Information Processing and Management 60 (2023) 103155

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

CD/CV: Blockchain-based schemes for continuous verifiability and traceability of IoT data for edge–fog–cloud

Cristhian Martinez-Rendon [a], J.L. González-Compeán [b], Dante D. Sánchez-Gallegos [b], Jesus Carretero [a,*]

[a] Universidad Carlos III de Madrid, Leganés, Spain
[b] CINVESTAV Unidad Tamaulipas, Ciudad Victoria, Mexico

ARTICLE INFO

Keywords:
Blockchain
Smart contracts
Edge–fog–cloud
Internet of Things
Dataflow model

ABSTRACT

This paper presents a continuous delivery/continuous verifiability (*CD/CV*) method for IoT dataflows in edge–fog–cloud. A CD model based on extraction, transformation, and load (ETL) mechanism as well as a directed acyclic graph (*DAG*) construction, enable end-users to create efficient schemes for the continuous verification and validation of the execution of applications in edge–fog–cloud infrastructures. This scheme also verifies and validates established execution sequences and the integrity of digital assets. CV model converts ETL and *DAG* into business model, smart contracts in a private blockchain for the automatic and transparent registration of transactions performed by each application in workflows/pipelines created by CD model without altering applications nor edge–fog–cloud workflows. This model ensures that IoT dataflows delivers verifiable information for organizations to conduct critical decision-making processes with certainty. A containerized parallelism model solves portability issues and reduces/compensates the overhead produced by CD/CV operations. We developed and implemented a prototype to create CD/CV schemes, which were evaluated in a case study where user mobility information is used to identify interest points, patterns, and maps. The experimental evaluation revealed the efficiency of CD/CV to register the transactions performed in IoT dataflows through edge–fog–cloud in a private blockchain network in comparison with state-of-art solutions.

1. Introduction

Internet of Things (IoT) is a paradigm in which physical objects, with features such as computing, communication, and sensing capabilities, are deployed in the field. These sensors and IoT devices collect and transmit environmental variables to the fog/cloud.[1] In this trip from the IoT devices to the cloud, large volumes of data are processed by sets of applications (*workflows/pipelines*) transforming data into information assets, which are relevant pieces for organizations to support decision-making processes (Esposito, Cafiero, Giannino, Mazzoleni, & Diano, 2017; Medvedev, Fedchenkov, Zaslavsky, Anagnostopoulos, & Khoruzhnikov, 2015; Miles, Zaslavsky, & Browne, 2018). Currently, workflows engines (Deelman et al., 2005, 2019; Sánchez-Gallegos, Di Luccio, Gonzalez-Compean, & Montella, 2019) and pipeline frameworks (Armenise, 2015; Sánchez-Gallegos et al., 2020)

* Corresponding author.
E-mail addresses: cristhma@pa.uc3m.es (C. Martinez-Rendon), joseluis.gonzalez@cinvestav.mx (J.L. González-Compeán), dante.sanchez@cinvestav.mx (D.D. Sánchez-Gallegos), jcarrete@inf.uc3m.es (J. Carretero).

[1] The total amount of data created (and not necessarily stored) by any device will reach 847 ZB per year by 2021, compared to 218 ZB per year in 2016 (Networking, 2018).

https://doi.org/10.1016/j.ipm.2022.103155
Received 24 April 2022; Received in revised form 29 October 2022; Accepted 1 November 2022
Available online 12 November 2022

have been successfully applied to the establishing of dataflows from the IoT devices to the cloud for decision-making procedures in domains such as health (Mrozek, Koczur, & Małysiak-Mrozek, 2020; Sánchez-Gallegos et al., 2020), transport (Ruan & Shi, 2016), smart cities (Mocanu, Pop, Mihaita, Dobre, & Castiglione, 2019), and environment, to name a few.

In *workflows/pipelines*, sets of applications (Terstyanszky et al., 2014) are managed as generic stages (i.e., acquisition, transmission, storage/processing, and analysis) for processing IoT data (Sánchez-Gallegos et al., 2019; Sánchez-Gallegos, Di Luccio, Kosta, Gonzalez-Compean, & Montella, 2021) regardless of the use case (Sánchez-Gallegos, Carrizales-Espinoza et al., 2020; Sánchez-Gallegos et al., 2021). Those applications are executed on different infrastructures (edge–fog–cloud) by following a sequence defined by a Directed Acyclic Graph (DAG), which produces dataflows from the IoT through the edge to the fog/cloud. In many cases, they also involve several actors (organizations and users), which poses a challenge for organizations to establish verification of the transactions performed by the participants in the dataflows built by the workflow engines and pipeline frameworks. It is essential for organizations, not only to verify that the execution of applications is performed in the strict sequence previously established in a DAG by authenticated participants, but also to verify that the incoming and outgoing IoT data of each stage of a workflow/pipeline have not been altered by third parties or by the users associated to the organizations participating in a workflow/pipeline. Those verification and validation tasks result crucial for organizations to carry out critical decision-making processes in a confident manner, as any alteration in any data exchange performed by the workflow/pipeline applications would produce erroneous information, which would hinder the work of decision-makers.

