizadas se muestran en la Fig. 5.15. En esta Figura se puede observar que los tiempos de respuesta se reducen en un orden de magnitud cuando se aplican paralelismo de tareas en las soluciones estudiadas.

Adicionalmente, en la Fig. 5.15 se muestra que sin importar la carga de trabajo, las configuraciones con 10 y 12 hilos (S_{10} y S_{12}) y con 1 o 2 nodos peers por organización (P_1 o P_2) obtienen aproximadamente los mismos tiempos de respuesta. Esto se debe a que se están utilizando todos los cores disponibles de la infraestructura definida y por otro lado, todos los nodos peers se encuentran ubicados en la misma ubicación geográfica.

Es importante notar que, para escenarios de dispersión geográfica, los costos de registro (registro de activo, registro de participante y registro de transacciones) realizados en la red de verificabilidad

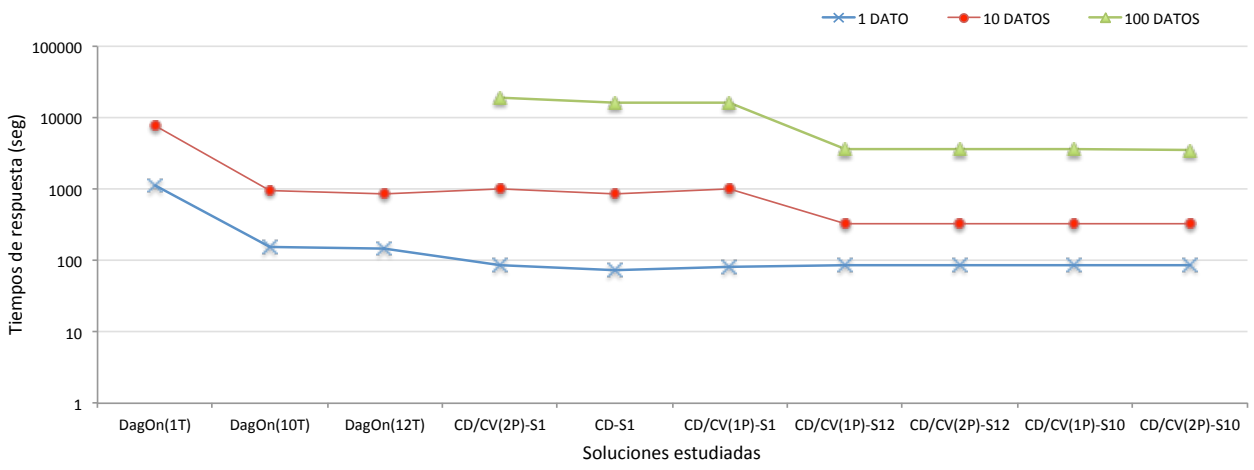


Figura 5.15: Tiempo de respuesta de las soluciones estudiadas considerando paralelismo de tareas con 10 y 12 hilos.

deberían considerar la latencia proporcional a la separación geográfica de los nodos peers que cada organización ha desplegado en la red.

Por ejemplo, asumamos que las organizaciones A y B, separadas geográficamente, ejecutan cada una de ellas una etapa de procesamiento diferente. En tal escenario el componente CD de la organización A deberá invocar a su componente CV para difundir la transacción que realice antes de entregar los resultados al componente CD de la organización B. Esto significa que, dicho protocolo impone retrasos al momento en el que MG-CD/CV de una organización entrega los resultados a otra organización porque se debe confirmar que la transacción ha sido difundida antes de ejecutar otra etapa.

El mejor tiempo obtenido para la solución de MG-CD/CV fue con los parámetros S_{10}-P_2 por encima de S_{12}-P_2. Esto se debe a que cada Si que se define en las etapas de un flujo de trabajo se mapean a la red de verificabilidad como un participante, los resultados con S_{10} son, en promedio, mejores que aquellas configuraciones con S_{12}, dado que al tener menos cantidad de servicios S_i, se tiene una menor gestión de los participantes en la red de verificabilidad.

El análisis presentado hasta el momento revela el sobrecosto de cada una de las soluciones MG-

CD/CV respecto a las versiones secuenciales de cada una de las soluciones y las versiones paralelizadas de DagOn con 10 y 12 hilos de procesamiento.

El siguiente análisis corresponde a medir el sobrecosto de las soluciones MG-CD/CV y DagOn respecto a MG-CD cuando las tres soluciones incorporan paralelismo de tareas con 10 hilos de procesamiento. Los resultados se presentan en la Fig. 5.16.

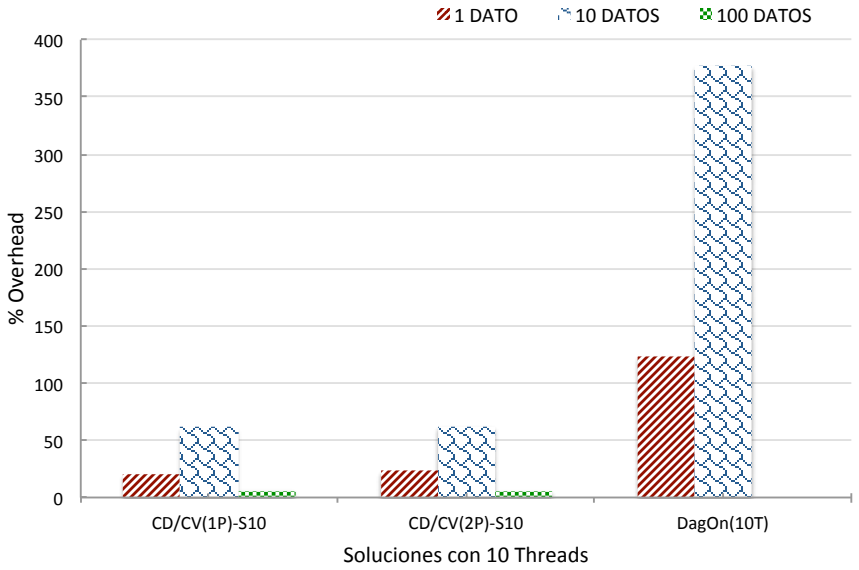


Figura 5.16: Comparativa del sobrecosto de las soluciones MG-CD/CV y DagOn respecto a MG-CD considerando paralelismo de tareas con 10 hilos de procesamiento en las tres soluciones

En la Fig. 5.16 se muestra que el sobrecosto que existe entre la solución MG-CD/CV con 1 o 2 peers respecto a la solución MG-CD con 10 hilos, es prácticamente el mismo. Sin embargo, comparando nuestras soluciones con DagOn, esta solución tiene en promedio un sobrecosto de $82.88\,\%$ adicional respecto a los cuatro resultados de MG-CD/CV con 1 y 2 nodos peers para las cargas de trabajo de 1 y 10 usuarios.

De la misma manera se llevó acabo un análisis comparando el sobrecosto de las soluciones MG-CD/CV y DagOn respecto a MG-CD cuando las tres soluciones incorporan paralelismo de tareas con 12 hilos de procesamiento. Los resultados se presentan en la Fig. 5.17, en dichos resultados

obtenidos las soluciones considerando 12 hilos de procesamiento tienen el mismo comportamiento y sobrecosto similar a los resultados de las soluciones considerando 10 hilos de procesamiento (ver Fig. 5.16) para soluciones propuestas en este trabajo de investigación. Sin embargo, para DagOn cuando la cantidad de hilos es aumentada, si logra obtener un porcentaje de mejora del $19.08\,\%$ para 1 dato y del $42.56\,\%$ para 10 datos.

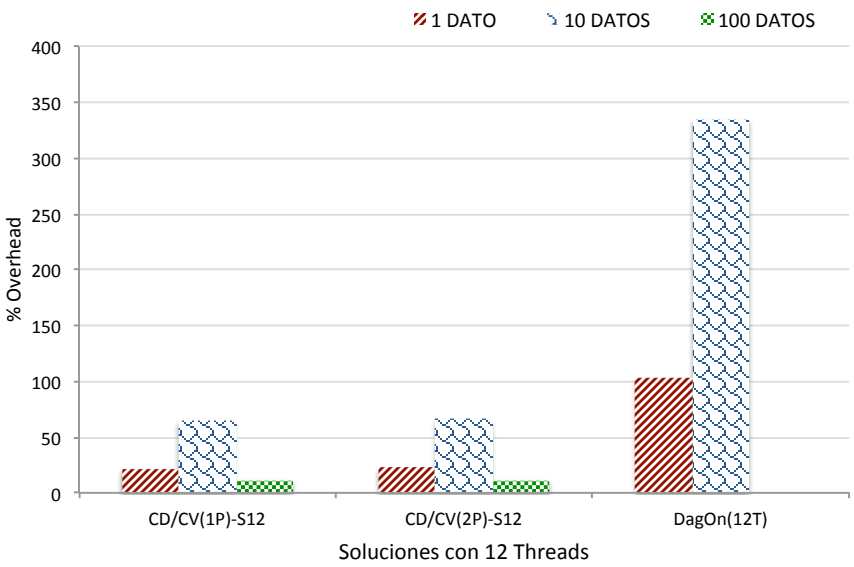


Figura 5.17: Comparativa del sobrecosto de las soluciones MG-CD/CV y DagOn respecto a MG-CD considerando paralelismo de tareas con 12 hilos de procesamiento en las tres soluciones

Los tiempos de respuesta de cada una de las soluciones estudiadas que consideran 10 y 12 hilos se presentan en la Fig. 5.18. Adicionalmente, se incluyó la configuración de DagOn* la cual hace un uso completo de los recursos disponibles de la infraestructura.

Los resultados de la Fig. 5.18 indican que la solución que mejor resultados obtuvo para procesar un dato en particular fue la de DagOn* con un tiempo de 66.71 seg, $21.38\,\%$ más rápido que la mejor configuración de MG-CD/CV que considera el componente de verificabilidad y no realiza un uso completo de la infraestructura como lo realiza DagOn.

Por otro lado, para una carga de 10 datos, las soluciones de MG-CD con 10 y 12 hilos, obtuvieron

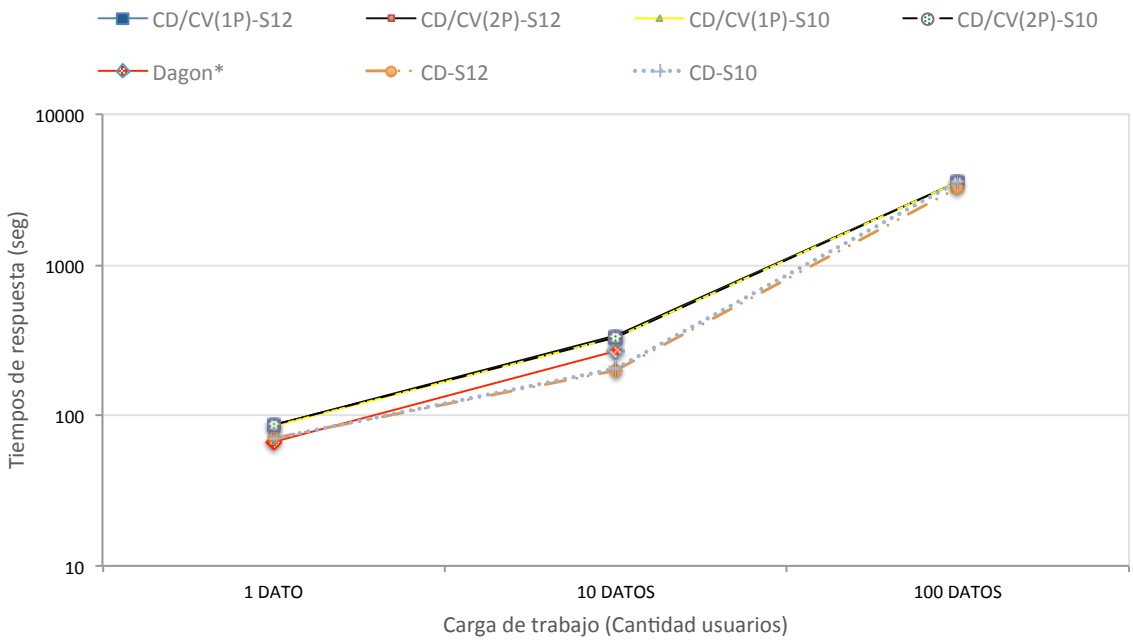


Figura 5.18: Tiempos de respuesta de las soluciones estudiadas considerando paralelismo de tareas.

los mejores tiempos de respuesta. Las configuraciones de MG-CD/CV ($S10-P1$, $S10-P2$, $S12-P1$ y $S12-P2$) en promedio tardan $39.27\,\%$ más que la mejor solución de MG-CD ($S12$). Sin embargo el resultado a destacar se encuentra al aplicar una carga de 100 datos, ya que la solución del estado del arte no se pudo evaluar correctamente dado el flujo de trabajo planteado en este estudio de caso. Mientras que dicha carga si pudo ser evaluada con las soluciones propuestas en esta investigación. La configuración ganadora para procesar dicha carga considerando el componente de verificabilidad con la solución MG-CD/CV fue con los parámetros S10-P2 (10 servicios por etapa y dos nodos peers por organización), la cual tiene un costo adicional del $8.80\,\%$ respecto a la mejor solución de MG-CD que fue con 12 hilos de procesamiento.

Los resultados anteriores indican que el costo adicional de considerar el componente de verificabilidad continua con la solución MG-CD/CV disminuye en la medida que aumenta la carga de trabajo a procesar. Esto se debe a que el diseño propuesto en esta investigación para generar los registros en la red de verificabilidad, considera un registro transaccional por el lote (directorio que

contiene trayectorias de un usuario) entrante y saliente de cada una de las etapas de procesamiento del flujo de trabajo y no por cada trayectoria de cada usuario. Esta consideración a grano grueso permite que los tiempos de los registros en la de verificabilidad no son dependientes de la cantidad de trayectorias de un usuario y sea prácticamente constante, mientras que para el componente de entrega continua, al tener mayor cantidad de trayectorias a procesar, mayor es el tiempo de respuesta del procesamiento de dichas tareas. En este sentido, se estima que a mayor carga de trabajo a evaluar en la solución propuesta MG-CD/CV, menor es el sobrecosto de utilizar una red verificable respecto a la solución MG-CD aplicándole la misma carga.

Los tiempos de respuesta junto con el porcentaje de sobrecarga de las soluciones propuestas en este trabajo de investigación respecto a la mejor configuración del estado del arte (DagOn*) para una carga de 1 y 10 datos se presentan a continuación.

5.2.7.4.1. Carga de 1 dato. Los tiempos de respuesta promedio de todas las configuraciones de las tres soluciones estudiadas aplicando una carga de 1 dato (D_1 igual para todas las soluciones) se ilustran en la Fig. 5.19. En esta Figura se muestra de mayor a menor los tiempos de respuesta obtenidos por las diferentes soluciones tanto secuenciales como paralelizadas. Sin embargo, dado que la carga de trabajo es solo un dato, utilizar más de un servicio resulta innecesario y genera un costo de gestión de hilos por parte del sistema operativo. Es por ello que en promedio la solución de MG-CD/CV que mejores resultados obtiene en tiempos de respuesta es la representada en la gráfica como CD/CV(1P)-S1, ya que solo hace uso de un hilo de procesamiento que es suficiente para procesar la carga de 1 dato.

El porcentaje de sobrecarga de la mejor solución de MG-CD/CV (CD/CV(1P)-S1) respecto a la mejor configuración de la solución del estado del arte DagOn* es del $25.50\,\%$. Cabe aclarar que nuestras soluciones propuestas no consideran un mecanismo de gestión de recursos como lo hace la mejor configuración del estado del arte. La desviación estándar y porcentaje de confianza de los resultados obtenidos de las 15 iteraciones para una carga de 1 dato se presentan en la tabla 5.9. En

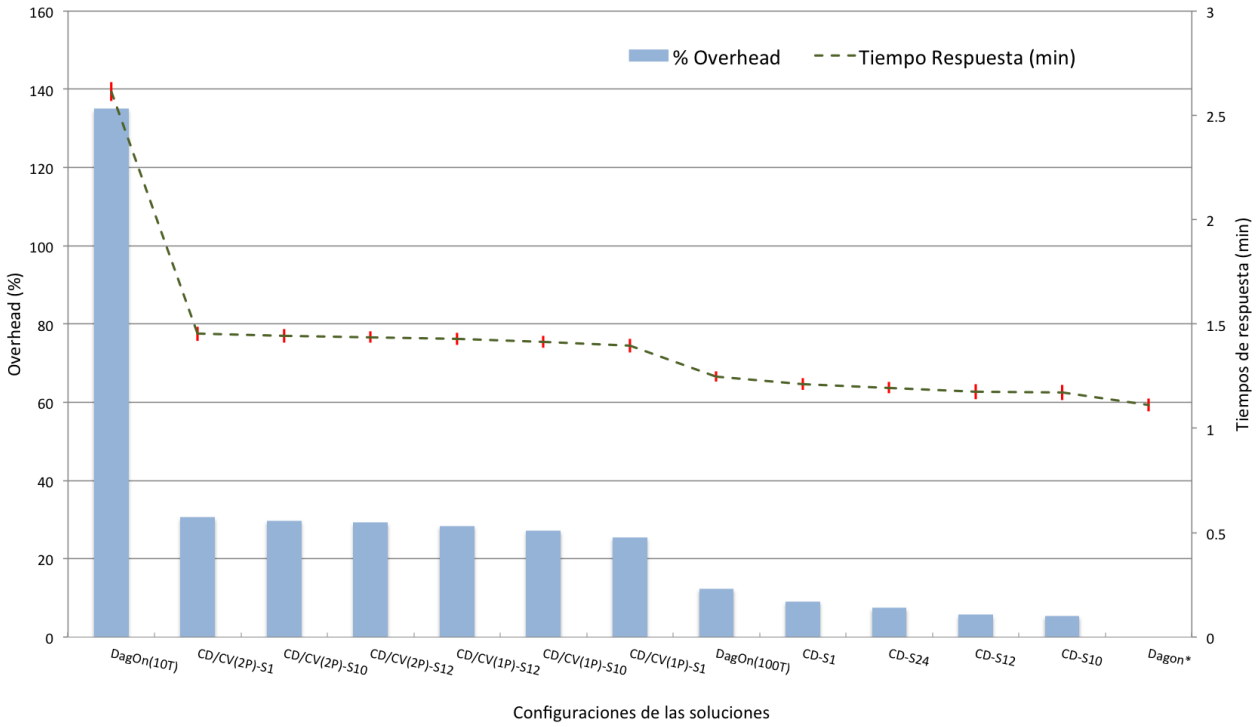


Figura 5.19: Porcentaje de sobrecarga de todas las soluciones respecto a DagOn* para una carga de 1 datos.

	MG-CD				MG-CD/CV						DagOn				
	S1	S10	S12	S24	S1-P1	S10-P1	S12-P1	S1-P2	S10-P2	S12-P2	T1	T10	T12	T100	T*
Promedio	72.7779	70.3221	70.4930	71.6804	83.7285	84.8606	85.6727	87.1320	86.5686	86.2528	1129.3628	156.7630	143.6899	74.9593	66.7133
Desviación Estándar	1.68438	1.90	2.26	2.13	1.52	2.03	1.69	1.93	1.69	1.76	2.69	2.00	2.35	1.73	1.63
% Desviación	2.31441	2.70	3.21	2.97	1.82	2.39	1.97	2.22	1.96	2.04	0.24	1.28	1.63	2.31	2.44
% Confianza	97.69	97.30	96.79	97.03	98.18	97.61	98.03	97.78	98.04	97.96	99.76	98.72	98.37	97.69	97.56

Tabla 5.9: Desviación estándar y porcentaje de confianza de las 15 iteraciones para todas las configuraciones de las soluciones estudiadas aplicando una carga de 1 dato.

dicha tabla se puede observar que el porcentaje de confianza de los valores obtenidos para las tres soluciones estudiadas aplicando una carga de 1 dato con 15 iteraciones, varia entre el $96.79\,\%$ y el $99.76\,\%$ el cual consideramos es un porcentaje aceptable para no realizar más iteraciones de dichas pruebas.

5.2.7.4.2. Carga de 10 datos. Los tiempos de respuesta promedio y porcentaje de sobrecarga de todas las configuraciones de las tres soluciones estudiadas aplicando una carga de 10 datos (D_{10} igual para todas las soluciones) se ilustran en la Fig. 5.20 junto con el porcentaje de sobrecarga de

todas las soluciones respecto a la mejor configuración de DagOn.

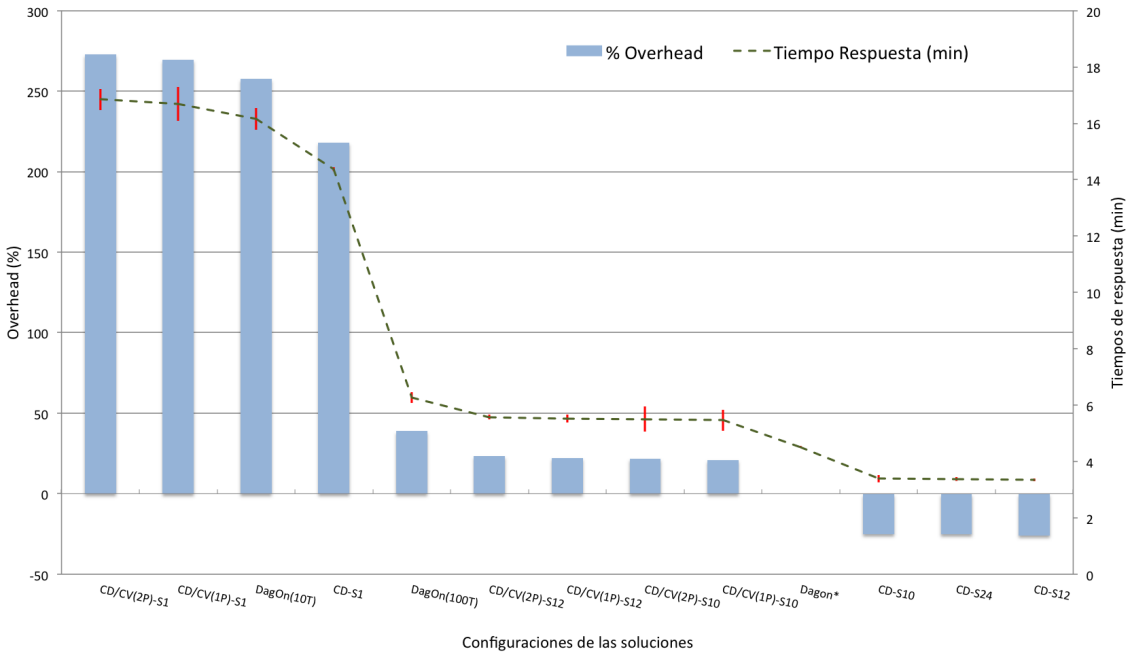


Figura 5.20: Porcentaje de sobrecarga de todas las soluciones respecto a DagOn* para una carga de 10 datos.

La desviación estándar y porcentaje de confianza de los resultados obtenidos de las 15 iteraciones para una carga de 10 datos se presentan en la tabla 5.10. En dicha tabla se puede observar que el porcentaje de confianza de los valores obtenidos para las tres soluciones estudiadas aplicando una carga de 10 dato con 15 iteraciones, varia entre el $91.67\,\%$ y el $99.79\,\%$ el cual consideramos es un porcentaje aceptable para no realizar más iteraciones de dichas pruebas.

	MG-CD				MG-CD/CV						DagOn				
	S1	S10	S12	S24	S1-P1	S10-P1	S12-P1	S1-P2	S10-P2	S12-P2	T1	T10	T12	T100	T*
Promedio	862.4	203.2	201.0	203.0	1001.7	327.9	331.6	1010.7	330.3	334.4	7609.2	969.3	873.1	376.3	271.1
Desviación Estándar	22.91	3.76	2.64	3.34	36.39	11.58	7.54	23.76	5.33	4.32	16.17	8.47	37.68	26.81	22.58
% Desviación	2.66	1.85	1.31	1.65	3.63	3.53	2.27	2.35	1.61	1.29	0.21	0.87	4.32	7.12	8.33
% Confianza	97.34	98.15	98.69	98.35	96.37	96.47	97.73	97.65	98.39	98.71	99.79	99.13	95.68	92.88	91.67

Tabla 5.10: Desviación estándar y porcentaje de confianza de las 15 iteraciones para todas las configuraciones de las soluciones estudiadas aplicando una carga de 10 datos.

5.2.7.4.3. Carga de 100 datos. Con respecto a la carga de 100 datos, dado que la solución de DagOn no permitió evaluar dicha carga, se realizó una comparativa con la mejor configuración encontrada de las soluciones propuestas, la cual corresponde a la solución MG-CD con S_{12}.

El porcentaje de sobrecarga de todas las soluciones respecto a la mejor configuración de la solución MG-CD (S_{12}), se presenta a continuación en la Fig. 5.21:

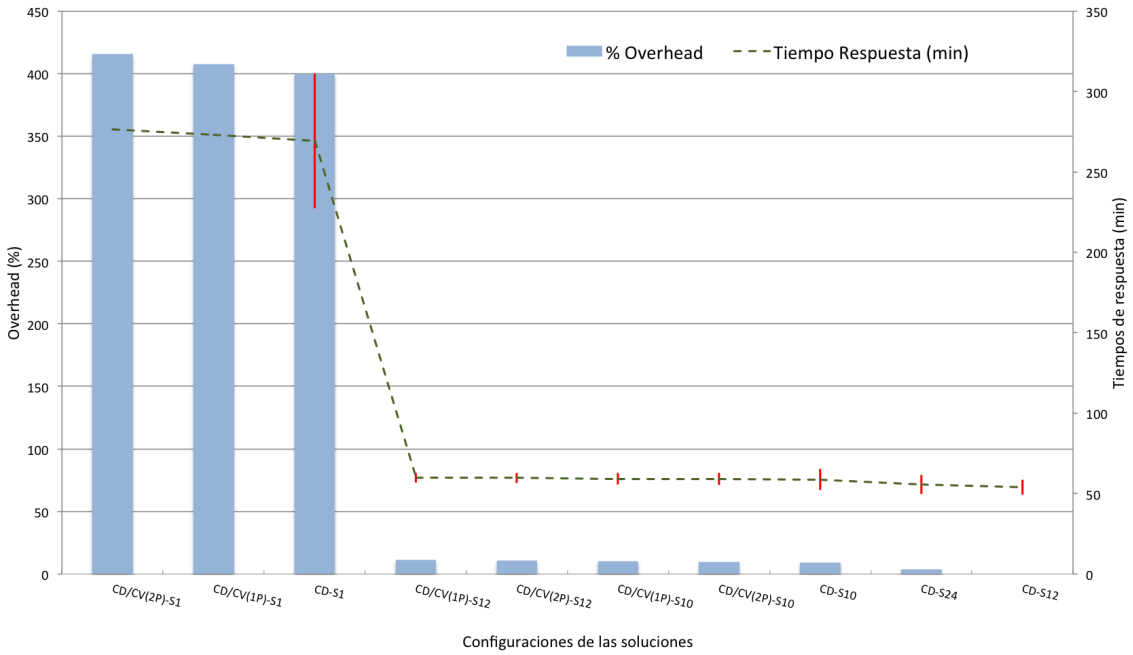


Figura 5.21: Porcentaje de sobrecarga de todas las soluciones respecto a MG-CD con 12 servicios para una carga de 100 datos.

La desviación estándar y porcentaje de confianza de los resultados obtenidos de las 15 iteraciones para una carga de 100 datos se presentan en la Tabla 5.11. En dicha tabla se puede observar que el porcentaje de confianza de los valores obtenidos para la solución MG-CD/CV aplicando una carga de 100 dato con 15 iteraciones, varia entre el $93.89\,\%$ y el $96.63\,\%$ el cual consideramos es un porcentaje aceptable para no realizar más iteraciones de las pruebas de MG-CD/CV. Con respecto a la solución MG-CD, las configuraciones paralelizadas (S_{10}, S_{12}, S_{24}) que fueron evaluadas en 15 iteraciones obtuvieron un porcentaje de confianza entre el $89.22\,\%$ y el $93.11\,\%$. Sin embargo, la configuración secuencial de MG-CD (S_1) obtuvo el porcentaje de confianza más bajo de toda la

experimentación con un $82.67\,\%$ y no se realizaron más iteraciones de dicha prueba dado que la carga de trabajo era considerable y al tener un solo servicio en cada etapa del flujo de trabajo, los tiempos de experimentación de dicha prueba eran considerables para el alcance del proyecto de investigación y no aportaba en gran medida al objetivo principal de este trabajo.

	MG-CD				MG-CD/CV						DagOn				
	S1	S10	S12	S24	S1-P1	S10-P1	S12-P1	S1-P2	S10-P2	S12-P2	T1	T10	T12	T100	T*
Promedio	16903.8	3357.3	3229.9	3341.7	16340.7	3553.2	3587.3	19196.9	3541.7	3584.3	!	!	!	!	!
Desviación Estándar	2929.2	231.2	277.1	360.2	551.3	210.7	173.6	1095.5	216.4	188.6	!	!	!	!	!
% Desviación	17.33	6.89	8.58	10.78	3.37	5.93	4.84	5.71	6.11	5.26	!	!	!	!	!
% Confianza	82.67	93.11	91.42	89.22	96.63	94.07	95.16	94.29	93.89	94.74	!	!	!	!	!

Tabla 5.11: Desviación estándar y porcentaje de confianza de las 15 iteraciones para todas las configuraciones de las soluciones estudiadas aplicando una carga de 100 datos.

5.2.7.5. Análisis del componente de verificabilidad de la mejor solución MG-CD/CV

En esta sección se realiza un análisis sobre los tiempos de construcción de la red de verificabilidad para la mejor configuración obtenida con la solución propuesta MG-CD/CV. Dado los resultados previamente descritos, consideramos que la mejor configuración encontrada para la solución MG-CD/CV es la de 2 nodos peers con 10 servicios por etapa (CD/CV-S10-P2). Esta consideración se toma teniendo en cuenta que fue la configuración que incorpora el componente de verificabilidad continua y que procesa la mayor carga de trabajo evaluada en esta experimentación (carga de 100 usuarios) en el menor tiempo posible. Si bien los resultados entre CD/CV-S10-P2 y CD/CV-S10-P1 son muy cercanos, es recomendable tener una mayor cantidad de nodos peers por organización en caso de una eventual falla de un nodo peer. En el caso de la configuración de 1 nodo peer (CD/CV-S10-P1) una falla en la comunicación con el único nodo peer de una organización podría ser catastrófico, ya que podría inhabilitar el acceso de dicha organización a la red de verificabilidad y no poder emitir registros a la red.

El análisis realizado en las siguientes secciones corresponde a una iteración de las 15 que fueron evaluadas para la configuración CD/CV-S10-P2 con una carga 100 datos. A continuación se presentan los tiempos de ejecución de los procesos que se ejecutan al desplegar nuestra solución propuesta.

Entre ellos se encuentran los procesos del componente de entrega continua y los del componente de verificabilidad continua abordado por los sub-componentes de Hyperledger Fabric y de Hyperledger Composer.

En la Fig. 5.22 se ilustran los tiempos de ejecución obtenidos de cada uno de los procesos o tareas en las diferentes fases (declaración, despliegue y operación) del marco de gestión:

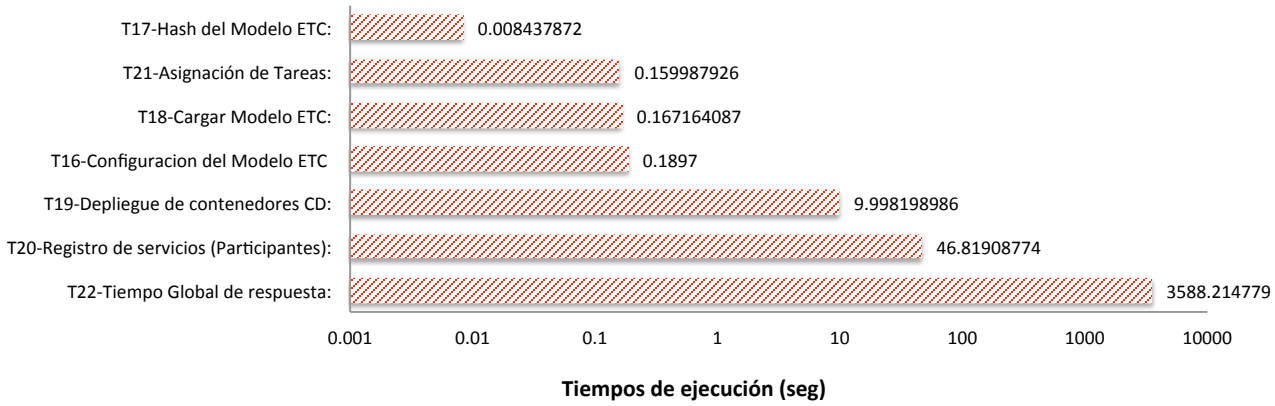


Figura 5.22: Tiempos de ejecución de procesos ejecutados por el Marco de Gestión.

Con respecto a los procesos que se muestran en la Fig. 5.22, las tareas de la fase de preparación (T16,T17,T18) corresponden al tiempo para generar los archivos de configuración necesarios para garantizar una entrega continua en el flujo de trabajo a partir de la notación dada por el usuario en el marco de gestión utilizando el modelo ETC. Posteriormente, en la fase de despliegue se ejecuta la tarea T19, la cual consiste en el despliegue o puesta en marcha de los servicios de cada una de las etapas las cuales denominamos contenedores CD. Una vez los servicios se encuentran correctamente desplegados, el gestor se encarga de registrarlos como participantes de la red de verificabilidad (T20) y les asigna la carga de trabajo que deberán ejecutar cada uno (T21). Finalmente, el flujo de trabajo es puesto en marcha en la fase de operación y se calculan el tiempo de respuesta (T22) de procesar dicha carga de 100 datos a través del flujo de trabajo.

Con respecto a los procesos que se ejecutan para la construcción, despliegue y operación de la red de verificabilidad, estos son ejecutados en dos fases: la primera corresponde a la infraestructura de la

red de verificación basada en la construcción de los bloques encadenados, y la segunda corresponde a la red de negocios que modela la lógica de las transacciones que se registran en los bloques encadenados. Los tiempos empleados en cada una de estas fases se describen a continuación:

5.2.7.5.1. Fase 1 de Verificación Continua - Creación y despliegue de la red de verificabilidad En esta fase se construye, despliega y se deja en producción la red de verificabilidad por medio del módulo de Hyperledger Fabric. Los tiempos obtenidos en cada una de los procesos que se realizan en esta fase, se muestran en la Fig. 5.23.

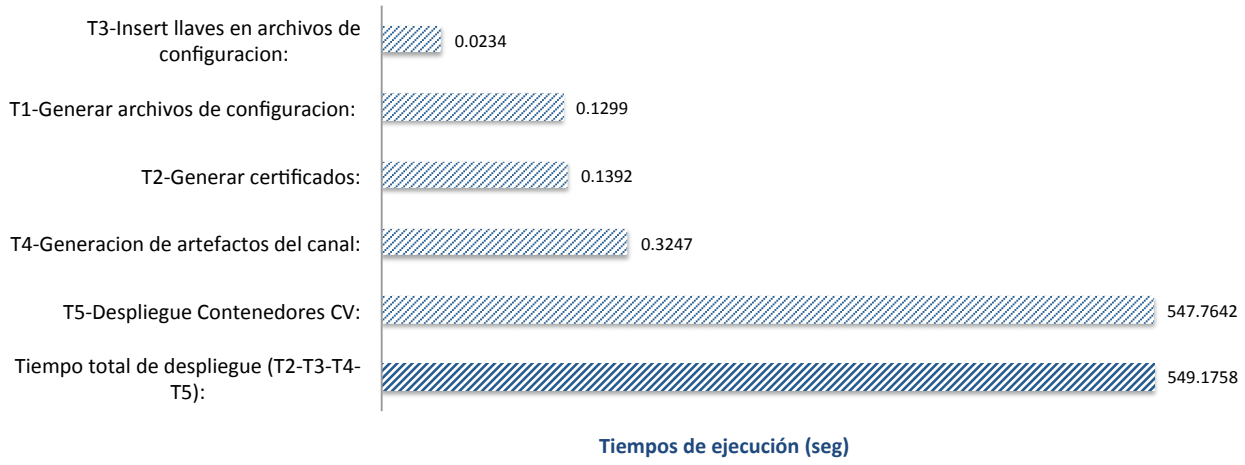


Figura 5.23: Tiempos de ejecución del módulo 1 (Hyperledger Fabric) del componente de verificabilidad continua.

5.2.7.5.2. Fase 2 de Verificación Continua - Red de negocio En esta fase se construye, despliega y se instala la red de negocio en la red de verificabilidad por medio del módulo de Composer. Los tiempos obtenidos en cada uno de los procesos que se realizan en esta fase, se muestran en la Fig. 5.24:

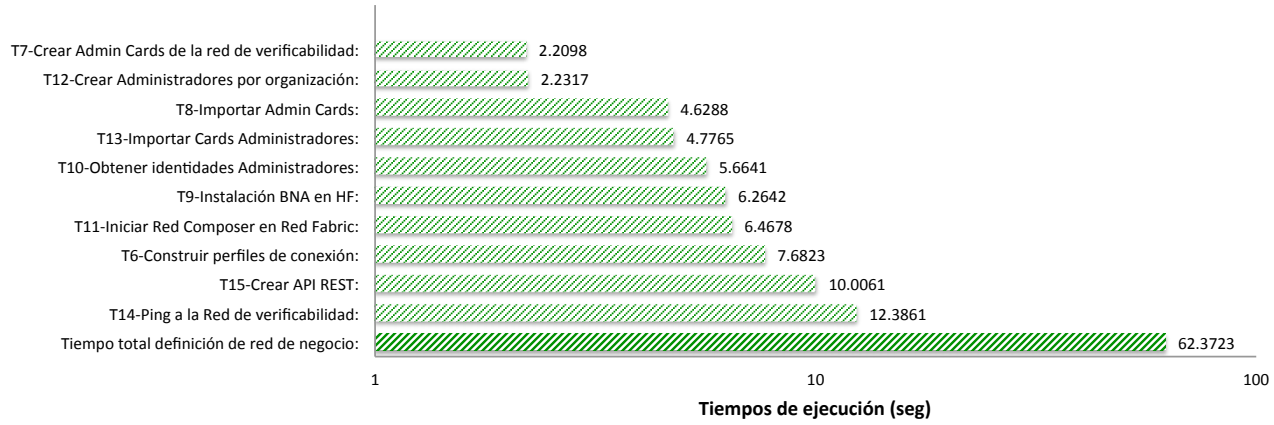


Figura 5.24: Tiempos de ejecución del módulo 2 (Hyperledger Composer) del componente de verificabilidad continua.

Los resultados globales se muestran en la Fig. 5.25. Estos resultados globales son los descritos en la fase 1 y 2 del proceso de verificabilidad continua junto con los procesos de entrega continua.

Entre los tiempos anteriormente mostrados y consolidados en la Fig. 5.25 se puede destacar que para la mejor configuración obtenida en la fase comparativa del estudio de caso de movilidad de usuarios se tiene que el tiempo requerido para el despliegue del componente CV (conformado por las tareas de T1 a T5 del componente de Hyperledger Fabric y de T6 a T15 para el módulo de Hyperledger Composer) es mucho mayor al requerido por el componente CD (Tareas T16 a la tarea T21) para realizar la entrega continua. Un escenario que considere mayor cantidad de nodos peers por organización que la descrita por la mejor configuración, generara un mayor sobre costo de despliegue del componente de CV con respecto al componente de CD. La tarea o proceso identificado como T22 incluye el tiempo de procesar toda la carga de trabajo junto con los tiempos de registro en la red de verificabilidad, los cuales incluye registro de activo y registro de transacción. El costo que

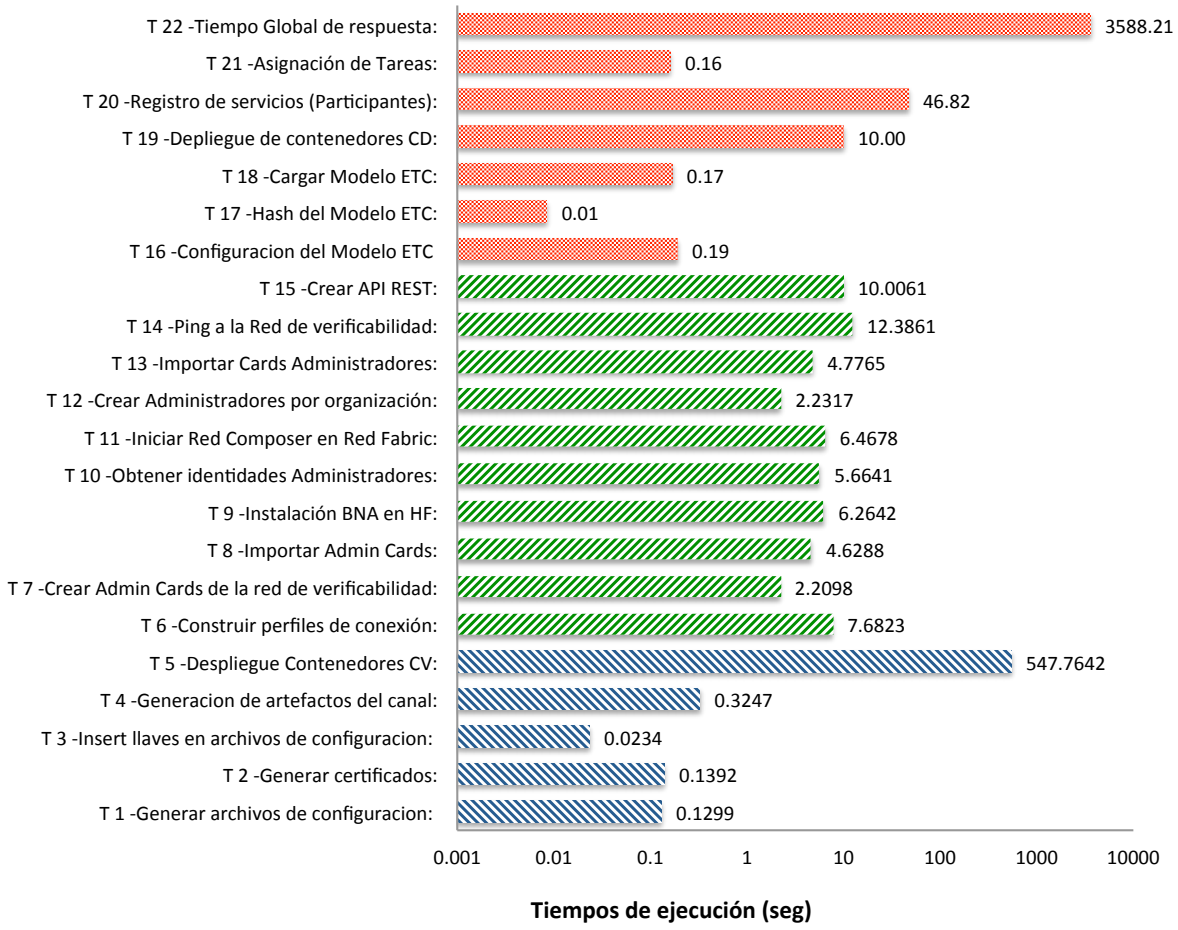


Figura 5.25: Consolidado de los tiempos obtenidos tanto en la verificabilidad continua como la entrega continua para MG-CD/CV con 2 nodos peers y 10 servicios por etapa.

genera cada uno de estos registros sobre el tiempo de respuesta global, se presenta en detalle en el segundo estudio de caso.