One approach to face up this issue is monitoring and registering all transactions performed by the applications, as well as the alterations of the contents performed during the transactions, in a trusted environment. However, performing this type of verification in dataflows deployed on edge–fog–cloud environments implies integrating the workflows/pipelines of applications with existing *verifiability networks* (public/private blockchain networks) (Nakamoto et al., 2008; Reyna, Martín, Chen, Soler, & Díaz, 2018). This integration is not trivial and becomes a challenge for organizations in two directions: the first one is losing the generalization of workflows engine and/or pipeline frameworks, which should be modified to include embedded blockchain module; the last one are the costs (overhead) of the registration of transactions performed by applications through the edge–fog–cloud environments, which produces an efficiency issue that is evident in the form of delays in the delivery of data to the applications creating the dataflows and the delivery of information assets to the decision-making processes. Current traditional workflow engines and pipeline frameworks usually do not consider services to face up these problems.

In this paper, we present a *Continuous Delivery/Continuous Verifiability* (CD/CV), a continuous transaction verification method for IoT data in edge–fog–cloud workflows. CD/CV is a schema that creates a private blockchain network and integrates it to the workflows/pipelines to keep a record of the transactions or processes performed in each stage (participants in a dataflow) of these workflows. The combination of CD and CV models creates CD/CV schemes that combine the design principles of the blockchain and the smart contracts with $DAGs$ of IoT workflows to successfully achieve an integration (Reyna et al., 2018) of both technologies.

The CD model is based on concepts such as continuous deliver, extraction, transformation, and load (ETL) and directed acyclic graphs (DAG). The continuous delivery ensures the coupling of applications in the form of workflows and/or pipelines. The CV model creates a business model based on ETL, smart contracts based on DAGs, and continuous verifiability processes based on continuous deliver. Our solution allows the registration of tasks' events (transactions) of the stages (business model) managed by the end-users (participants) for each type of content (assets) processed during IoT dataflows, which allows end-users to verify the accomplishment of the contracts created in the blockchain by CV components during an IoT dataflow.

The main contributions of this paper are:

1. A *Continuous Delivery/Continuous Verifiability* model allowing automatic deployment of integrated workflows for IoT–edge–fog–cloud systems and blockchain networks to record the actions performed in the workflows for continuous verification along system lifecycle. The deployment is made based on user specifications defined as extract/transform/load (ETL) operations and connections of components defined as a Directed Acyclic Graph (DAG).
2. *Automatic deployment of smart contracts*. By allowing the association of any workflow stage with the blockchain adding ETL mechanisms, CD/CV allows the deployment of smart contracts to verify the transactions of each stage in the workflow is automatic and independent of the particular purpose of each application producing an IoT dataflow to preserve the generality of the solutions.
3. *Reducing the overhead of transaction registration*. Blockchain is slow to cope with real-time transactions, which affects the user service experience (decision-makers) of IoT workflows. As an initial step to alleviate this overhead, we have used an enhanced implementation of the Hyperledger blockchain transactions that was previously presented by the authors (Martinez-Rendon, Camarmas-Alonso, Carretero, & Gonzalez-Compean, 2021). Moreover, CD/CV provides transparently parallel patterns to enhance data transmission and logging.

To assess the feasibility of CD/CV, we have developed and implemented a prototype to create CD/CV schemes, which was evaluated in a transport case study where mobility and temperature information from fleet of trucks were used to create patterns and maps in a geographical information system. Registering temperatures of the load in the trucks and the routes, speed, and other parameters was critical to accept the load in destination, to provide maintenance to trucks, and to solve problems with the load before arriving to destination. The experimental evaluation revealed the efficacy of CD/CV to register the transactions performed in edge–fog–cloud workflows in a private blockchain network, as well as the performance efficiency in comparison with state-of-art solutions.

The organization of the rest of the document is as follows. Section 2 presents related work. Section 3 presents the proposed schemes, the methodology followed for its development, its design, and each of its components. The case study solved with our prototype is presented in Section 4. Section 5 presents the experiments made with the case study and their results. Finally, Section 7 presents the main conclusions of this work.

2. Related work

Popular scientific workflow engines such as Pegasus (Deelman et al., 2019), Triana (Kousalya, Balakrishnan, & Raj, 2017), and Sacbe (Gonzalez-Compean, Sosa-Sosa, Diaz-Perez, Carretero, & Yanez-Sierra, 2018) are not designed to cope with IoT environments, but to work with heavy processes in most cases. Other solutions based on libraries, such as DagOnStar (Montella, Di Luccio and Kosta, 2018) and Parsl (Scalable Parallel Scripting) (Babuji et al., 2019), provides a model for managing workflows composed of Python functions and external applications in arbitrary computational resources. Any of them are designed to deploy efficient workflows on IoT systems.