5.3 Estudio de Caso dos: Datos Medioambientales

Un nuevo estudio de caso se definió en este trabajo de investigación con el objetivo de ver la factibilidad de la solución propuesta (MG-CD/CV) para ser aplicado a un escenario IoT completamente diferente al descrito en el estudio de caso uno de movilidad de usuarios.

La diferencia entre el estudio de caso de movilidad y este segundo caso de estudio radica en el uso directo de sensores o dispositivos IoT con el fin de considerar un flujo de trabajo desde la adquisición de los datos recabados por dichos dispositivos, ya que esta etapa del flujo de trabajo no fue considerada en el estudio de caso de movilidad (los datos ya estaban recabados por un tercero y ubicados en la etapa de almacenamiento).

A continuación, se describen las actividades que permitieron conducir el estudio de caso sobre Datos Medioambientales usando el Gestor CD/CV: 1) Las características del estudio de caso de datos medioambientales; 2) La preparación, despliegue y operación del flujo de trabajo, para el procesamiento de información del estudio de caso usando el Marco de Gestión CD/CV; 3) La variación experimental que permitirá evaluar diferentes configuraciones del Gestor CD/CV; 4) Las métricas para evaluar el rendimiento de dichas configuraciones del Gestor CD/CV y 5) El análisis y discusión de los resultados obtenidos.

5.3.1 Descripción del estudio de caso: Datos Medioambientales

El segundo estudio de caso definido en este trabajo de investigación considera la adquisición de los datos en el flujo de trabajo con verificabilidad continua. Esta fase de adquisición básicamente son las aplicaciones encargadas de conectarse directamente con los dispositivos IoT. El proceso de adquisición se ubica en el paradigma de computación en la niebla (comúnmente conocido como Fog Computing) con el fin de obtener las ventajas que brinda el uso de este paradigma en un escenario IoT [8].

Los datos capturados, son manejados a través de diferentes procesos (preprocesamiento, preservación) que en conjunto conforman una etapa del flujo de trabajo para este estudio de caso.

El objetivo de manejar estos datos es realizar una predicción ambiental operativa. Esta predicción consiste en determinar cuando se podría alcanzar un determinado umbral de temperatura, esto con el fin de emitir o entregar una alerta. Por ejemplo, en un bosque donde se tienen desplegados sensores de temperatura, una alerta de que se ha superado un determinado umbral de temperatura podría

prevenir o evitar un incendio forestal en dicha zona.

Dispositivos IoT y datos capturados.

A diferencia del estudio de caso de movilidad anteriormente descrito, en este estudio de caso se consideró el flujo de los datos desde su etapa de adquisición. Para ello, fueron utilizados los dispositivos IoT descritos en la Tabla 5.12.

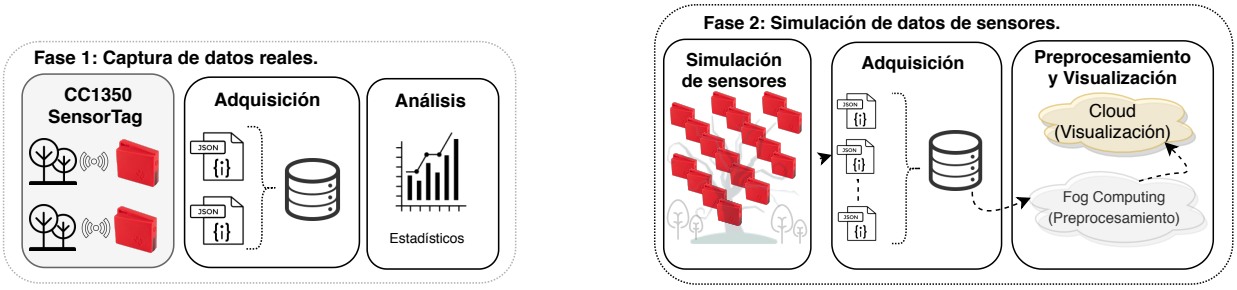
Dispositivos IoT	
Sensor	CC1350 SensorTag
Descripción	CC1350STKUS Simplelink CC1350 SensorTag Bluetooth and Sub-1GHz Long Range Wireless Development Kit.
Cantidad	2
Sensor	CC1350 SensorTag
Descripción	SimpleLink™ Bluetooth low energy/Multi-standard SensorTag.
Cantidad	2

Tabla 5.12: Infraestructura Hardware del estudio de caso 2 - Dispositivos IoT

Estos dispositivos permiten capturar datos tales como temperatura del medio ambiente, movimiento en nueve ejes, aceleración, humedad, intensidad lumínica, entre otros. Sin embargo, por efectos prácticos de este trabajo de investigación, sólo se tuvieron en cuenta los datos de temperatura del medio ambiente para el preprocesamiento a través del flujo de trabajo descrito en la sección 5.3.1.

Mediante el uso de cuatro sensores para la captura de los datos, se tomaron un total de 20418 muestras de temperatura en el Centro de Investigación y de Estudios Avanzados del IPN (Cinvestav) en Ciudad Victoria, México. Las muestras fueron adquiridas en diferentes periodos de tiempo con el fin de tener una base de datos inicial (Fig. 5.26(a)). Esta base de datos fue analizada con el fin de obtener la distribución de estos datos y realizar un simulador (Fig. 5.26(b)) que permita generar datos sintéticos basados en la distribución de los datos reales y de este modo simular una cantidad variable de sensores mucho mayor a la cantidad disponible de sensores. Esto se realizó con el fin de simular escenarios en los cuales se tiene desplegado gran cantidad de sensores y se recaban grandes volúmenes de información (mayores a 1000 datos).

Tanto los datos de temperatura reales o en caso de ser simulados, son enviados al Fog Computing



(a) (b)

Figura 5.26: Alternativas para la obtención de datos en el estudio de caso medioambiental. (a) Adquisición de datos reales o (b) Generación de datos simulados.

donde se encuentra ubicada el Gestor CD/CV (solucion MG-CD/CV) con el fin de acercar el procesamiento de los datos recabados por los dispositivos IoT a la computación en la nube (Cloud) donde pueden ser visualizados por cualquier entidad u organización para la toma de decisiones.

5.3.2 Preparación, despliegue y operación del flujo de trabajo usando el Marco de Gestión CD/CV

En esta sección se describe la preparación del prototipo CD/CV para crear, desplegar y poner en operación el flujo de trabajo que es objeto de estudio.

5.3.2.1. Preparación del prototipo Gestor CD/CV

Tal como se mencionó en la sección 4.3.2.1, para preparar el prototipo CD/CV se realizan dos tareas, la primera es una tarea declarativa y la segunda es una tarea de configuración.

5.3.2.1.1. Declaración del Modelo ETC. Al igual que en el escenario anterior, en la declaración inicial del prototipo del Gestor CD/CV se determinó el número de etapas y servicios.

El flujo de trabajo anteriormente descrito se declara en el modelo ETC de la siguiente forma:

$$W = \{St_1\}, \text{ donde:}$$
$$St_1 \rightarrow \text{Etapa de Preprocesamiento y preservación.}$$

Posteriormente, en la declaración del modelo ETC, se define la cantidad de servicios a ser desplegados para la única etapa del flujo de trabajo que es objeto de estudio. La cantidad de servicios y/o aplicaciones para este estudio de caso se declaran en el modelo de interconexión ETC de la siguiente forma:

$$St_{1,NuSer} = N_1 \rightarrow \text{Número de servicios de Preprocesamiento}$$

El valor de $St_{1,NuSer}$, se define en la variación experimental de este estudio de caso (sección 5.3.4). Se espera que a mayor cantidad de sensores disponibles enviando datos (solicitudes), mayor es el número de servicios necesarios para atender dichas solicitudes.

Por ejemplo, para una configuración $St_{1,NuSer} = 5$, la etapa descrita en el flujo de trabajo de este estudio de caso, quedaría denotada en el modelo de la siguiente forma:

$$W = \{St_1\}, \text{ donde:}$$
$$St_1 = \{S1_{St1}, S2_{St1}, S3_{St1}, S4_{St1}, S5_{St1}\} \rightarrow \text{Etapa de Preprocesamiento y entrega con 5 servicios (clones)}$$

Por defecto, para este estudio de caso, al igual que el de movilidad, se tiene solo un servicio que representa la base de datos donde son almacenados los resultados obtenidos en el flujo de trabajo.

Por otro lado, para concluir la fase de preparación de este estudio de caso, es necesario configurar los componentes del modelo ETC para la verificación continua. En este sentido, el número de organizaciones para este estudio de caso corresponde a una organización dado que se definió en el flujo de trabajo una sola etapa. Para el flujo de trabajo descrito anteriormente, se tiene una organización que realiza el preprocesamiento y la entrega de los datos a la nube donde son exhibidos los resultados del flujo de trabajo.

En cuanto al número de nodos que pertenecen a la red de verificabilidad, como se describió anteriormente, se tienen tres tipos de nodos en la red de verificabilidad (nodo orderer, nodos CAs y nodos peers), para este estudio de caso se tiene en la red de verificabilidad: 1) un nodo orderer, 2) un nodo CA y 3) $N_{peers} = 2$ que fue la mejor configuración obtenida en el estudio de caso

uno. La organización y la cantidad de peers de la red de verificabilidad se declaran en el modelo de interconexión ETC de la siguiente forma:

$$
\begin{gathered}
PxO = 2 \\
Orgs = \{Org1\} \\
Nodos = \{Peers_{Org1}, Nodos_{CAs}, N_{or}\} \\
Donde: \\
Peers_{Org1} = \{N_{pe0-Org1}, N_{pe1-Org1}\} \in Org1 \\
Nodos_{CAs} = \{N_{ca-Org1}\} \\
N_{or} = \text{Nodo para establecer consenso.}
\end{gathered}
$$

Como resultado de la notación de los parámetros previamente descritos en el Modelo ETC, se muestra en la Fig. 5.27, una representación visual de la red de verificabilidad conformada para este estudio de caso.

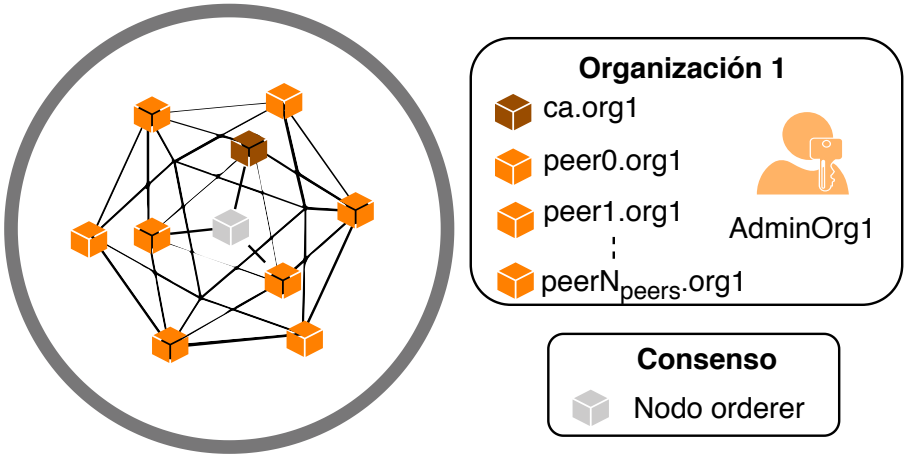


Figura 5.27: Red de verificabilidad para el estudio de caso medioambiental.

Por defecto, el marco de gestión configura un $administrador$ (AdminOrg1) para la única organización definida en este estudio de caso.

5.3.2.1.2. Configuración del gestor CD/CV La tarea de configuración en la fase de preparación del sistema CD/CV para este estudio de caso, consistió en definir las siguientes rutas para el componente de entrega continua (CD):

- Fuente de datos: Esta ruta apunta a los datos recabados (reales o simulados) por los diferentes sensores o dispositivos IoT con la cual iniciará el flujo de trabajo de este estudio de caso.
- ETC de cada servicio: Tres rutas son definidas en este apartado para cada uno de los servicios de única etapa definida para este estudio de caso (Preprocesamiento y preservación).
- Configuración de contenedores virtuales: Esta ruta apunta a la configuración de Docker para cada imagen de contenedor virtual de los servicios considerados en la etapa del flujo de trabajo de este estudio de caso. Esta ruta es usada por el gestor para crear instancias de contenedor de las imágenes de los servicios de preprocesamiento y preservación, dado este estudio de caso.

Como se mencionó previamente, el gestor hace que todas las rutas antes mencionadas apunten a un volumen compartido. Este volumen compartido se desplegó sobre el sistema de archivos del servidor descrito en la Tabla 5.13. Con respecto a las rutas del componente de verificabilidad continua (CV) las rutas de Hyperledger Fabric e Hyperledger Composer son establecidas por defecto por parte del Gestor CD/CV y no son dependientes del estudio de caso.

5.3.2.2. Despliegue del flujo de trabajo creado por CD/CV

El prototipo del Gestor CD/CV fue evaluado usando la infraestructura hardware (dispositivos IoT y nube) descrita en las Tablas 5.12 y 5.13 respectivamente, junto con la infraestructura software descrita en la Tabla 5.14.

Hardware	
Servidor	Compute12
Descripción	Asignados a la nube. Cuenta con Docker CE
Procesador	Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz
Memoria RAM	64 GB
Almacenamiento	1 x Disco duro de 2.7 TB 1 x Disco duro de 2.7 TB
Sockets	1
Cores por Socket	12
Threads por core	1
Ip	10.0.0.42 148.247.201.224

Tabla 5.13: Infraestructura hardware del estudio de caso 2 - Infraestructura de nube

Software	
Descripción	**Versiones**
Sistema Operativo con instalación tipo "Compute Node"	Linux CentOS 7 x64
Cloudera	Manager
Docker y Docker Compose	>= 17.06.2-ce
cURL	Latest
Go	1.11.x
Python	2.7.x
Node.js Runtime y NPM	8.9.x
Hyperledger Fabric	1.2
Hyperledger Composer	0.20
Android Studio	3.2.1

Tabla 5.14: Infraestructura software del estudio de caso 2

La implementación de cada una de las etapas de dicho flujo de trabajo fueron encapsuladas en contenedores virtuales usando la plataforma Docker. Se definió un volumen compartido para que los contenedores de las etapas realizaran la entrega continua, este volumen fue desplegado en el sistema de archivos del servidor descrito en la Tabla 5.13. Los contenedores de las etapas de este flujo de trabajo también fue desplegado en dicha infraestructura.

En este punto, el prototipo se encuentra preparado para ser desplegado con diferentes configuraciones sobre una infraestructura dada.

5.3.2.3. *Operación del flujo de trabajo creado por CD/CV*

Para este estudio de caso en particular, se definió una etapa para el flujo de trabajo con verificabilidad continua (ver figura 5.28), la cual consiste en aplicar los siguientes procesos una vez el flujo de trabajo ha sido desplegado y los datos de los sensores han sido adquiridos: 1) Un preprocesamiento a los datos de temperatura capturados por los sensores (dispositivos IoT), el cual consiste en identificar y eliminar valores atípicos que han sido recabados por los dispositivos IoT en la fase de adquisición y 2) Almacenar los datos resultantes en una base de datos que se encuentra en la nube (*Cloud*) y donde pueden ser accedidos por cualquier organización para su posterior análisis o toma de decisiones a través de un framework para la visualización de datos IoT, como por ejemplo el servicio web de Amazon IoT (AWS IoT, por sus siglas en inglés) [65].

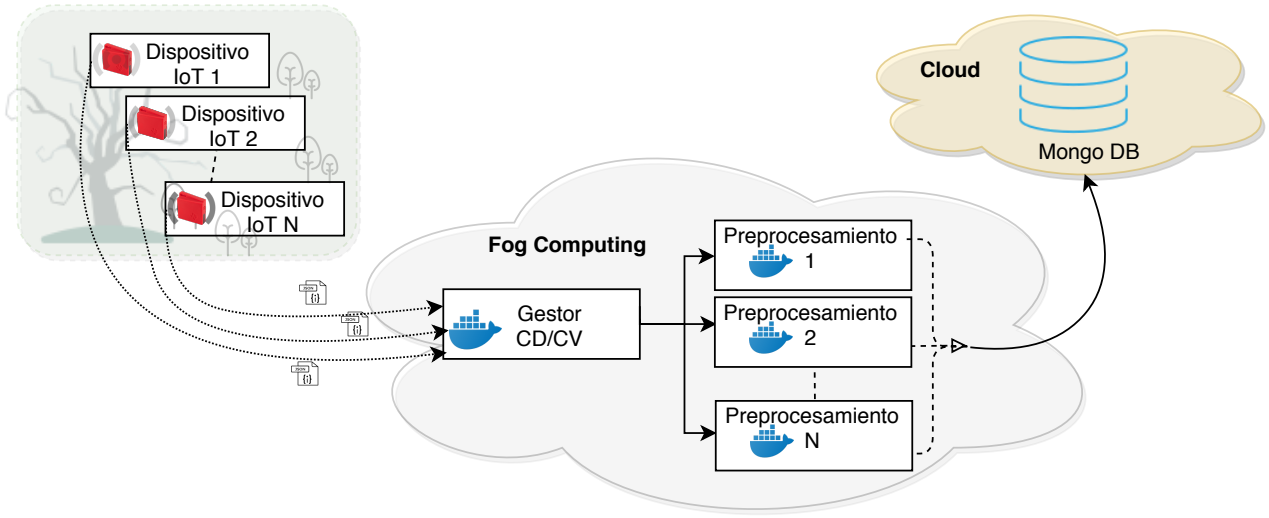


Figura 5.28: Diseño del flujo de trabajo para el estudio de caso 2.

5.3.3 Soluciones estudiadas

En este escenario medioambiental se estudió la solución propuesta MG-CD/CV que fue descrita en la sección 5.2.3 del primer estudio de caso, con el objetivo de ver la factibilidad de aplicar el Gestor CD/CV a un escenario IoT diferente al de movilidad.

5.3.4 Variación experimental

Dado que el propósito de este estudio de caso es principalmente ver la factibilidad de aplicar el gestor propuesto (CD/CV) a un escenario IoT diferente al de movilidad y considerando además el alcance de este proyecto de investigación, se decidió evaluar el Marco de Gestión CD/CV con tres pruebas diferentes las cuales se describen a continuación:

1. La prueba número uno consistió en una simulación de 10 sensores emitiendo cada uno 100 datos de temperatura para un total de 1000 datos adquiridos que fueron procesados a través del flujo de trabajo descrito anteriormente. La cantidad de servicios en la única etapa del flujo de trabajo para este estudio de caso, fue de un servicio con el fin de evaluar el comportamiento secuencial del flujo de trabajo. Los procesos realizados en dicha etapa fueron registrados en una red de verificabilidad con 2 nodos peers para la única organización definida en este estudio de caso.

2. En la segunda prueba se modificó con respecto a la primera prueba, la cantidad de servicios en la etapa del flujo de trabajo encargados de procesar la carga de 1000 datos de temperatura producidos por los 10 sensores. En este caso se consideran cinco servicios para atender la carga generada por los sensores.

3. Finalmente, una tercera prueba se realizó aumentando la carga de trabajo (cantidad de datos de temperatura) a ser procesada en el flujo de trabajo, para este caso se consideraron los mismos 10 sensores y una red de verificabilidad de 2 nodos peers como se evaluaron las pruebas anteriores, pero en este caso se proceso una carga de 10000 datos de temperatura generados entre todos los sensores de forma equitativa.

5.3.5 Métricas

Para este estudio de caso, se calcularon las métricas de tiempo de respuesta, tiempo de despliegue, sobrecarga y tiempo de ejecución definidas en la sección 5.2.5 para cada una de las fases (preparación, despliegue y operación) de la solución MG-CD/CV. Sin embargo, para este estudio de caso dado que implica menos cantidad de servicios en el flujo de trabajo respecto al estudio de caso uno, se realizó el cálculo de los tiempos que tardan cada uno de los registros en la red de verificabilidad con el objetivo de hacer un análisis mas afondo respecto al componente de verificabilidad continua en este estudio de caso.

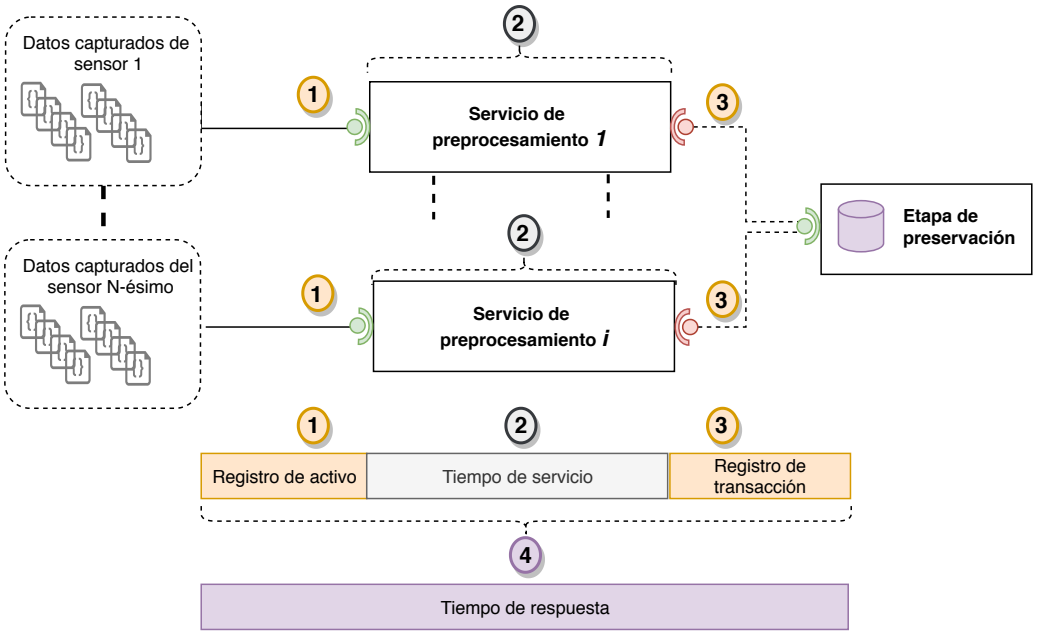


Figura 5.29: Tiempos de ejecución calculados al procesar un lote de datos capturados por un sensor.

En la Fig. 5.29 se muestran los tiempos que fueron capturados para este estudio de caso. En este caso, cuando un sensor captura datos, este conjunto representa un lote de información que debe ser procesado por un servicio (p.e. preprocesamiento). Cuando este lote es asignado por el Gestor CD/CV a un determinado servicio del flujo de trabajo, esta asignación se registra en la red de verificabilidad de bloques encadenados como un activo (dada la lógica de negocio definida en el

módulo de Composer). Posteriormente, el tiempo que tarda cada servicio en procesar dicha carga asignada se denomina el tiempo de servicio. Finalmente, cuando un determinado servicio entrega un resultado, dicho resultado se registra en la red de verificabilidad de bloques encadenados como una transacción. La suma de los tres tiempos mencionados anteriormente, representa el tiempo de respuesta de un determinado servicio.

5.3.6 Resultados y discusión

En esta sección se presentan los resultados obtenidos en cada una de las pruebas definidas en la variación experimental del estudio de caso medioambiental (ver sección 5.3.4). En la prueba uno se describe detenidamente cada uno de los métricas calculadas para este estudio de caso, con el fin de presentar de manera breve los resultados en las pruebas dos y tres.

Para el componente de verificabilidad continua específicamente en la fase de construcción y despliegue de la red de verificabilidad, se tienen prácticamente los mismos tiempos entre las tres pruebas dado que se está construyendo la misma red de verificabilidad (dos nodos peers para una organización) para las tres pruebas ejecutas en este estudio de caso medioambiental.

5.3.6.1. Prueba 1: Carga de 1000 datos procesada por un servicio de Preprocesamiento - Configuración secuencial

A continuación se presentan los tiempos de ejecución de los procesos ejecutados tanto en el componente de entrega continua como en el componente de verificabilidad continua. En la Fig. 5.30 se ilustran los tiempos de ejecución obtenidos en las diferentes fases (preparación, despliegue y operación) del marco de gestión CD/CV para la presente prueba.

En esta prueba, se generaron 1000 datos provenientes de 10 sensores, el tiempo requerido para generar dichos datos se muestra en la anterior Figura con el identificador T1. Los procesos o tareas que corresponden a la fase de preparación requeridos para poder desplegar el sistema CD/CV son identificados en la Fig. 5.30 con los identificadores T16, T17, y T18.

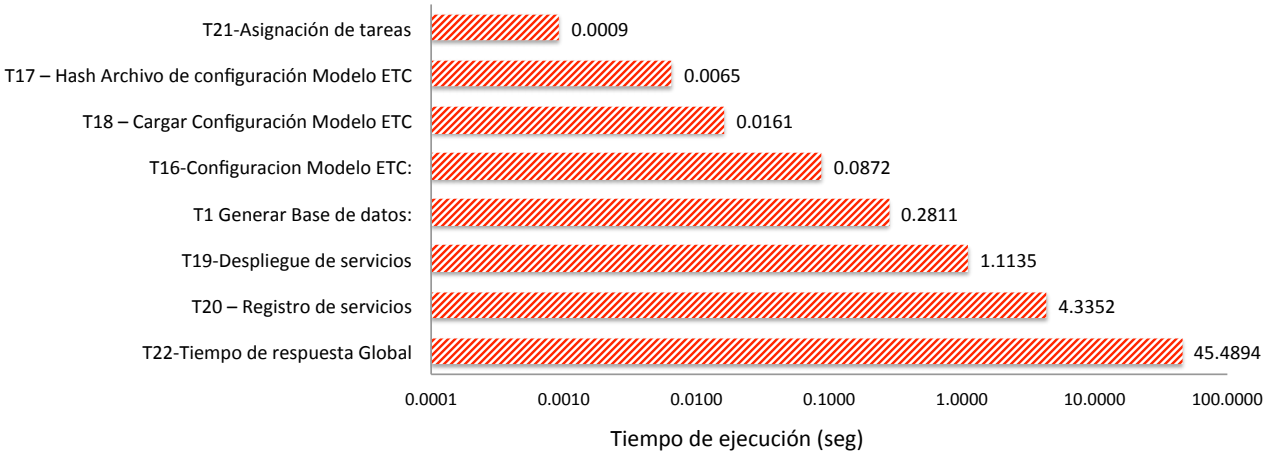
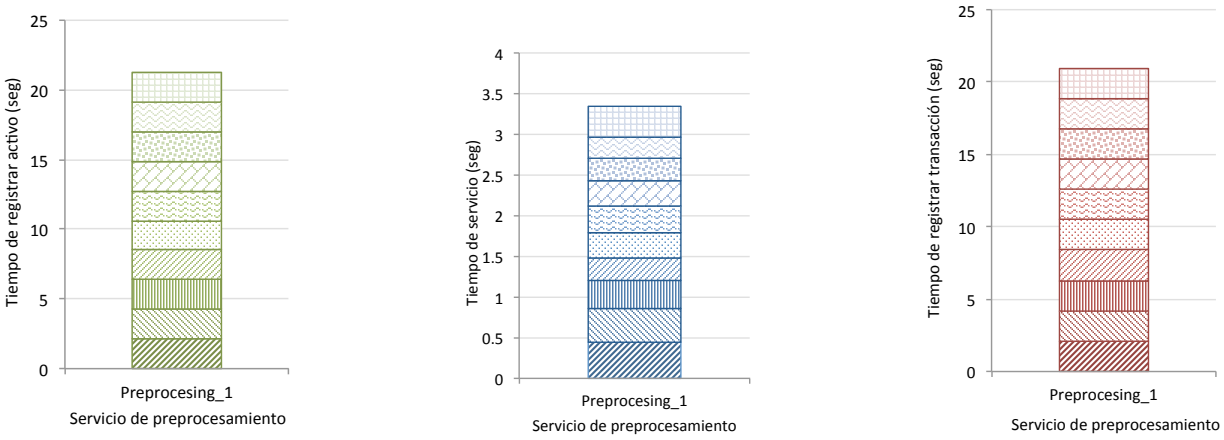


Figura 5.30: Tiempos de ejecución de procesos ejecutados por el Marco de Gestión CD/CV para el componente de entrega continua.

Posteriormente, el tiempo requerido en la fase de despliegue para poner en alta el único servicio de preprocesamiento y la base de datos se muestra con el identificador T19. Una vez que estos dos servicios son desplegados correctamente, el gestor CD/CV los registra como participantes en la red de verificabilidad, este tiempo de registro se muestra con el identificador T20. Dado el resultado obtenido en T20, un participante es registrado en 2.15 seg en promedio en la red de verificabilidad.

Finalmente, una vez que el gestor CD/CV esta en operación, asigna la carga de trabajo a cada uno de los servicios. Esta asignación se identifica en la Fig. 5.30 como el proceso T21. En este punto el gestor esta listo para procesar la carga en el flujo de trabajo. El tiempo que tarda en procesar toda la carga de trabajo a través de cada uno de los servicios hasta que registran los resultados en la base de datos, es el tiempo de respuesta Global identificado con T22.

5.3.6.1.1. Detalles de los tiempos obtenidos en la fase de operación para el componente de verificabilidad continua y entrega continua Como resultado de los tiempos mostrados en la Fig. 5.30, se observa que el Gestor involucra más tiempo en procesar la carga de trabajo (Proceso T22- Tiempo de Respuesta Global). Este tiempo incluye por parte del componente de verificabilidad continua dos tiempos: 1) el *tiempo de registro de cada activo* en la red de verificabilidad, 2) *el tiempo de registro*



(a) (b) (c)

Figura 5.31: Tiempos obtenidos en el modo de operación del gestor MG-CD/CV. (a) Tiempo de registro de cada activo (b) Tiempo de servicio de la etapa de preprocesamiento por cada conjunto de datos a procesar (solicitudes de un sensor). (c) Tiempo de registrar el proceso realizado.

de la transacción emitida por el responsable de procesar dicha carga. Con respecto al componente de entrega continua, el tiempo de respuesta Global incluye el *tiempo de servicio* de cada servicio definido en la etapa del flujo de trabajo de entrega continua.

Para este estudio de caso, un activo representa el conjunto de datos de temperatura capturados por un sensor. En este sentido, dado que son 10 sensores que se utilizaron para realizar la presente prueba, se tiene un total de 10 activos registrados en la red de verificabilidad.

Los tiempos de registro de cada uno de los 10 activos en la red de verificabilidad se muestran en la Fig. 5.31(a). El tiempo total de registrar los activos producidos por los 10 sensores fue de 21.21 seg. En promedio, un activo tarda 2.12 seg en ser registrado en la red de verificabilidad. En este registro, se especifica a que servicio fue asignado el conjunto de datos recabados por un sensor para aplicarle el preprocesamiento y registrarlo en una base de datos en la nube. Este registro de activo se realiza con el fin de establecer un control posterior en caso de fallas, donde un servicio tiene asignada una tarea por realizar pero eventualmente no realiza dicha tarea.

Por otro lado, los tiempos de servicio de la etapa de preprocesamiento de cada uno de los conjuntos de datos recibidos por los 10 sensores se muestran en la Fig. 5.31(b).

Finalmente, una ves el servicio de preprocesamiento realiza su tarea sobre el conjunto de datos de un determinado sensor, emite una transacción en la red de verificabilidad por cada conjunto de datos procesado. En esta transacción se registran meta-datos como por ejemplo una descripción del procedimiento aplicado junto con la firma digital del proceso resultante con el objetivo de establecer control en caso de fallas. Los tiempos de registrar dichas transacciones se muestran en la Fig. 5.31(c).

Los resultados del registro de activos (Fig. 5.31(a)) y de registros de transacciones (Fig. 5.31(c)) que corresponden al componente de verificabilidad continua (CV), indican que aproximadamente el $92.66\,\%$ del tiempo de respuesta global, corresponde al componente de verificabilidad continua (CV) en contraste con el componente de entrega continua (medido por los tiempos de servicio - Fig.5.31(b)) que ocupa cerca del $7.33\,\%$ del tiempo de respuesta global. Este sobrecosto del componente CV respecto al componente CD, se debe a que la configuración secuencial definida para esta prueba, hace que cada uno de los registros a emitir en los bloques encadenados, se realice uno a uno, este proceso se torna lento con respecto al procesamiento de la carga definida para esta prueba en el componente de entrega continua.

Con respecto a los tiempos del componente de verificabilidad continua (tiempos de registro de activo, tiempos de registro de transacción y tiempos de registro de participantes) se ha podido observar durante los resultados hasta el momento presentados, que el tiempo promedio de realizar cada uno de estos registros en los bloques encadenados es aproximadamente de 2 segundos. Esto da un indicio de un comportamiento constante en los tiempos emitidos por cada registro en la red de verificabilidad.

5.3.6.1.2. Tiempos obtenidos en la fase de preparación y despliegue para el componente de verificabilidad continua. Para el componente de verificabilidad continua recordemos que se tienen dos fases: una fase de construcción y despliegue de la red de verificabilidad por parte de Hyperledger Fabric, y por otro lado, la construcción y el despliegue de la red de negocio en la red de verificabilidad por parte de Hyperledger Composer. En las figuras 5.32 y 5.33 se muestran los tiempos de ejecución

de cada uno de los procesos o tareas realizadas en las fases de Fabric y Composer respectivamente.

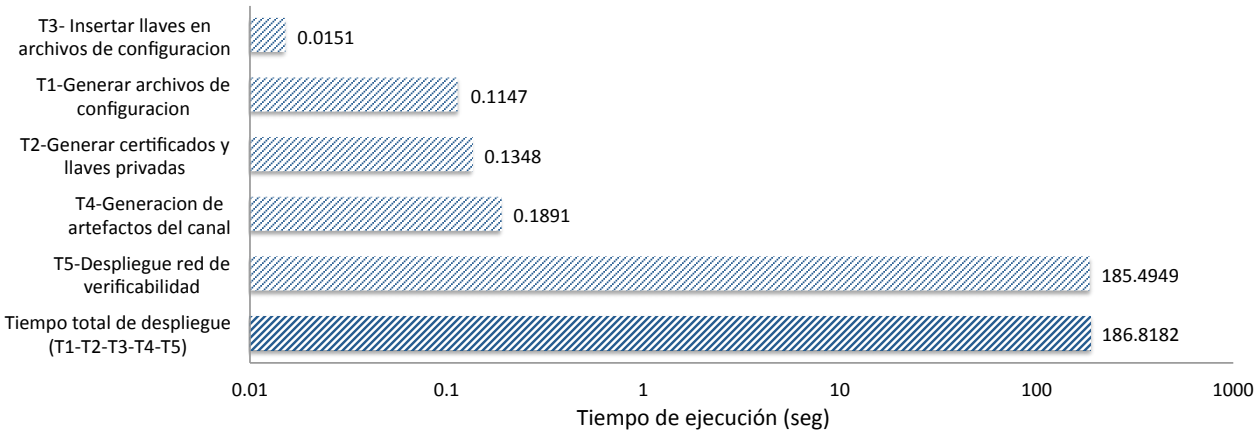


Figura 5.32: Tiempos de ejecución del módulo 1 (Hyperledger Fabric) del componente de verificabilidad continua.

En la fase de preparación se ejecutan los procesos identificados con T1,T2,T3 y T4 en la Fig. 5.32. Con respecto a la tarea T5, el despliegue de los contenedores CV dado el flujo de trabajo descrito en este estudio de caso, corresponde al tiempo de poner en servicios los nodos de la red de verificabilidad con una organización, es decir, un nodo orderer, un nodo entidad certificadora para dicha organización y dos nodos peers. Los tiempos de cada uno de los procesos ejecutados en el segundo módulo (Hyperledger Composer) del componente de verificabilidad continua se muestran en la Fig. 5.33.

Los procesos que son ejecutados en la fase de preparación por parte del Gestor CD/CV respecto al módulo dos de verificabilidad continua (Hyperledger Composer) son los identificados como T6, T7 y T8 en la Fig. 5.33. En la fase de despliegue, dicho módulo ejecuta los procesos identificados con T9,T10,T11,T12,T13,T14 y T15. Los resultados en la Fig. 5.33 indican que el tiempo requerido para desplegar la red de negocios en la red de verificabilidad, es prácticamente el mismo que se requiere para procesar la carga de trabajo de la presente prueba.

Cabe destacar que los tiempos de preparación y despliegue se obtienen una sola vez en la creación del sistema CD/CV. Una vez el sistema se encuentra desplegado es posible aplicarle diferentes cargas de trabajo (modo de operación) durante diferentes periodos de tiempo. Lo cual hace que los 187

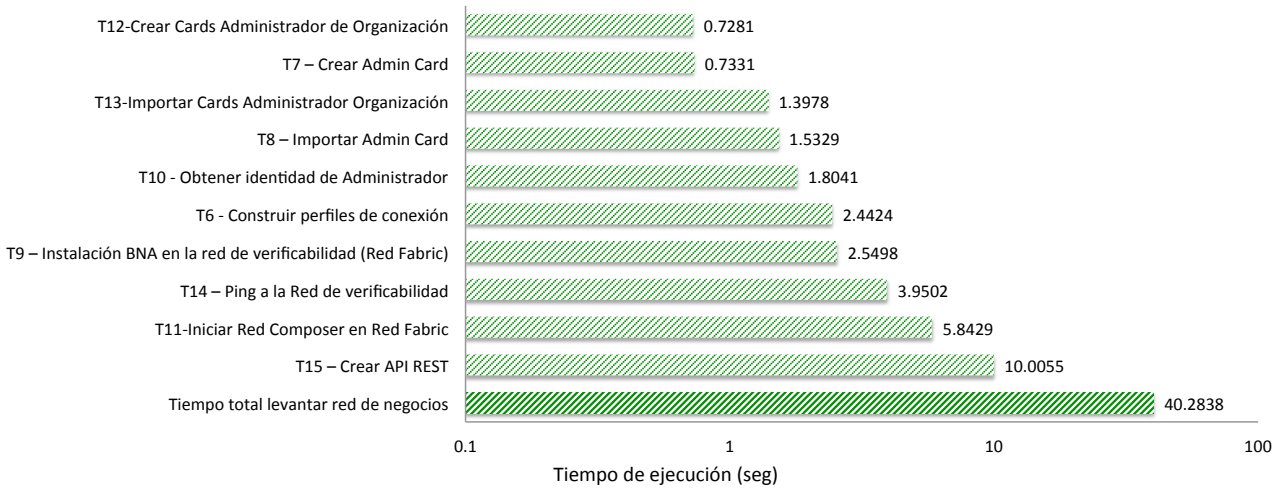


Figura 5.33: Tiempos de ejecución del módulo 2 (Hyperledger Composer) del componente de verificabilidad continua.

seg requeridos para la creación y despliegue de la red de verificabilidad y los 40 seg requeridos para desplegar la red de negocios en la red verificable, sean despreciables respecto a los tiempos generados en la fase de operación.

5.3.6.2. Prueba 2: Carga de 1000 datos con cinco servicios de Preprocesamiento - Configuración Paralelizada

Para la segunda prueba en la cual se aumentan los servicios de preprocesamiento respecto a la prueba uno, se realizaron las mismas mediciones que las ilustradas en la prueba anterior.

El cambio para esta prueba de 1000 datos y cinco servicios con respecto a la prueba número uno, radica en el componente de entrega continua dado que se tienen más servicios que reciben las peticiones de los sensores. En este sentido, en la Fig. 5.34 se ilustran los tiempos de ejecución obtenidos en los procesos de entrega continua del marco de gestión:

Una vez el marco de gestión MG-CD/CV está en operación y es aplicada la carga a los cinco servicios de preprocesamiento en la única etapa del flujo de trabajo, se calcularon los tiempos de servicio del registro de cada activo en la red de verificabilidad con el fin de determinar si había un

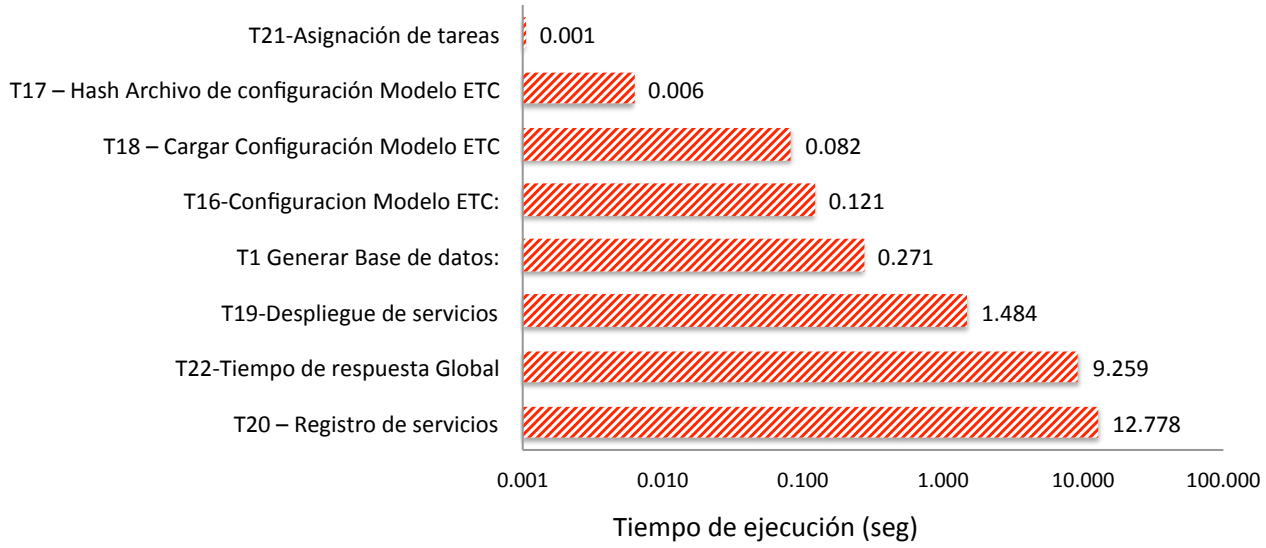


Figura 5.34: Tiempos de ejecución de procesos ejecutados por el Marco de Gestión.

cambio de comportamiento de los tiempos de registro de activos en la red de verificabilidad cuando la cantidad de servicios incrementa.

En este sentido, dado que son 10 sensores que se utilizaron para realizar la presente prueba, se tienen un total de 10 activos registrados en la red de verificabilidad. Los tiempos de registro de cada uno de los 10 activos se muestran en la Fig. 5.35.

Para esta prueba dado que se tienen 10 sensores y cinco servicios que reciben las peticiones de dichos sensores, mediante un algoritmo de balanceo de carga como Two Choices [48], se distribuyó la carga total de 1000 datos entre los cinco servicios de preprocesamiento.