As an example, we used DagOnStar in a previous work to orchestrate Internet of Things devices (Sánchez-Gallegos et al., 2019) and we found that the schema that DagOnStar uses to declare the tasks needs a one-to-one declaration for each data to be processed. For a scenario in which there are millions of data to be processed this would represent a major problem. On the other hand, there are not implicit procedures to establish some control operations (security, reliability, traceability, etc.) in the processes performed in the workflows for each of the declared tasks. One approach to this is shown in Meroni, Baresi, Montali, and Plebani (2018), where the authors perform Multi-party business process compliance monitoring by applying business process notation (BPMN) and artifact-centric analysis, which are enabled with IoT. However, the monitoring is focused on the way physical objects are affected by the execution of a workflow process.

2.1. Workflows engines for IoT

Trying to cope with the integration of IoT and workflows, there have been new proposals for the integration of IoT into operational workflows for real-time and automated decision-making environments. An example was presented in Louis and Dunston (2018) for repetitive construction operations. Another interesting idea was Osmotic flow computing (Nardelli, Nastic, Dustdar, Villari, & Ranjan, 2017), a paradigm that enables the automatic deployment of microservices over inter-connected Edge and Cloud Data Centers. In the Osmotic Flow model, an IoT workflow application is modeled as a directed graph with data transformation tasks as its nodes, and dataflow dependencies (or control flow dependencies for computational synchronization, if/as needed) between data transformation tasks as its vertices. Osmotic Flow model permits data transformation tasks to be distributed, managed, and executed across any available CDC and EDC provider coordinated by a Osmotic Resource Manager. FairWind (Montella, Ruggieri and Kosta, 2018) is an intelligent navigation system for obtaining high resolution 3-D maps from the bottom of the sea. These maps are necessary for different processes: (1) modeling the impact of storm waves in coastal human activities, (2) the diffusion and dispersal of pollutants, (3) drift of floating objects (safety at sea - bathymetry). These processes are carried out through workflows, cloud computing and CUDA, and Internet of Things to collect data captured by vessel sensors. The main problem of FairWind is that is was designed for a specific use case, and it is not easy to adapt to other applications. Moreover, the infrastructure of computation depends on a third party (Amazon Web Services).

Recently, a new IoT workflow composition system (IoTWC), which allows IoT users to define their workflows with proposed IoT workflow activity abstract patterns (e.g., data capture, data store, data visualization), has been presented (Li et al., 2020). IoTWC leverages the analytic hierarchy process (AHP) to compose the multi-level IoT workflow that satisfies the requirements of any IoT application. An analysis, made using a smart home scenario, showed the effectiveness of IoTWC in terms of IoT workflow abstraction and composition. However, the paper lacks a real-world evaluation to measure performance of the solution, that seems to be mostly theoretical. All the former tools lack verifiability and traceability in the workflow stages deployed.

From the literature reviewed, we could see that major challenges for IoT workflows in collaborative edge, fog, and cloud environments are related to data placement strategies (Firouzi, Farahani, & Marinšek, 2021; Shao, Li, & Tang, 2019), providing self-adapting services orchestration (Serhani et al., 2020), interoperability of data and processes between the different stages (Ahmad, Cuomo, Wu, & Jeon, 2019; Firouzi et al., 2021), portability of tasks (Qasha, Cala, & Watson, 2016), computation offloading (Shahhosseini et al., 2021), programmability and flexibility (Municio, Marquez-Barja, Latré, & Vissicchio, 2018), efficiency (Simpkin et al., 2020), and finally verifiability.

Verifiability is a critical property in IoT workflows where data are used to control quality of goods, medicines, food, etc. (Yu, Yan, & Vasilakos, 2017). One crucial problem is that data processing result in IoT workflows may be incorrect, thus cannot be fully trusted. How to process and compute the outsourced data in a secure and trustworthy way becomes a crucial security issue with high challenges. To cope with this problem, one solution proposed is to integrate blockchain and smart contracts with IoT operations (Christidis & Devetsikiotis, 2016; Fernández-Caramés & Fraga-Lamas, 2018; Li et al., 2021; Qin, Huang, Yang, & Li, 2021; Singh & Singh, 2020; Taherkordi & Herrmann, 2018). As some examples, in Bumblauskas, Mann, Dugan, and Rittmer (2020) the authors showed a supply chain traceability system for food safety based on blockchain and Internet of things, and in Mazzei et al. (2020) the authors presented a Blockchain tokenizer for industrial IoT trustless applications,