Como resultado de aplicar el algoritmo mencionado, se obtuvo una carga balanceada (ver Fig. 5.36) en la cual cada servicio de preprocesamiento recibía las peticiones de dos sensores con un total de 200 solicitudes entre los dos sensores asignados. Dado que los cinco servicios se ejecutan de forma paralelizada, cada servicio registra simultáneamente un activo, lo cual permitió que los tiempos de registro de los activos se redujeran en un $79.34\,\%$ respecto a la prueba uno en la cual se tiene un solo servicio de preprocesamiento. Este tiempo de registro de los activos esta dado por el proceso que tarda más tiempo de los 5 servicios paralelizados, el cuál fue de 4.38 seg. El promedio

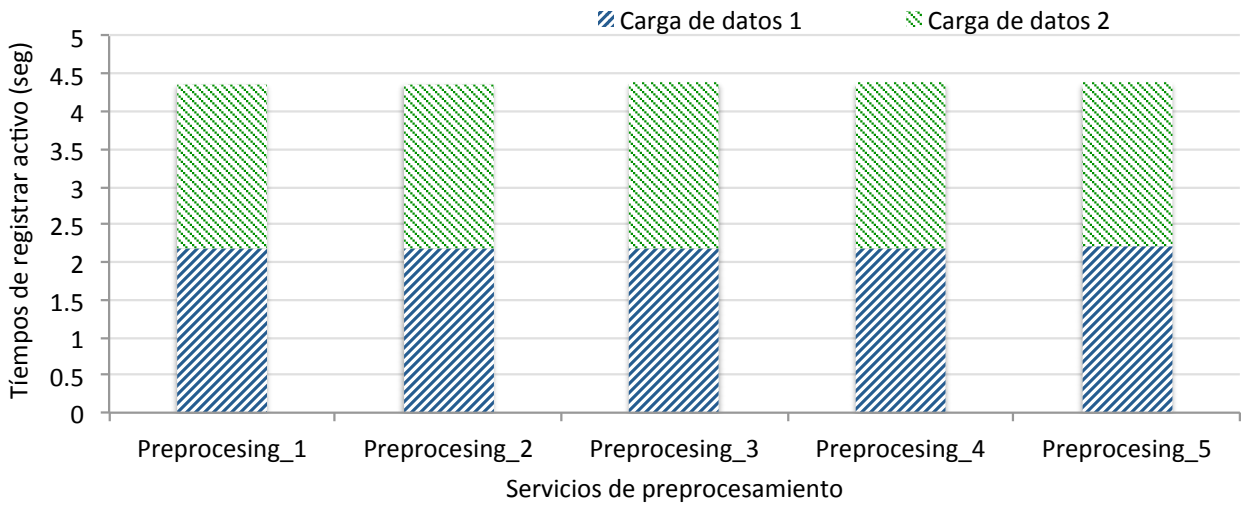


Figura 5.35: Tiempo de registro de 10 activos en la red de verificabilidad con 5 servicios por etapa.

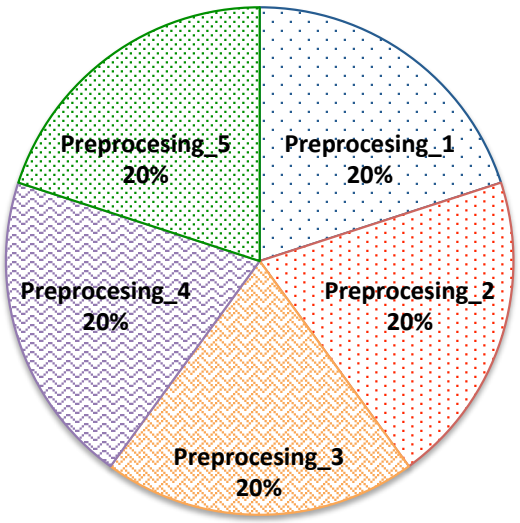


Figura 5.36: Distribución de la carga generada por los 10 sensores entre los cinco servicios de preprocesamiento.

de registro de activo en la red de verificabilidad independientemente del servicio al que perteneciera o hilo de ejecución en el cual se ejecutara, fue de 2.1873 seg.

Por otro lado, se obtuvieron los tiempos de servicio de cada uno de los cinco servicios de la etapa de preprocesamiento encargados de procesar los conjuntos de datos de los 10 sensores. Estos tiempos de servicio se muestran en la Fig. 5.37.

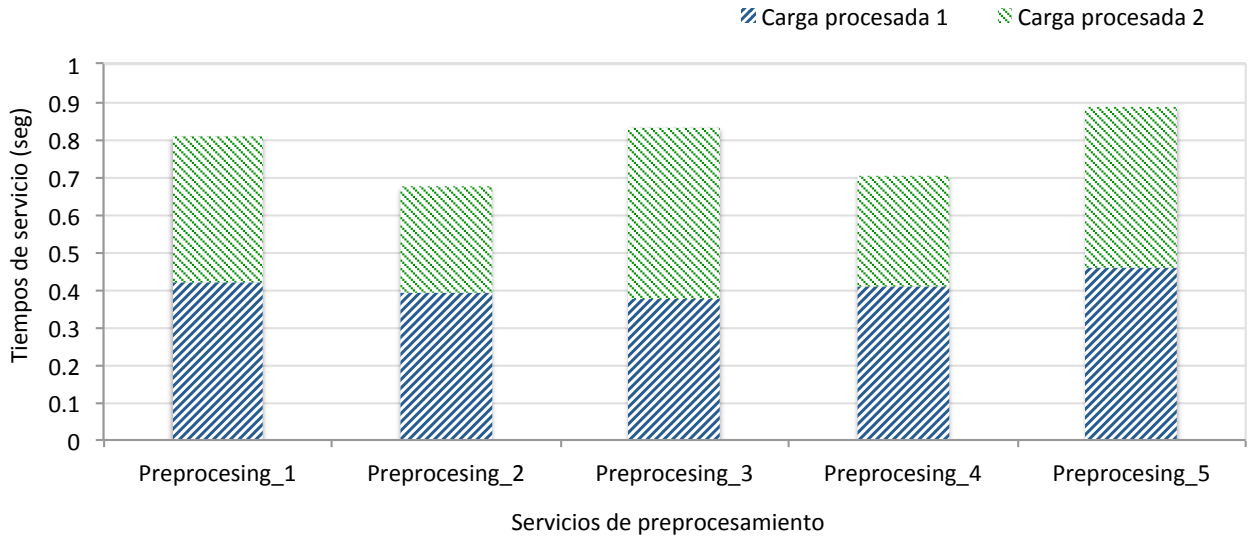


Figura 5.37: Tiempo de servicio en la etapa de preprocesamiento sobre una carga 1000 datos con cinco servicios de preprocesamiento.

Finalmente, una vez los servicios de preprocesamiento realizan su tarea sobre el conjunto de datos asignado, emite una transacción en la red de verificabilidad por dicho conjunto de datos procesado. Los tiempos de registrar dichas transacciones se muestran en la Fig. 5.38.

Los resultados en está prueba indican que al tener un esquema paralelizado en el procesamiento de los datos del flujo de trabajo, es posible reducir el sobrecosto generado por el componente CV dado que los registros a la red de verificabilidad son emitidos de forma paralelizada por cada servicio que recibe la carga de trabajo.

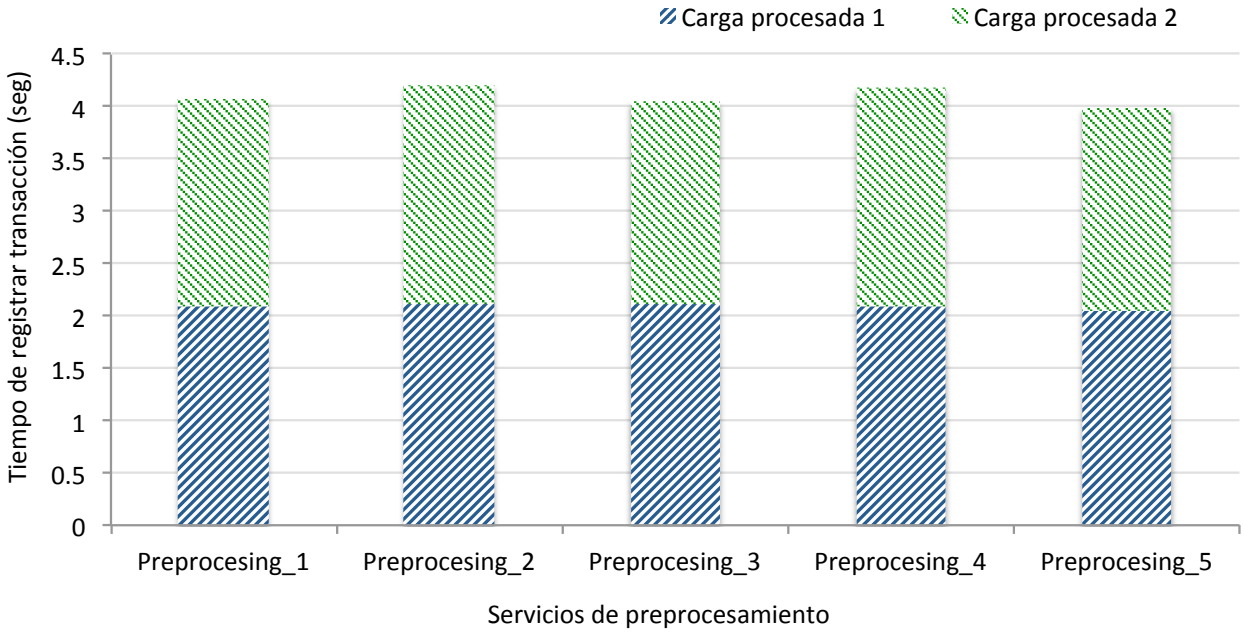


Figura 5.38: Tiempos de registro de transacciones en la etapa de preprocesamiento sobre una carga de 1000 datos.

5.3.6.3. Prueba 3: Carga de 10000 datos con cinco servicios de Preprocesamiento - Configuración Paralelizada

En esta prueba se conservaron los parámetros definidos en la prueba numero dos recientemente descrita, a excepción de la cantidad de solicitudes enviadas por los 10 sensores. En este caso, fueron evaluados en el gestor MG-CD/CV un total de 10000 solicitudes entre los 10 sensores. Esta prueba fue llevada a cabo con el objetivo de saber el comportamiento del sobrecosto de las transacciones generadas por el componente CV cuando la carga es mayor a las evaluadas en las anteriores pruebas. En este sentido, se realizaron las mismas mediciones que las ilustradas en la prueba número uno.

Para esta prueba de 10000 datos y cinco servicios se examinó los tiempos de registro en la red de verificabilidad tanto de activos, participantes y transacciones como los tiempos de servicio en la etapa de preprocesamiento, en función del incremento de las solicitudes de cada sensor, que para esta prueba seria de 1000 solicitudes por cada sensor.

La diferencia de esta prueba con respecto a las dos anteriores, radica en el componente de entrega continua dado que se tiene una mayor carga de trabajo a ser procesada. En este sentido, en la Fig. 5.39 se ilustran los tiempos de ejecución obtenidos en los procesos de entrega continua del Marco de Gestión.

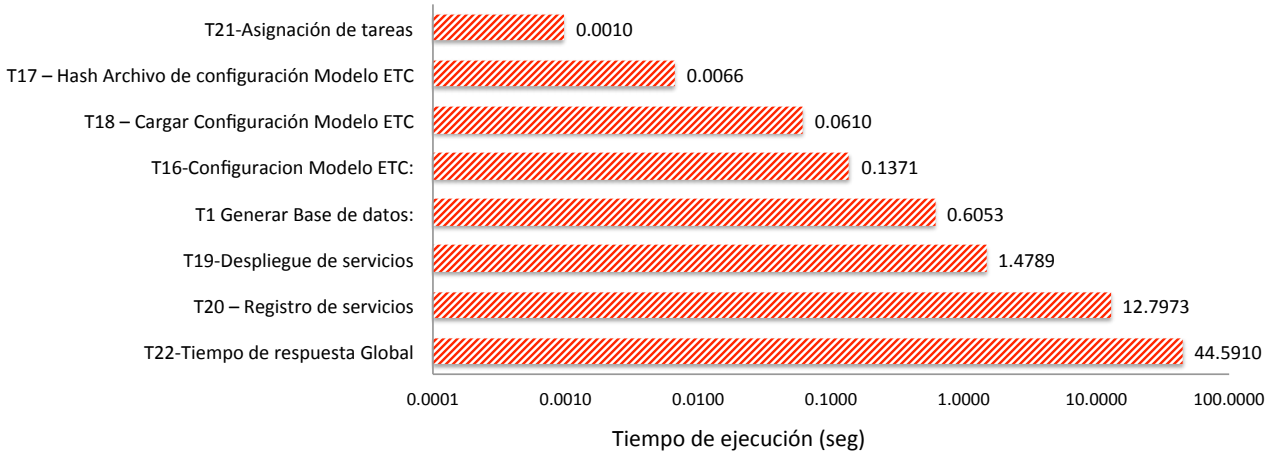


Figura 5.39: Tiempos de ejecución de procesos ejecutados por el Marco de Gestión CD/CV con una carga de 10000 solicitudes.

Una vez el marco de gestión MG-CD/CV está en operación y es aplicada la carga de 10000 datos a los cinco servicios de preprocesamiento en la única etapa del flujo de trabajo, se calcularon los tiempos de servicio del registro de cada activo en la red de verificabilidad. En este contexto, dado que son 10 sensores que se utilizaron para realizar la presente prueba, se tienen un total de 10 activos registrados en la red de verificabilidad. Los tiempos de registro de cada uno de los 10 activos se muestran en la Fig. 5.40.

Al igual que en la prueba número dos (sección 5.3.6.2), para esta prueba se distribuyó la carga total de 10000 datos entre los cinco servicios de preprocesamiento mediante un algoritmo de balanceo de carga como *Two Choices*. Así mismo, a cada servicio le fue asignado dos sensores con un total de 2000 solicitudes entre los dos sensores asignados.

Dado que en esta prueba los cinco servicios también se ejecutan de forma paralelizada, cada servicio registra simultáneamente un activo, tal cómo sucede en la prueba número dos. Sin embargo,

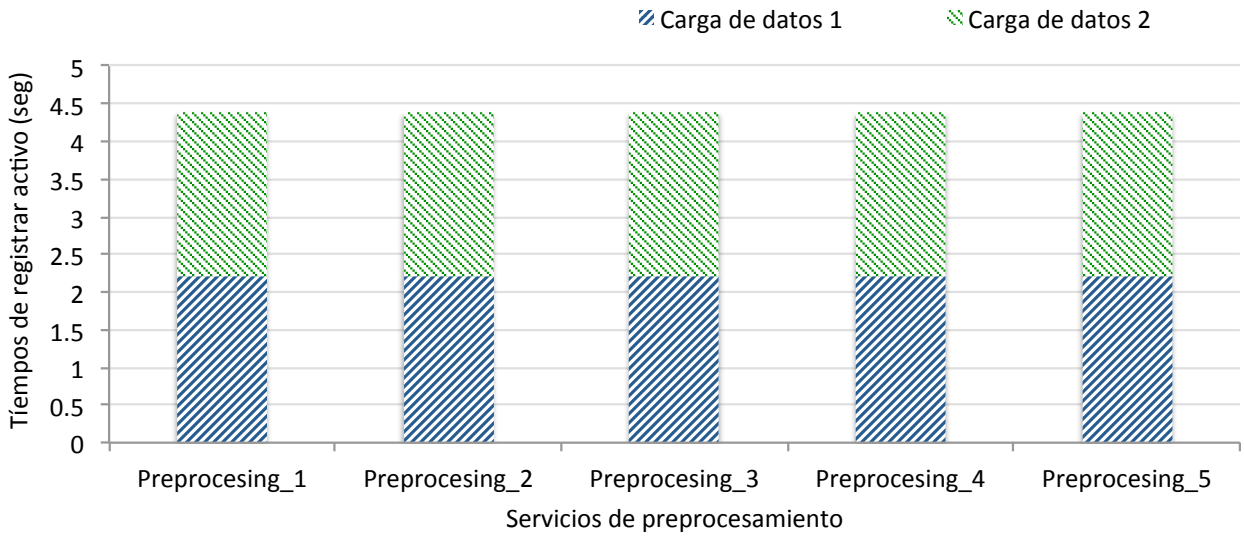


Figura 5.40: Tiempo de registro de 10 activos (carga de 100000) en la red de verificabilidad con 5 servicios por etapa.

a pesar que la carga de trabajo para esta prueba se aumento 10 veces más con respecto a la prueba número dos, dado el diseño en el módulo de *hashing* para la generación de transacciones en nuestra propuesta de investigación, los tiempos en los registros de activos no se ve afectado. Esto se debe a que la firma digital que se calcula en el módulo de *hashing* se realiza sobre la meta-data del directorio que contiene todo el conjunto de datos a procesar (para este escenario son los datos de temperatura recabados por un sensor). Este diseño permite que a mayor cantidad de carga a procesar, si bien los tiempos de servicio en las etapas del flujo de trabajo van a incrementar, los tiempos de registro de activos o transacciones prácticamente permanecerán constantes y provocarán un sobre-costo menor al usar el gestor MG-CD/CV en la medida que crece la carga de trabajo, tal como ocurrió en el escenario de movilidad para volúmenes de 100 datos.

Para corroborar lo anteriormente descrito se muestra en la Fig. 5.41 los tiempos de registro de transacciones emitidas al obtener los resultados de cada uno de los 5 servicios para cada una de las cargas de datos asignadas.

Con respecto a los tiempos de servicio, como es de esperarse, se incrementaron dado el incremento

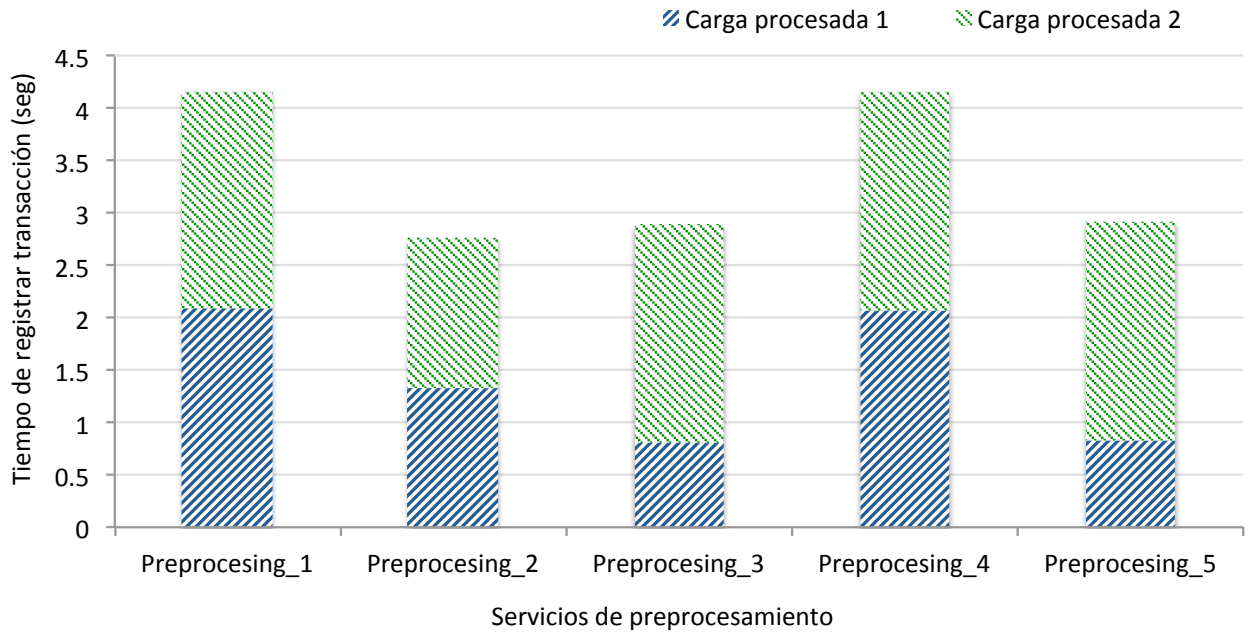


Figura 5.41: Tiempos de registro de transacciones en la etapa de preprocesamiento sobre una carga 1000 datos.

en las solicitudes emitidas por cada sensor. Estos tiempos de servicio se muestran en la Fig. 5.42.

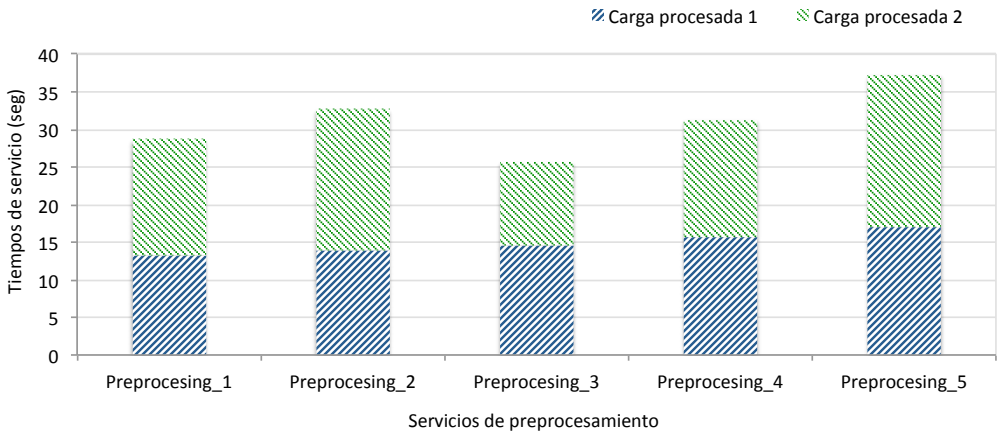


Figura 5.42: Tiempo de servicio en la etapa de preprocesamiento sobre una carga 1000 datos con cinco servicios de preprocesamiento.

5.3.6.4. Análisis comparativo de las pruebas realizadas

En esta sección se presenta un análisis comparativo de las tres pruebas realizadas y descritas anteriormente. El primer análisis corresponde a la sobrecarga que generan los dos tipos de registros que se llevan a cabo en el componente de verificabilidad continua (registro de activo y registro de transacción) en cada una de las pruebas realizadas.

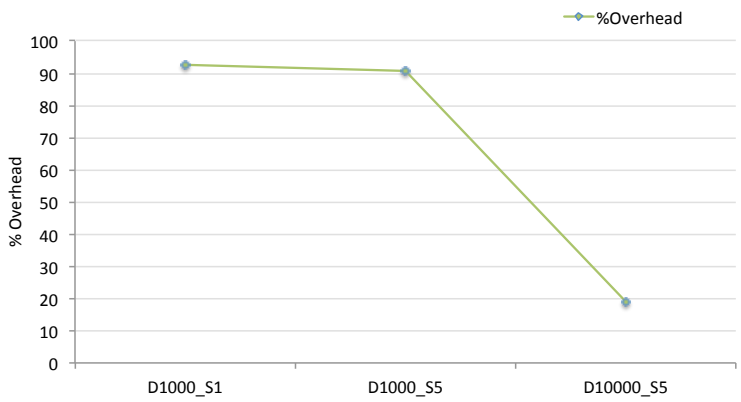


Figura 5.43: Porcentaje de sobrecarga (Overhead) de las tres pruebas realizadas.

Los resultados en la Fig. 5.43 indican efectivamente que al aumentar la carga de trabajo a procesar en el flujo de trabajo con verificación continua, menor es el sobrecosto del Gestor CD/CV, dado que los tiempos de registro en la red de verificabilidad, permanecen constantes independientemente del tipo de registro que se realice (de activo, de transacción o de participante) en la red verificable. Esto se debe principalmente al diseño a grano grueso del Gestor CD/CV para generar una firma digital de un lote de información en lugar de generar una firma digital por cada dato de cada sensor.

La Fig. 5.44 muestra de manera clara, cómo disminuye notablemente el porcentaje de sobrecosto del Componente CV respecto al componente CD cuando la carga de trabajo es aumentada. Adicionalmente, se puede observar que entre las configuraciones de la prueba 1 ($D_{1000}\_S_1$) y la prueba 2 ($D_{1000}\_S_5$) dado el esquema paralelizado que se implementa en la prueba 2 (sección 5.3.6.2), se reducen los tiempos de servicio correspondientes al componente de entrega continua, sin embargo, como los tiempos en los registros a la red de verificabilidad permanecen constantes

para este estudio de caso, no hay una reducción notoria del sobrecosto del Componte CV respecto al sobrecosto obtenido en la prueba uno.

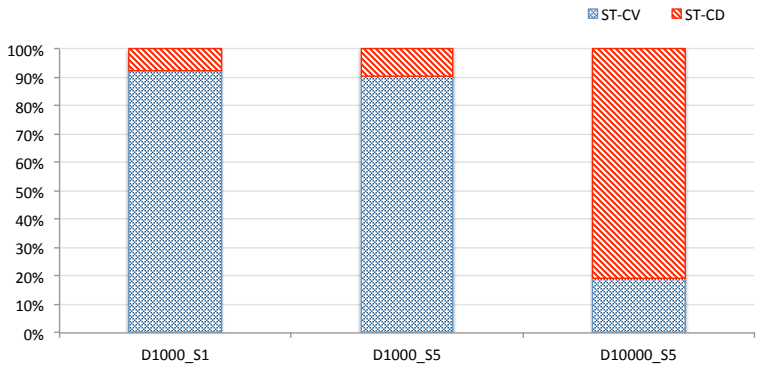


Figura 5.44: Tiempo de servicio (ST) de cada uno de los componentes del marco de gestión (entrega continua - CD y verificabilidad continua - CV) en las tres pruebas ejecutadas.

5.4 Resumen

En este capítulo fue descrita la metodología de evaluación de la solución propuesta en el Capítulo 4. Esta evaluación fue conducida por dos estudios de caso: uno basado en información de movilidad de usuarios y el segundo basado en datos medioambientales capturados por sensores IoT. Los estudios de caso revelaron la factibilidad de aplicar el Marco de Gestión propuesto a los procesos de integración entre flujos de trabajo y redes de verificabilidad. Adicionalmente, se observó como es posible reducir e incluso eliminar la sobrecarga de los procesos de verificabilidad continua en comparación con soluciones del estado del arte.

6

Conclusiones y trabajo futuro

En este capítulo se presentan las conclusiones que resultaron de la evaluación experimental realizada en el Capítulo 5. Adicionalmente, se mencionan las principales contribuciones del presente trabajo de Tesis, así como los retos que se han presentado al conducir esta investigación y las limitaciones/acotamientos de la solución propuesta en este trabajo Tesis. Finalmente, se describen áreas de oportunidad para llevar a cabo trabajo a futuro, con el fin de dar continuidad a este trabajo de investigación.

En este trabajo de investigación se hace frente a los problemas de proveer verificabilidad continuada a las transacciones realizadas en la entrega continua de datos que ocurre en un flujo de trabajo. Un modelo ETC aplicado por un motor de ejecución de aplicaciones permite la incorporación automática y transparente de una red de verificabilidad (basada en bloques encadenados) a un flujo de trabajo convencional.

La implementación de un modelo de verificabilidad continuada impone restricciones de eficiencia que en este trabajo de Tesis se solventa con mecanismos de manejo eficiente del procesamiento

de datos, lo cual reduce el costo de operación de la red de verificabilidad y su impacto sobre el rendimiento del flujo de trabajo.

Los modelos de manejo ETC para verificabilidad continua, el motor de ejecución de aplicaciones para entrega continua de datos y los mecanismos de procesamiento eficiente integraron un marco de gestión de entrega continua y verificabilidad continua llamado *MG-CD/CV* (compuesto por sus siglas en inglés).

La propuesta MG-CD/CV (o **M**arco de **G**estión de entrega continua y verificabilidad continua-**C**ontinuous**D**elivery/**C**ontinuous**V**erifiability) fue correctamente construida, desplegada y evaluada al procesar datos en dos escenarios IoT completamente diferentes, donde se tiene la necesidad de verificar las etapas de procesamiento realizadas a datos provenientes de dispositivos móviles y sensores.

El primer estudio de caso, enfocado en análisis de datos de movilidad de usuarios, se procesa una cantidad de datos conocida con antelación. En tal escenario, se mostró que MG-CD/CV no sólo puede reducir la sobrecarga generada por los procesos de verificabilidad sino que en algunos casos se podría observar una mejora en el rendimiento de la solución completa. Lo anterior se observó en una comparación directa con dos soluciones del estado del arte que producen entrega continua, pero que no hacen uso de la red de verificabilidad.

El segundo estudio, enfocado en el procesamiento de datos medioambientales capturados por sensores, se procesa información que se asume que arriba de forma constante a la etapa de procesamiento. Por tanto, el volumen final de datos a procesar es desconocido pero el volumen de procesamiento por unidad de tiempo puede ser determinado por el gestor. Se observó que la sobrecarga producida por los registros realizados por MG-CD/CV se puede gestionar en forma eficiente, tanto con el uso de patrones de distribución y balanceo de carga, como consolidando la cantidad de datos a procesar para cada registro a realizar en la red de verificabilidad.

Tomando en cuenta lo antes mencionado y el diseño de la solución propuesta en este trabajo de Tesis, se asume que MG-CD/CV puede ser aplicado a cualquier flujo de trabajo científico cuyo fin sea

el procesamiento de datos provenientes de IoT, incluso de cualquier dato cuyo formato es entendido por las aplicaciones o procesos definidos en las etapas del flujo de trabajo. Lo anterior es conseguido con ciertas limitaciones y restricciones que serán enunciadas y descritas en la sección 6.2.

Como resultado de la experimentación realizada en el estudio de caso de movilidad y de datos medioambientales, se describen a continuación las conclusiones obtenidas en cada uno de los siguientes apartados:

MG-CD/CV gestiona los registros en la red de verificabilidad con estrategias de grano grueso y fino.

La evaluación experimental mostró que la estrategia de grano grueso, aplicada a lotes de información, es adecuada para escenarios donde la generación de datos es constante (sensores). Como resultado de aplicar la estrategia de grano grueso al conjunto de tareas a ser transformados (declaradas en el ETC), la generación de firmas digitales se realiza sobre lote de tareas a procesar proveniente de una entidad. Por ejemplo, un conjunto de trayectorias de un usuario dado en el caso del escenario de movilidad o un conjunto de datos producidos por un sensor en el caso medioambiental.

La estrategia de grano fino, en cambio, es aplicada a cada proceso de transformación del ETC (por cada archivo transformado) para generar una firma digital por cada dato proveniente de dicha entidad.

La primera estrategia reduce los costos de la verificabilidad continua pero registra menos procesos en la red de verificabilidad. La segunda estrategia ingresa más información a la red pero incrementa los costos de los registros de información, lo cual hace poco viable su implementación en escenarios del mundo real.

Es importante notar que, para escenarios de dispersión geográfica, los costos de registro (registro de activo, registro de participante y registro de transacciones) realizados en la red de verificabilidad deberían considerar la latencia proporcional a la separación geográfica de los nodos peers que cada organización ha desplegado en la red de verificabilidad.

La evaluación experimental reveló que la cantidad de nodos peers en el componente CV

desplegado para cada organización impacta significativamente en el rendimiento del componente CD. Un número de dos nodos peers por organización resultaba suficiente para realizar el registro de transacciones con disponibilidad de servicio sin imponer costos significativos al componente CD.

El aumento en la cantidad de servicios por etapa (en función de la cantidad de cores de la infraestructura disponible) en el componente CD, permite realizar en una forma eficiente las tareas de procesamiento en un flujo de trabajo de entrega continua.

Sin embargo, este incremento produce también un incremento en la gestión de los participantes del componente CV. Esto se debe a que cada uno de los servicios que es desplegado por el MG-CD/CV es considerado como un participante en la red de verificabilidad con el fin de establecer cuales son los responsables del procesamiento de cada activo (conjunto de datos). Esto resulta en que cada servicio registra las transacciones de cada tarea que realiza, lo cual incrementa el número de registros por unidad de tiempo. Por este motivo, es importante considerar un balance entre la cantidad de servicios que se definen para cada etapa y la cantidad de nodos peers para la red de verificabilidad. Como evidencia de lo anteriormente expuesto, se pudo observar en la evaluación experimental que al desplegar en el componente CD tantos servicios como cores en un servidor y manteniendo el número de peers acotado al mínimo de disponibilidad (por ejemplo 10 servicios y 2 peers) representa la configuración que ofrece el mejor rendimiento.

En la comparativa de MG-CD/CV con una solución del Estado del Arte (DagOn), la evaluación experimental reveló que DagOn no permite el procesamiento altamente concurrente de cargas de trabajo (no fue posible procesar cargas de 100 usuarios en el escenario de movilidad) mientras que MG-CD/CV no mostró esta limitación. DagOn sólo es competitivo cuando se procesan tareas cortas y no se le limita el lanzamiento de hilos de ejecución para las etapas (e.g. el procesamiento de 10 usuarios). La mejor configuración de DagOn produjo un 20 % de mejora en el rendimiento en comparación con MG-CD/CV pero sabiendo que DagOn no utiliza la red de verificabilidad. En escenarios en los cuales MG-CD/CV y DagOn usaban la misma cantidad de recursos, la sobrecarga no sólo se eliminaba sino que incluso MG-CD/CV conseguía producir un mejor rendimiento que DagOn

(82.88 % en promedio para las configuraciones con 10 y 12 hilos con cargas de 1 y 10 usuarios).

Dado el crecimiento de los Dispositivos IoT y el aumento exponencial del volumen de datos generados en este entorno, una solución como MG-CD/CV resulta crucial para dar verificabilidad continua al procesamiento de dichos datos.

La incorporación del modelo ETC en la solución MG-CD/CV permitió definir un esquema de entrega continua de datos para el manejo eficiente de grandes volúmenes de datos IoT sensibles mediante paralelismo basado en tareas. Adicionalmente, cómo resultado de la integración de dicho esquema con una red de verificabilidad como son los bloques encadenados, se obtuvieron mecanismos de verificabilidad continua de transacciones realizadas con datos IoT sensibles en flujos de trabajo.

Por tanto, con base en la experimentación realizada y las conclusiones dadas previamente, se concluye que la hipótesis de este trabajo de investigación es aceptada, comprobando que un gestor intermediario entre un flujo de trabajo para el procesamiento de datos IoT basado en un modelo de extracción, transformación y carga (ETC) permite realizar verificabilidad continua a las transacciones realizadas en flujos de trabajo, mientras que el uso de patrones de paralelismo encapsulados en contenedores virtuales permite reducir el impacto de dicha verificabilidad en el rendimiento de dichos flujos de trabajo.

6.1 Contribuciones

Como resultado del trabajo de investigación se obtuvo un Marco de Gestión de flujos de trabajo basado en bloques encadenados para la verificación continua de datos en escenarios del Internet de las Cosas. A partir de este trabajo se aporta lo siguiente:

1. Un gestor de flujos de trabajo basado en un concepto que en este trabajo de investigación se denomina *entrega continua con verificación continua (CD/CV)*.

2. Un modelo de flujo de trabajo y modelos de negocio con bloques encadenados basado en ETC.

3. Una solución de software que implementa el gestor CD/CV basado en el modelo ETC.

6.2 Retos y Limitaciones

Uno de los principales desafíos a los que se hizo frente en la realización de este trabajo de investigación fue definir una abstracción para la implementación de un flujo de trabajo que fuese verificable mediante una red como la de bloques encadenados, el cual se pudo solventar mediante un modelo ETC y que en la literatura no se suele utilizar en la gestión de flujos de trabajo.

Otro desafío fue intentar generalizar la utilización del gestor para que el usuario final pudiera desplegar un CD/CV para sus flujos de trabajo. Existen limitación y requerimientos para conseguir tal generalidad. Para establecer la generalidad de la creación de soluciones mediante MG-CD/CV se establecen el seguimiento y obligado cumplimento de pautas que se listan a continuación:

1. Etapas del flujo de trabajo: MG-CD/CV requiere que cada etapa del flujo de trabajo tenga la forma de un servicio o aplicativo ejecutable. Esta restricción se debe a que el diseño del Gestor CD/CV está pensado inicialmente, para entregar como respuesta de una etapa i a una etapa $i+1$, la salida del servidor donde se encuentra expuesto el servicio (respuesta del API REST) ó los logs de un aplicativo ejecutable.

2. Proceso declarativo: MG-CD/CV asume que el usuario declara el ETC de cada aplicativo.

3. Disponibilidad de la fuente de datos: MG-CD/CV requiere la ubicación de una fuente de datos (conjunto de tareas a procesar), a partir de la cual se inicie el proceso de entrega continua.

4. Intercambio de datos: MG-CD/CV requiere que los resultados intercambiados entre las etapas sean empaquetados en un formato estándar como JSON para garantizar la entrega continua. Se asume que los datos colocados en el archivo de intercambio son conocidos por las etapas que realizan el intercambio.

5. Red de verificabilidad: MG-CD/CV ya incorpora el gestor de bloques encadenados, si el usuario requiere cambiar el gestor de bloques encadenados tendría que realizar ajustes al motor de despliegue del MG-CD/CV (por ejemplo utilizar Etherum u otro mecanismo para usar criptomonedas en las transacciones).

6.3 Trabajo Futuro

Con el objetivo de dar continuidad a este trabajo de investigación, se plantea como trabajo futuro los siguientes aspectos:

La incorporación de un esquema de alta disponibilidad para MG-CD/CV. Se requiere de un mecanismo de consenso en los sistemas a gran escala para que los cambios a una determinada entidad sean percibidos en todo el sistema distribuido de una manera consistente. En este trabajo de investigación, un cambio a una entidad puede estar determinada por el fallo en alguno de los Contenedores Virtuales CD/CV. Un fallo de un servicio CD podría interrumpir el proceso de entrega continua, y un fallo en los servicios CV (nodos peers) podría interrumpir o inhabilitar los registros a la red de verificabilidad. En este sentido, se requiere de un mecanismo de consenso para identificar qué componente ha fallado y poder re-establecer el servicio. Uno de los algoritmos para alcanzar consenso en este tipo de sistemas es el algoritmo de Paxos, propuesto originalmente por Lamport [39].

Definir y evaluar un escenario de múltiples organizaciones distribuido geográficamente.

Dado el alcance del proyecto y la infraestructura disponible para la realización del mismo, no fue posible definir un escenario en el cual se encuentre desacoplado la red de verificabilidad con el flujo de trabajo. En este sentido, es de interés evaluar el desempeño de MG-CD/CV en un escenario donde cada etapa de cada organización se encuentre desplegada en un lugar geográfico completamente diferente, al igual que los nodos peers que pertenecen a cada organización y que conforman la red

verificable. La definición y evaluación de este escenario distribuido, implica la incorporación de un esquema de intercambio de llaves privadas de forma segura, entre los administradores que generan las llaves y los nuevos participantes que se van a incorporar a la red de verificabilidad.

Método de función hash para la construcción del registro transaccional.

Dado el diseño de MG-CD/CV para generar el valor *hash* del proceso entrante y saliente de cada etapa del flujo de trabajo, se requiere que cada proceso resultante sea preservado en el sistema de archivos de la infraestructura propuesta. Esta preservación de los resultados implica un costo de lectura y escritura que afecta directamente los tiempos de respuesta que percibe el usuario. Este costo podría ser reducido si el intercambio de información entre las etapas se hace directamente desde memoria. Sin embargo, es necesario estudiar como podría garantizarse la verificabilidad continua bajo este enfoque.

A

Anexos

A.1 Implementación de Prototipo MG-CD/CV

El modelo declarativo de interconexión ETC que incorpora el Gestor CD/CV o Gestor Global se implementó a través de un formato de archivo JSON. El Gestor Global (Gestor de construcción y Gestor de operación) fueron implementados en el lenguaje de Python. Con respecto al Gestor de Operación, este incluye un módulo de *Hashing* en el cual se construye una transacción. El algoritmo utilizado para realizar el calculo del valor Hash de los diferentes contenidos de los datos a procesar en los flujos de trabajo de los diferentes escenarios fue el SHA256, el cual es uno de los más utilizados en la literatura.

A.1.1 Orquestadores

En esta sección se describen los Orquestadores tanto del componente de entrega continua (CD) como del componente de verificación continua (CV).

A.1.1.1. Orquestador CD

En cuanto a los Orquestadores se implementó el Orquestador CD en un módulo de Python. Este Orquestador genera un archivo de configuración en formato YAML, el cual es un formato requerido por los Lanzadores para realizar el despliegue dado que se incorpora la tecnología de Docker para ello.

A.1.1.2. Orquestador CV

Para los Orquestadores CV se incorporó la tecnología de Hyperledger Fabric, la cual tiene unos archivos binarios ejecutables que permiten generar los artefactos necesarios para configurar y desplegar una red de verificabilidad de bloques encadenados. Estos binarios y los artefactos que generan se describen a continuación:

1. *cryptogen*: Esta herramienta permite generar todo el material criptográfico (certificados, llaves privadas) de cada uno de los nodos (peer, orderer, autoridades certificadoras o ca) que

pertenecen a la red de verificabilidad para poder establecer una comunicación segura entre ellos.

2. *configtxgen*: Esta herramienta permite configurar el canal a través del cual los nodos de la red de verificabilidad se comunicarán. Inicialmente, genera el bloque inicial de los bloques encadenados (o Blockchain). Este bloque se conoce como el *bloque génesis* el cual se utiliza para el arranque del nodo orderer encargado del mecanismo de consenso. Al momento de redactar este documento de tesis, la documentación[1] referente a esta herramienta solo permitía la generación del bloque génesis, creación del canal de comunicación y configuración de los peers. Sin embargo, esperan mejorar dicha herramienta a futuro para generar nuevas configuraciones de canales, así como para reconfigurar los canales existentes.

3. *configtxlator*: La herramienta configtxlator[2] de Hyperledger, permite la reconfiguración independiente de los SDK. La configuración del canal se almacena como una transacción en los bloques de configuración de un canal y se puede manipular directamente. Sin embargo, en el momento de escribir este artículo, ningún SDK admite de forma nativa la manipulación de la configuración directamente, por lo que la herramienta *configtxlator* está diseñada para proporcionar una API con la que los consumidores de cualquier SDK puedan interactuar para ayudar con las actualizaciones de configuración.

   En este trabajo de tesis, la herramienta fue aplicada simplemente para realizar comprobaciones básicas de seguridad para asegurarse de que estén disponibles las versiones apropiadas de imágenes/binarios de Hyperledger Fabric. La configuración de los canales en cada una de las pruebas realizadas solo variaba en función de la cantidad de nodos peers que harían parte por cada organización según el estudio de caso a implementar.

4. *peer*: Este binario permite el despliegue de la configuración del canal en cada uno de los nodos peer que pertenecerán a dicho canal para poderse comunicar entre sí en la red de verificabilidad.

[1] https://hyperledger-fabric.readthedocs.io/en/release-1.0/configtxgen.html
[2] https://hyperledger-fabric.readthedocs.io/en/release-1.0/configtxlator.html

A.1.2 Coreógrafos

En esta sección se describen los Coreógrafos tanto del componente de entrega continua (CD) como del componente de verificación continua (CV).

A.1.2.1. Coreógrafo CD

Los componentes implementados para obtener un Coreógrafo CD fue un Despachador que incorpora un Balanceador de Carga implementado en un módulo de Python. El Balanceador de carga seleccionado fue el Algoritmo Two Choices [48], el cual consiste básicamente en generar dos números aleatorios entre la cantidad de servicios disponibles para recibir la carga de trabajo (datos a procesar), y aquel servicio entre los dos seleccionados que tenga una menor cantidad de carga asignada, será el seleccionado para procesar la carga actual.

Adicionalmente, el Coreógrafo CD implementado distribuye y procesa la carga a través de un patrón Maestro/Esclavo. La cantidad de Esclavos que procesarán la carga, está definido por la cantidad de Contenedores Virtuales CD que defina un usuario para cada etapa del flujo de trabajo.

A.1.2.2. Coreógrafo CV

La verificación continua en esta investigación fue diseñada e implementada a través de una red de verificabilidad blockchain construida con la tecnología de Hyperledger Fabric (HF) e Hyperledger Composer (HC) para modelar la lógica de negocio general.

A.1.2.2.1. Red de verificabilidad - Hyperledger Fabric El framework de HF permite la implementación de Blockchain mediante la tecnología de contenedores que se encargan de albergar contratos inteligentes llamados "chaincode" que comprenden la lógica de la aplicación. Por otro lado, HC permite que los propietarios y desarrolladores de negocios creen rápidamente y de manera fácil los contratos y las aplicaciones de Blockchain para resolver problemas de negocio.

Los elementos necesarios para diseñar, construir y desplegar una red de verificabilidad con el framework de Hyperledger, son considerados en el modelo de interconexión ETC.

El diseño de la red Blockchain en Hyperledger Fabric (HF) incluye un conjunto de componentes tales como nodos, Autoridades de Certificación (CAs), mecanismo de consenso, chaincode, ledger, canales entre otros. Los cuales se describen a continuación:

1. **Nodos**: Cada nodo de la red de verificabilidad se implementa a través de un contenedor virtual mediante la tecnología de Docker, los nodos en Hyperledger pueden ser de 3 tipos:

   a) **Orderer (Nor):** Dado que son pocas organizaciones, se definió un solo nodo orderer para toda la red Blockchain en los dos estudios de caso.

   b) **Autoridades de Certificación (CAs):** Se definió una entidad certificadora por organización. Por ejemplo, para el estudio de caso de movilidad, se tuvieron un total de tres entidades certificadoras con los siguientes nombres de dominio, ca.org1.example.com para la Org1, ca.org2.example.com para la Org2 y ca.org3.example.com para la Org3.

   c) **Peer (Npe):** La cantidad de nodos peer por cada organización fue definida por parámetro del Gestor. Por ejemplo, para el estudio de caso de movilidad, en la configuración que obtuvo mejores resultados, se definió que cada organización dispone de dos nodos peers para conformar la red Blockchain. En este sentido, dado que son tres organizaciones se tiene un total de seis peers (peer0.org1.example.com, peer1.org1.example.com, peer0.org2.example.com, peer1.org2.example.com, peer1.org3.example.com y peer1.org3.example.com)

2. **Canal:** Para los dos estudios de caso, fue creado un canal denominado 'mychannel', al cual cada uno de los nodos peer de cada organización se han unido a el.

A.1.2.2.2. Red de negocios - Hyperledger Composer

1. **Perfil de conexión** Por medio de un archivo de configuración se describe cada uno de los componentes de la red HF (nodos peers, orderers, autoridades de certificación, organizaciones, canal, etc) en el cual se indican los certificados de cada uno de los componentes, a fin de

que los usuarios administradores de cada organización puedan conectarse a HF. Para cada organización o administrador se tiene un perfil de conexión diferente.

2. **Usuarios** Para cada una de las organizaciones se ha configurado un usuario Administrador a fin de que cada organización tenga un responsable por las transacciones realizadas en dicha organización.

   Por ejemplo, para el estudio de caso uno, para la Org1 se configuro el usuario Admin@org1.example.com, para la Org2 el usuario Admin@org2.example.com y para la Org3 el usuario Admin@org3.example.com. Cada usuario administrador posee un certificado que es la parte pública de la identidad y una clave privada que se usa para firmar transacciones como esta identidad (mediante firmas digitales). Para las CAs se han configurado tanto para las Org1, Org2 y Org3, un usuario predeterminado. Este usuario cuenta con una identificación de inscripción de administrador y un secreto de inscripción. Sin embargo, este usuario no tiene los permisos para implementar la red de negocios definida por Hyperledger Composer en la red de HF.

3. **Business Network Cards - BNC** Estas tarjetas permiten a los administradores de las organizaciones desplegar la red de negocio en HF. Se creo una BNC por cada administrador de una organización. Para ello, se requiere el perfil de conexión, el certificado y la clave privada del administrador al cual se le va crear la tarjeta. Por ejemplo, para el estudio de caso uno, dado que son tres organizaciones, se crearon tres BNCs, PeerAdmin@byfn-network-org1.card para el administrador de la Org1, PeerAdmin@byfn-network-org2.card para el administrador de la Org2 y PeerAdmin@byfn-network-org3.card para el administrador de la Org3. Estas tarjetas se deben importar al wallet de Composer para poderlas utilizar.

4. **Política de aprobación** Una política de aprobación define las reglas en torno a las cuales las organizaciones deben respaldar las transacciones antes de que puedan comprometerse con la Blockchain. Para el estudio de caso uno, la política que se definió establece que las tres organizaciones definidas deben aprobar las transacciones en la red comercial antes de que

puedan comprometerse con la Blockchain. En caso tal de que la Org1, Org2 u Org3 no respalden las transacciones, la red comercial rechazará la transacción. Esta política se definió en el formato JSON utilizado por el SDK de Hyperledger Fabric Node.js que es diferente al formato de CLI de Hyperledger Fabric.

A.1.3 Encapsulación del Gestor CD/CV

En un contenedor de Docker se instaló el entorno de desarrollo de Composer, con el fin de ejecutar los comandos de Hyperledger Composer necesarios para interactuar con la red de Hyperledger Fabric. Entre las funciones que se pueden realizar con los comandos de Hyperledger Composer están: 1) Crear Business Network Cards para los administradores; 2) Agregar participantes y generar identidades; 3) Desplegar red comercial (BNA) en la red de Hyperledger Fabric; 4) Emitir transacciones a la Blockchain, entre otras.

Sobre la base de este contenedor creado con la herramienta de Hyperledger Composer fue desarrollado el Gestor CD/CV y posteriormente una nueva imagen de contenedor fue creada. Como resultado, se obtuvo un Gestor CD/CV portable y fácil de desplegar en diferentes infraestructuras que tengan instalada la plataforma de contenedores de Docker.

Configuración de la red de verificabilidad para el estudio de caso de movilidad Para el estudio de caso de movilidad se consideraron 3 organizaciones, los nombres de dominio utilizados para cada una de ellas son org1.example.com,
org2.example.com y org3.example.com respectivamente. El proveedor de servicios de membresía (MSP) para Org1 se llama Org1MSP, para la Org2 se llama Org2MSP y Org3MSP para la Org3.

Componentes de la red A modo de ejemplo, para la mejor configuración obtenida en el estudio de caso uno, la cual incluye una red de verificabilidad con dos nodos peers por cada organización, se muestra como se conforma la red de Hyperledger Fabric (ver Figura A.1):

1. **Dos nodos** pares(peer) para Org1, llamados
   peer0.org1.example.com y peer1.org1.example.com.

   a) El puerto de solicitud para peer0 es 7051.

   b) El puerto del hub de evento para peer0 es 7053.

   c) El puerto de solicitud para peer1 es 8051.

   d) El puerto del hub de evento para peer1 es 8053.

2. Una CA única (**Autoridad de certificación**) para Org1, llamada ca.org1.example.com. El puerto CA es 7054.

3. **Dos nodos** pares (peer) para Org2, llamados peer0.org2.example.com y peer1.org2.example.com.

   a) El puerto de solicitud para peer0 es 9051.

   b) El puerto del hub de evento para peer0 es 9053.

   c) El puerto de solicitud para peer1 es 10051.

   d) El puerto del hub de evento para peer1 es 10053.

4. Una CA única (Autoridad de certificación) para Org2, llamada ca.org2.example.com. El puerto CA es 8054.

5. **Dos nodos** pares (peer) para Org3, llamados peer0.org3.example.com y peer1.org3.example.com.

   a) El puerto de solicitud para peer0 es 11051.

   b) El puerto del hub de evento para peer0 es 11053.

   c) El puerto de solicitud para peer1 es 12051.

   d) El puerto del hub de evento para peer1 es 12053.

6. Una CA única (Autoridad de certificación) para Org3, llamada ca.org3.example.com. El puerto CA es 9054.

7. Un único **nodo orderer**, llamado orderer.example.com. El puerto de orderer es 7050.

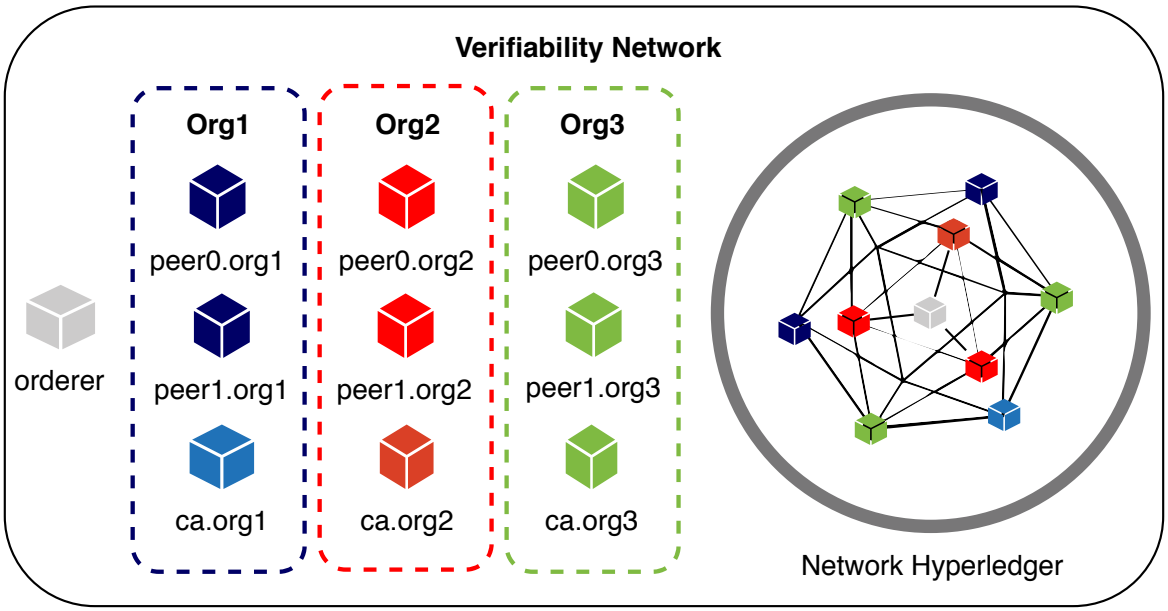


Figura A.1: Implementación de la red de verificabilidad en la mejor configuración de la solución MG-CD/CV para el estudio de caso de movilidad.

Estos **componentes se ejecutan dentro de contenedores Docker**. Al ejecutar Hyperledger Composer dentro de un contenedor Docker, los nombres anteriores (por ejemplo, peer0.org1.example.com) se pueden usar para interactuar con la red de Hyperledger Fabric.

Cada uno de los componentes que conforman la red, están protegidos mediante el protocolo TLS (Transport Layer Security) para cifrar las comunicaciones.

Para poder conectarse a cada uno de los componentes de la red, es necesario obtener los certificados de la autoridad de certificación (CA) para cada uno.

A.2 Implementación Estudio de Caso: Movilidad de Usuarios

El prototipo (marco de gestión) implementado permite desplegar un flujo de trabajo verificable para datos de mobilidad (figura A.2). Este prototipo incorpora el modelo declarativo de interconexión ETC descrito anteriormente. Por medio de este modelo declarativo, el Gestor es capaz de realizar una

entrega continua de los datos a través de las etapas del flujo de trabajo y realizar una verificación continua mediante una red de verificabilidad de las transacciones que se producen en el flujo de trabajo y cuya implementación se describió anteriormente.

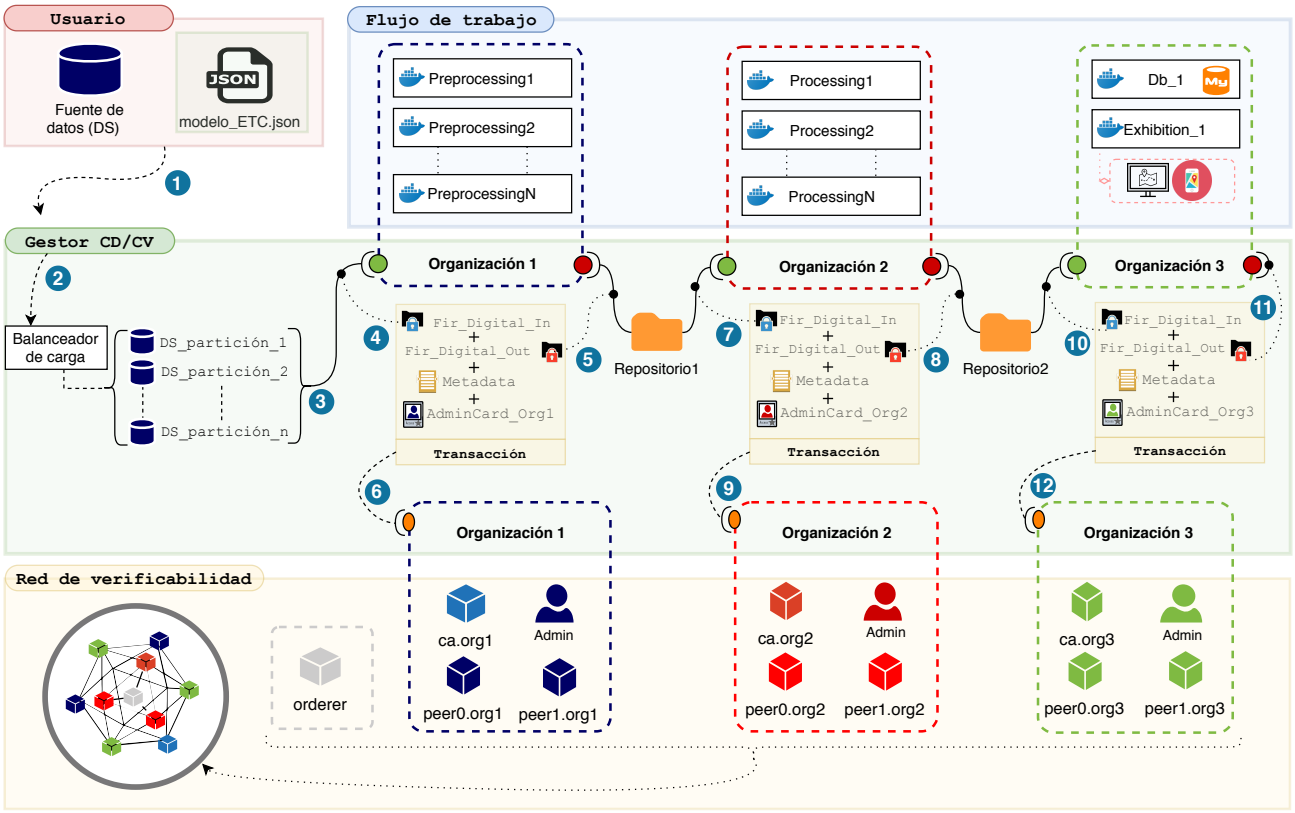


Figura A.2: Implementación Estudio de Caso de Movilidad.

La implementación de estos componentes de entrega continua y verificación continua que brinda el Marco de Gestión, se describen a continuación:

A.2.0.0.3. Implementación Entrega Continua - Flujo de trabajo En la presente sección se describe el desarrollo seguido para la implementación del proceso de análisis de extracción de información que se muestran en la Figura A.3.

La fase de preprocesamiento de datos fue implementada utilizando el lenguaje de programación R, mientras que las fases de procesamiento y análisis fueron evaluadas utilizando el lenguaje de programación Python.

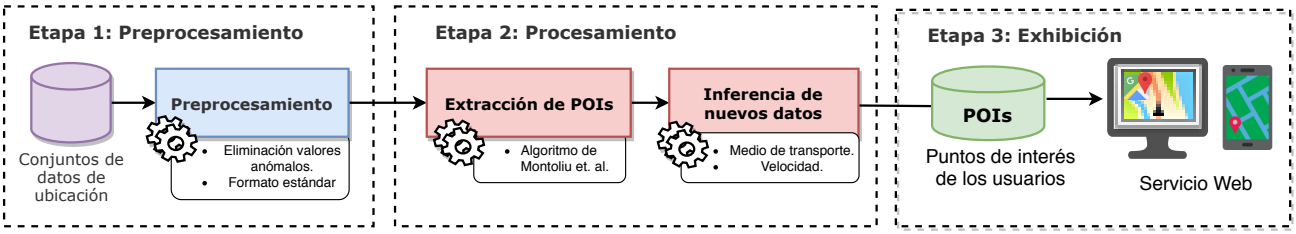


Figura A.3: Estudio de Caso de Movilidad.

Preprocesamiento - Detección y eliminación de datos atípicos En esta fase se implementó un mecanismo de detección de valores atípicos propuesta por Tamizh et. al. [10], la cual utiliza el estimador estadístico conocido como $Z - Score$, el cual es comúnmente utilizado para la detección de valores atípicos, como para la normalización de conjuntos de datos a partir de su media aritmética y desviación estándar. La detección de valores atípicos con esta medida se realiza de la siguiente manera:

- Centrar cada uno de los valores sustrayendo la media de la población.
- Reducir los valores dividiéndolos entre la desviación estándar de la población.
- Los valores obtenidos serán independientes de la media, por lo que tendrán una misma dispersión
- Aquellos valores que estén fuera de un umbral, típicamente entre $(-3, 3)$, serán considerados valores atípicos.

Lo anterior, se modela matemáticamente en la Ecuación A.1, donde μ es la media aritmética de la muestra, y σ es la desviación estándar.

$$z = \frac{x - \mu}{\sigma} \tag{A.1}$$

El mismo principio es utilizado para la eliminación de valores atípicos, en los datos de localización. En donde se calcula el valor z para cada muestra del conjunto de datos, y son descartadas las que se encuentren fuera del umbral. Primeramente, se calculan las que estén fuera del umbral de la latitud,

y posteriormente, se hace lo mismo con la longitud. Prácticamente, lo que hace este método es remover todos aquellos valores que se encuentren fuera de un rectángulo marcado (Figura A.4).

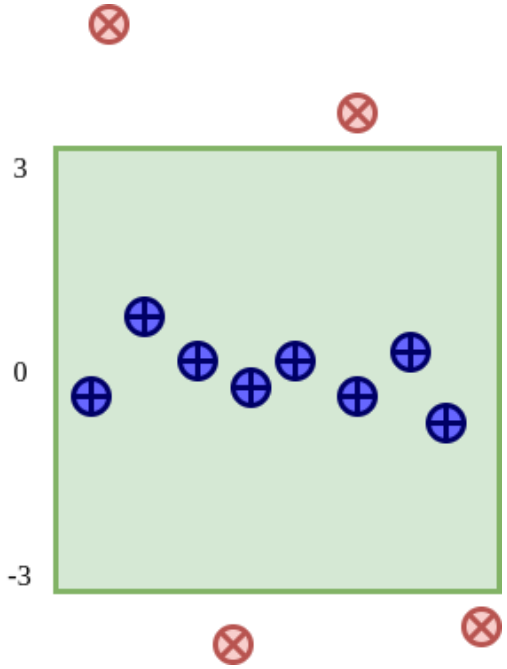
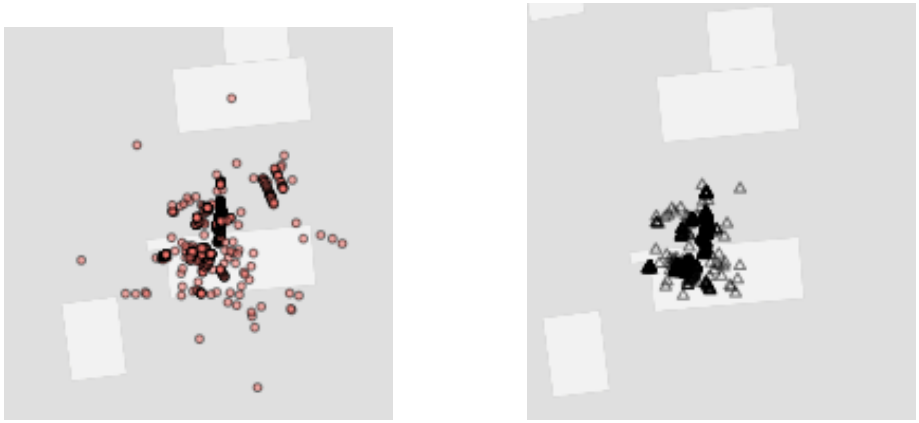


Figura A.4: Eliminación de puntos atípicos utilizando la medida z. Valores fuera del umbral $(-3, 3)$ son considerados valores atípicos, y se eliminan.

En la Figura A.5 se muestra un ejemplo del funcionamiento del mecanismo para remover valores atípicos, mostrando un mapa con los datos en crudo (Figura A.5(a)) y otro con los datos sin valores atípicos (Figura A.5(b)).



(a) (b)

Figura A.5: Prueba del mecanismo para remover valores atípicos utilizando valores GPS reales. a) Puntos en crudo. b) Puntos sin valores atípicos.

Procesamiento: En esta etapa se aplican los siguientes procesos:

a) Detección y extracción de puntos de interés (POIs): Como se mencionó anteriormente, para la extracción de los puntos de interés se utilizó el algoritmo propuesto por Montoliu et. al. [53], el cual a partir de una lista de puntos de ubicación del usuario extrae los puntos de interés sintonizando el algoritmo en base a tres parámetros: la distancia máxima entre puntos, y el tiempo mínimo y máximo que debió de haber pasado el usuario en el punto para considerarlo un punto de interés. En el Algoritmo 1 se muestra el procedimiento seguido para extraer los POIs, el cuál básicamente verifica que dos puntos GPS están dentro de un umbral espacial y temporal.

b) Inferencia del medio de transporte: Una vez limpiados los datos, son extraídos los POIs para cada usuario por día. Con la finalidad de poder obtener información extra se realizó una inferencia del medio de transporte que utilizó un usuario para ir de un POI a a un POI b. Dado un conjunto de POIs este se denota como $LSP_{u,t} = \{SP_1, SP_2, SP_3, ..., SP_n\}$ donde u es un usuario y t es la componente temporal utilizada para la extracción de los puntos (día, semana, quincena, etc.), y donde cada SP_i tiene un conjunto de puntos $P = \{p_1, p_2, ..., p_n\}$ que lo conforman, el medio de transporte entre dos SP_a y SP_b se realiza de la siguiente manera:

- Se toma la tupla de valores $< latitud, longitud, fecha >$ del último punto p_n que es parte del SP_a. Este punto ahora será conocido como $origen$.

Algorithm 1 Algoritmo de extracción de POIs

Require: $l_p = (p_1, ..., p_N)$: List of location points.
T_{min}, T_{max}: Time thresholds.
D_{max}: Distance threshold.
Ensure: l_{sp}: List of resulting stay points.

1: $i \leftarrow 1$;
2: $c \leftarrow 0$;
3: $l_{sp} \leftarrow \emptyset$;
4: **while** $i < N$ **do**
5: $j \leftarrow i + 1$;
6: $c \leftarrow 0$;
7: **while** $j < N$ **do**
8: $t \leftarrow TimeDifference(p_j, p_{j-1})$;
9: **if** $t > T_{max}$ **then**
10: $i \leftarrow j$
11: $c \leftarrow 0$;
12: $break$
13: **end if**
14: $d \leftarrow SpaceDistance(p_i, p_j)$;
15: **if** $d > D_{max}$ **then**
16: $t \leftarrow TimeDifference(p_i, p_{j-1})$;
17: **if** $t > T_{min}$ **then**
18: $[lat, long] \leftarrow EstimateCentroid(p_k | k \in [i, j-1])$;
19: $radius \leftarrow EstimateRadius(p_k | k \in [i, j-1])$;
20: $T^{start} \leftarrow p_i.T$;
21: $T^{end} \leftarrow p_{j-1}.T$;
22: $sp \leftarrow [lat, long, T^{start}, T^{end}, radius]$;
23: $l_{sp} \leftarrow l_{sp} \cup sp$;
24: $c \leftarrow 1$;
25: $break$
26: **end if**
27: $i \leftarrow j$;
28: $break$;
29: **end if**
30: $j \leftarrow j + 1$
31: **end while**
32: **if** $c! = 1$ **then**
33: $i \leftarrow i + 1$
34: $break$
35: **end if**
36: **end while**

- Se toma la tupla de valores $< latitud, longitud, fecha >$ del último punto p_n que es parte del SP_b. Este punto ahora será conocido como $destino$.

- Se recorre cada punto que esté entre el conjunto de puntos desde el $origen$ hasta el $destino$ de la siguiente manera que $\{p|p \in LP \wedge p \in$ (origen,destino)$\}$, donde LP es la lista total de puntos.

  - Se calculá la distancia (d) entre p_i y p_i, así como la diferencia de tiempo (Δt).
  - Posteriormente se calcula la velocidad (v) de la forma: $v = \frac{d}{\Delta t}$.
  - De una tabla de velocidades promedio de cada medio de transporte, se verifica con cual de estas tiene una menor diferencia, y la trayectoria es etiqueta. La tabla de medios de transporte está conformada por n renglones donde cada uno corresponde a un medio de transporte, y cada uno tiene la forma $< clave, valor >$, donde la $clave$ es el medio de transporte y el $valor$ es el número de ocurrencias.

- Finalmente, el medio de transporte con el $valor$ más alto, será tomado como el medio que uso el usuario para ir del SP_a al SP_b.

Para la identificación del medio de transporte conforme a la velocidad, se utilizaron las medidas de velocidad promedio para diferentes medios de transporte en China recolectados por Zong et. al. [79], complementado con los datos recogidos en el conjunto de datos de *GeoLife* [78], en la cuál se especifican las distancias y tiempos recorridos por cada medio de transporte, por lo que solo se calculo la velocidad de cada medio. En la Tabla A.1 se presentan la velocidad promedio de cada medio de transporte utilizado para etiquetar los datos.

Tabla A.1: Velocidades promedio en m/s de los diferentes medios de transporte.

Medio de transporte	Velocidad promedio (m/s)
Walking	0.515
Bike	0.7472222
Bus	3.7361111
Car or Taxi	3.8277778
Train	13.51
Airplane	172.1458333

Desarrollo de un servicio en la nube para la extracción de POIs. El flujo descrito anteriormente fue desarrollado como un servicio en la nube mediante una arquitectura de servicios.

En general existen tres servicios principales los cuales son el servicio de preprocesamiento, procesamiento y exhibición. El servicio de preprocesamiento se encarga de transformar los archivos de la base de datos Geolife en un archivo JSON eliminando *outliers* y unificando los formatos. El servicio de procesamiento se encarga de la estimación de POIs y la inferencia del medio de transporte. Finalmente, el servicio de exhibición se encarga exhibir mediante un mapa (API de Google Maps) los datos extraídos en los servicios anteriores. Cada servicio desarrollado tiene entradas y salidas por medio de un API REST.

Para mejorar el rendimiento del preprocesamiento y el procesamiento de grandes volúmenes de datos, por medio de paralelismo de tareas (ver Fig. A.6) se ejecutan N contenedores de preprocesamiento y N contenedores de procesamiento, los cuáles reciben los archivos a procesar por medio de un *dispatcher* que cuenta con un balanceador de carga de trabajo.

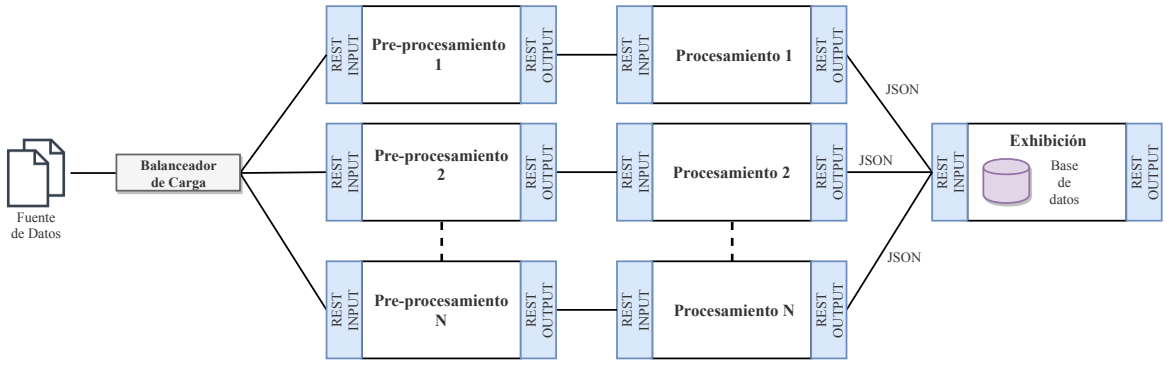


Figura A.6: Patrón arquitectural propuesto para el análisis de datos GPS.

Bibliografía

[1] Al-Sakran, H. O. (2015). Intelligent traffic information system based on integration of internet of things and agent technology. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 6(2):37–43.

[2] Alphand, O., Amoretti, M., Claeys, T., Dall'Asta, S., Duda, A., Ferrari, G., Rousseau, F., Tourancheau, B., Veltri, L., and Zanichelli, F. (2018). Iotchain: A blockchain security architecture for the internet of things. In *Wireless Communications and Networking Conference (WCNC), 2018 IEEE*, pages 1–6. IEEE.

[3] Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., et al. (2018). Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, page 30. ACM.

[4] Babuji, Y. N., Chard, K., Foster, I. T., Katz, D. S., Wilde, M., Woodard, A., and Wozniak, J. M. (2018). Parsl: Scalable parallel scripting in python. In *IWSG*.

[5] Bahga, A. and Madisetti, V. K. (2016). Blockchain platform for industrial internet of things. *Journal of Software Engineering and Applications*, 9(10):533.

[6] Bass, L., Weber, I., and Zhu, L. (2015). *DevOps: A software architect's perspective*. Addison-Wesley Professional.

[7] Boettiger, C. (2015). An introduction to docker for reproducible research. *ACM SIGOPS Operating Systems Review*, 49(1):71–79.

[8] Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

[9] Cachin, C. (2016). Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*, volume 310.

[10] Chelvam, T. T. and Samundeeswari, V. V. (2012). Processing outliers in gps trajectory data set. In *Proceedings of International Conference on Electronics*, volume 96.

[11] Choosri, N., Park, Y., Grudpan, S., Chuarjedton, P., and Ongvisesphaiboon, A. (2015). Iot-rfid testbed for supporting traffic light control. *International Journal of Information and Electronics Engineering*, 5(2):102.

[12] Christidis, K. and Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *Ieee Access*, 4:2292–2303.

[13] Churches, D., Gombas, G., Harrison, A., Maassen, J., Robinson, C., Shields, M., Taylor, I., and Wang, I. (2006). Programming scientific and distributed workflow with triana services. *Concurrency and Computation: Practice and Experience*, 18(10):1021–1037.

[14] Conoscenti, M., Vetro, A., and De Martin, J. C. (2016). Blockchain for the internet of things: A systematic literature review. In *Computer Systems and Applications (AICCSA), 2016 IEEE/ACS 13th International Conference of*, pages 1–6. IEEE.

[15] Cosmina, I. (2017). Spring microservices with spring cloud. In *Pivotal Certified Professional Spring Developer Exam*, pages 435–459. Springer.

[16] Deelman, E., Gannon, D., Shields, M., and Taylor, I. (2009). Workflows and e-science: An overview of workflow system features and capabilities. *Future generation computer systems*, 25(5):528–540.

[17] Deelman, E., Peterka, T., Altintas, I., Carothers, C. D., van Dam, K. K., Moreland, K., Parashar, M., Ramakrishnan, L., Taufer, M., and Vetter, J. (2018). The future of scientific workflows. *The International Journal of High Performance Computing Applications*, 32(1):159–175.

[18] Deelman, E., Singh, G., Su, M.-H., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Vahi, K., Berriman, G. B., Good, J., et al. (2005). Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming*, 13(3):219–237.

[19] Dhillon, V., Metcalf, D., and Hooper, M. (2017). The hyperledger project. In *Blockchain enabled applications*, pages 139–149. Springer.

[20] Dorri, A., Kanhere, S. S., and Jurdak, R. (2016). Blockchain in internet of things: challenges and solutions. *arXiv preprint arXiv:1608.05187*.

[21] Dorri, A., Kanhere, S. S., and Jurdak, R. (2017a). Towards an optimized blockchain for iot. In *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*, pages 173–178. ACM.

[22] Dorri, A., Kanhere, S. S., Jurdak, R., and Gauravaram, P. (2017b). Blockchain for iot security and privacy: The case study of a smart home. In *Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on*, pages 618–623. IEEE.

[23] Esposito, S., Cafiero, A., Giannino, F., Mazzoleni, S., and Diano, M. M. (2017). A monitoring, modeling and decision support system (dss) for a microalgae production plant based on internet of things structure. *Procedia Computer Science*, 113:519 – 524. The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2017) / The 7th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2017) / Affiliated Workshops.

[24] Fridgen, G., Radszuwill, S., Urbach, N., and Utz, L. (2018). Cross-organizational workflow management using blockchain technology-towards applicability, auditability, and automation.

[25] Gonzalez, J. L. and Marcelín-Jiménez, R. (2011). Phoenix: a fault-tolerant distributed web storage based on urls. In *Ninth IEEE International Symposium on Parallel and Distributed Processing with Applications*, pages 282–287. IEEE.

[26] Gonzalez, J. L., Perez, J. C., Sosa-Sosa, V. J., Sanchez, L. M., and Bergua, B. (2015). Skycds: A resilient content delivery service based on diversified cloud storage. *Simulation Modelling Practice and Theory*, 54:64–85.

[27] Gonzalez-Compean, J., Sosa-Sosa, V., Diaz-Perez, A., Carretero, J., and Yanez-Sierra, J. (2018a). Sacbe: A building block approach for constructing efficient and flexible end-to-end cloud storage. *Journal of Systems and Software*, 135:143–156.

[28] Gonzalez-Compean, J., Sosa-Sosa, V. J., Diaz-Perez, A., Carretero, J., and Marcelin-Jimenez, R. (2018b). Fedids: a federated cloud storage architecture and satellite image delivery service for building dependable geospatial platforms. *International Journal of Digital Earth*, 11(7):730–751.

[29] Hale, J. S., Li, L., Richardson, C. N., and Wells, G. N. (2017). Containers for portable, productive, and performant scientific computing. *Computing in Science & Engineering*, 19(6):40–50.

[30] Huckle, S., Bhattacharya, R., White, M., and Beloff, N. (2016). Internet of things, blockchain and shared economy applications. *Procedia computer science*, 98:461–466.

[31] Huh, S., Cho, S., and Kim, S. (2017). Managing iot devices using blockchain platform. In *Advanced Communication Technology (ICACT), 2017 19th International Conference on*, pages 464–467. IEEE.

[32] Hukkinen, T., Mattila, J., Seppälä, T., et al. (2017). Distributed workflow management with smart contracts. Technical report, The Research Institute of the Finnish Economy.

[33] Jesus, E. F., Chicarino, V. R., de Albuquerque, C. V., and Rocha, A. A. d. A. (2018). A survey of how to use blockchain to secure internet of things and the stalker attack. *Security and Communication Networks*, 2018.

[34] Karlsson, K., Jiang, W., Wicker, S., Adams, D., Ma, E., van Renesse, R., and Weatherspoon, H. (2018). Vegvisir: A partition-tolerant blockchain for the internet-of-things. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 1150–1158. IEEE.

[35] Kasinathan, P. and Cuellar, J. (2018). Securing the integrity of workflows in iot. In *EWSN*, pages 252–257.

[36] Khan, M. A. and Salah, K. (2018). Iot security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems*, 82:395–411.

[37] Kim, K. D., Kim, Y. H., Kwak, B. K., and Lim, G. Y. (2010). Systems and methods for automating a process of business decision making and workflow. US Patent 7,653,566.

[38] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J. B., Grout, J., Corlay, S., et al. (2016). Jupyter notebooks-a publishing format for reproducible computational workflows. In *ELPUB*, pages 87–90.

[39] Lamport, L. (1998). The part-time parliament. *ACM Transactions on Computer Systems (TOCS)*, 16(2):133–169.

[40] Li, X., Jiang, P., Chen, T., Luo, X., and Wen, Q. (2017). A survey on the security of blockchain systems. *Future Generation Computer Systems*.

[41] Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., and Njilla, L. (2017). Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and

availability. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 468–477. IEEE Press.

[42] Madsen, M. F., Gaub, M., Høgnason, T., Kirkbro, M. E., Slaats, T., and Debois, S. (2018). Collaboration among adversaries: Distributed workflow execution on a blockchain. In *2018 Symposium on Foundations and Applications of Blockchain*.

[43] Majithia, S., Shields, M., Taylor, I., and Wang, I. (2004). Triana: A graphical web service composition and execution toolkit. In *Proceedings. IEEE International Conference on Web Services, 2004.*, pages 514–521. IEEE.

[44] Medvedev, A., Fedchenkov, P., Zaslavsky, A., Anagnostopoulos, T., and Khoruzhnikov, S. (2015). Waste management as an iot-enabled service in smart cities. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, pages 104–115. Springer.

[45] Miles, A., Zaslavsky, A., and Browne, C. (2018). Iot-based decision support system for monitoring and mitigating atmospheric pollution in smart cities. *Journal of Decision Systems*, 27(sup1):56–67.

[46] Mingxiao, D., Xiaofeng, M., Zhe, Z., Xiangwei, W., and Qijun, C. (2017). A review on consensus algorithm of blockchain. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2567–2572. IEEE.

[47] Misbahuddin, S., Zubairi, J. A., Saggaf, A., Basuni, J., Sulaiman, A., Al-Sofi, A., et al. (2015). Iot based dynamic road traffic management for smart cities. In *2015 12th International Conference on High-capacity Optical Networks and Enabling/Emerging Technologies (HONET)*, pages 1–5. IEEE.

[48] Mitzenmacher, M. (2001). The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1094–1104.

[49] Montella, R., Brizius, A., Di Luccio, D., Porter, C., Elliot, J., Madduri, R., Kelly, D., Riccio, A., and Foster, I. (2018a). Using the face-it portal and workflow engine for operational food quality prediction and assessment: An application to mussel farms monitoring in the bay of napoli, italy. *Future Generation Computer Systems*.

[50] Montella, R., Di Luccio, D., and Kosta, S. (2018b). Dagon*: Executing direct acyclic graphs as parallel jobs on anything. In *2018 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, pages 64–73. IEEE.

[51] Montella, R., Di Luccio, D., Marcellino, L., Galletti, A., Kosta, S., Brizius, A., and Foster, I. (2017). Processing of crowd-sourced data from an internet of floating things. In *Proceedings of the 12th Workshop on Workflows in Support of Large-Scale Science*, page 8. ACM.

[52] Montella, R., Kelly, D., Xiong, W., Brizius, A., Elliott, J., Madduri, R., Maheshwari, K., Porter, C., Vilter, P., Wilde, M., et al. (2015). Face-it: A science gateway for food security research. *Concurrency and Computation: Practice and Experience*, 27(16):4423–4436.

[53] Montoliu, R., Blom, J., and Gatica-Perez, D. (2013). Discovering places of interest in everyday life from smartphone data. *Multimedia tools and applications*, 62(1):179–207.

[54] Mortimore, S. and Wallace, C. (2013). *HACCP: A practical approach*. Springer Science & Business Media.

[55] Mukherjee, D., Pal, D., and Misra, P. (2017). Workflow for the internet of things. In *ICEIS (2)*, pages 745–751.

[56] Nakamoto, S. et al. (2008). Bitcoin: A peer-to-peer electronic cash system.

[57] Networking, C. V. (2018). Cisco global cloud index: Forecast and methodology 2016–2021. *White paper*.

[58] Pinno, O. J. A., Grégio, A. R. A., and De Bona, L. C. (2017). Controlchain: Blockchain as a central enabler for access control authorizations in the iot. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE.

[59] Reyna, A., Martín, C., Chen, J., Soler, E., and Díaz, M. (2018). On blockchain and its integration with iot. challenges and opportunities. *Future Generation Computer Systems*, 88:173–190.

[60] Sadouskaya, K. et al. (2017). Adoption of blockchain technologyin supply chain and logistics.

[61] Sanchez, D., Di Luccio, D., Gonzalez, J., and Montella, R. (2019). Internet of things orchestration using dagon* workflow engine. *URL: http://www.jkjmanagement.com/wfiot201-2/papers/1570520591.pdf*.

[62] Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and Tschofenig, H. (2017). Authentication and authorization for constrained environments (ace). *Internet Engineering Task Force, Internet-Draft draft-ietf-aceoauth-authz-07*.

[63] Sharma, P., Chaufournier, L., Shenoy, P., and Tay, Y. (2016). Containers and virtual machines at scale: A comparative study. In *Proceedings of the 17th International Middleware Conference*, page 1. ACM.

[64] Siddiqa, A., Hashem, I. A. T., Yaqoob, I., Marjani, M., Shamshirband, S., Gani, A., and Nasaruddin, F. (2016). A survey of big data management: Taxonomy and state-of-the-art. *Journal of Network and Computer Applications*, 71:151–166.

[65] Tärneberg, W., Chandrasekaran, V., and Humphrey, M. (2016). Experiences creating a framework for smart traffic control using aws iot. In *Proceedings of the 9th International Conference on Utility and Cloud Computing*, pages 63–69. ACM.

[66] Taylor, I., Shields, M., Wang, I., and Harrison, A. (2007). The triana workflow environment: Architecture and applications. In *Workflows for e-Science*, pages 320–339. Springer.

[67] Terstyanszky, G., Kukla, T., Kiss, T., Kacsuk, P., Balaskó, Á., and Farkas, Z. (2014). Enabling scientific workflow sharing through coarse-grained interoperability. *Future Generation Computer Systems*, 37:46–59.

[68] Tian, F. (2017). A supply chain traceability system for food safety based on haccp, blockchain & internet of things. In *Service Systems and Service Management (ICSSSM), 2017 International Conference on*, pages 1–6. IEEE.

[69] van Der Aalst, W. M., Ter Hofstede, A. H., Kiepuszewski, B., and Barros, A. P. (2003). Workflow patterns. *Distributed and parallel databases*, 14(1):5–51.

[70] Vassiliadis, P. (2009). A survey of extract–transform–load technology. *International Journal of Data Warehousing and Mining (IJDWM)*, 5(3):1–27.

[71] Vučinić, M., Tourancheau, B., Rousseau, F., Duda, A., Damon, L., and Guizzetti, R. (2015). Oscar: Object security architecture for the internet of things. *Ad Hoc Networks*, 32:3–16.

[72] Walker, M. A., Dubey, A., Laszka, A., and Schmidt, D. C. (2017). Platibart: a platform for transactive iot blockchain applications with repeatable testing. In *Proceedings of the 4th Workshop on Middleware and Applications for the Internet of Things*, pages 17–22. ACM.

[73] Wood, G. et al. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32.

[74] Wu, L., Du, X., Wang, W., and Lin, B. (2018). An out-of-band authentication scheme for internet of things using blockchain technology. In *2018 International Conference on Computing, Networking and Communications (ICNC)*, pages 769–773. IEEE.

[75] Yanez-Sierra, J., Diaz-Perez, A., Sosa-Sosa, V., and Gonzalez, J. L. (2015). Towards secure and dependable cloud storage based on user-defined workflows. In *2015 IEEE 2nd international conference on cyber security and cloud computing*, pages 405–410. IEEE.

[76] Yoo, A. B., Jette, M. A., and Grondona, M. (2003). Slurm: Simple linux utility for resource management. In *Workshop on Job Scheduling Strategies for Parallel Processing*, pages 44–60. Springer.

[77] Zhang, Q., Liu, L., Pu, C., Dou, Q., Wu, L., and Zhou, W. (2018). A comparative study of containers and virtual machines in big data environment. *arXiv preprint arXiv:1807.01842*.

[78] Zheng, Y., Fu, H., Xie, X., Ma, W.-Y., and Li, Q. (2011). Geolife gps trajectory dataset-user guide, july 2011. *URL: https://www. microsoft. com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide*.

[79] Zong, F., Bai, Y., Wang, X., Yuan, Y., and He, Y. (2015). Identifying travel mode with gps data using support vector machines and genetic algorithm. *Information*, 6(2):212–227.

[80] Zyskind, G., Nathan, O., et al. (2015). Decentralizing privacy: Using blockchain to protect personal data. In *Security and Privacy Workshops (SPW), 2015 IEEE*, pages 180–184. IEEE.