However, the unique characteristics of IoT systems, such as heterogeneity and pervasiveness, pose challenges in designing smart contracts for such systems (Panarello, Tapas, Merlino, Longo, & Puliafito, 2018; Reyna et al., 2018). Moreover, even if the blockchain technology provide decentralized security and privacy (Zhao, Chen, Liu, Baker, & Zhang, 2020), efficiency is a major issue, as it may involve power surge, high latency operations, and considerable computation, which is not adequate in some settings (Bhushan, Sahoo, Sinha, & Khamparia, 2020). A cost analysis of internet of things sensor data storage on blockchain via smart contracts is shown in Kurt Peker, Rodriguez, Ericsson, Lee, and Perez (2020). The study showed that even though expensive, for those applications where the integrity and transparency of data are crucial, storing IoT sensor data on Ethereum could be a reliable solution.

In a previous work (Martinez-Rendon et al., 2021) we made a proposal to target efficiency issues in blockchain. The technical approach used consist of performing real-time data acquisition for each sensor value independently and recording those values as single blocks in the chain. Those blocks are later consolidated in a bigger block for each sensor, but this is made outside the recording process and, thus, it does not delay data capturing.

The first optimization proposed consisted of storing each transaction carried out by a sensor in an atomic form, creating a new block for each transaction from a sensor. This block includes the minimum information needed for the transaction to be executed, which makes it feasible to be integrated with microservices. This avoids the need of recovering all values previously stored by the sensor to concatenate the new value afterward, as it happened in the models currently using blockchain. The effect of this optimization is critical, as the size of the transactions made by each sensor remains constant over time and the transaction replication time in all the peer nodes is also constant for every sensor. We proposed a second optimization technique in our model: modifying the maximum number of transactions that can be stored within a validation block. It also uses atomic transactions to ensure that they are independent of each other. This optimization allows increasing the performance of the blockchain network since the validation process (the phase that produces the highest latency) is not carried out every time a transaction is made, but by groups of transactions, which reduces the number of times the validation process is performed on the network.

We are using those enhancements, together with the ETL facilities in our model, which allows to include it in any IoT without affecting the workflow tool. This is not possible in other existing tools, such as DagOnStar.

2.2. Blockchain technology and consensus protocols

Blockchain technology can be classified attending to the nature of the blockchain network created. Basically, there are two main types: open and provide networks. In open networks, anyone can participate and register operations in the blockchain. Ethereum (Ethereum, 2020) is the main representative of this category, being its most successful example of usage the blockchain implementation of modern cryptocurrency developments, like bitcoin Vujičić, Jagodić, and Randjić (2018). In private networks, only some peers have the capacity to record and manage operations in the blockchain system, being Hyperledger Fabric (Hyperledger-Fabric, 2020) the de-facto standard platform for this kind of networks. In contrast to public networks, private networks only have participants who are members of a particular organization (Guegan, 2017). This removes the anonymity of the participants in the transactions, because they must authenticate their identity with a certificate. In those networks, it is a lot easier to modify a transaction, but the immutability level is still high. In our work, we chose to use Hyperledger Fabric, as we are creating networks of entities with different privileges.

Hyperledger Fabric achieves consensus by relying on a backend service (known as the ordering service) that intermediates the messages between senders and receivers (Xu et al., 2021). This backend service will ensure that all receivers will see messages in the same order — it follows that if all receivers see messages in the same order, they will perform the same actions/commits, etc. and the consensus is achieved. Currently Fabric 2.x uses Raft, a Leader/Follower type, consensus algorithm (Pahlajani, Kshirsagar, & Pachghare, 2019). Moreover, Hyperledger Fabric uses Crash Fault Tolerance (CFT) to achieve consensus for single as well as multiple org systems. Crash Fault Tolerant model guaranties to withstand system failures, such as crashes, network partitioning. Having N nodes in your consensus system CFT capable to withstand up to N/2 such crashes.

However, blockchain performance has been always an issue, especially when applied to real-time systems, like IoT. Several authors have made performance analysis on the different available development platforms (Nasir, Qasse, Abu Talib, & Nassif, 2018; Nguyen, Jourjon, Potop-Butucaru, & Thai, 2019) and other studies have recently proposed optimizations to increase the performance and scalability of the tools based on RAFT (Fu, Wei, & Tong, 2021; Wang, Li, Xu, & Xu, 2021). Those aspects are especially important for those environments including IoT devices, which send a big volume of data in a short time, making the management and storage of the data more difficult. However, most works propose to modify the architecture of the Fabric development tool so that the validation phase has a shorter execution time, as it is the main bottleneck detected in blockchain technology. The result is that those optimizations depend on the development tool and the modified consensus protocol used to validate the transactions. Thus, they are not generic optimizations and in many cases they are not adopted by the users.

In a previous work, trying to alleviate this overhead, we developed an enhanced implementation of the Hyperledger blockchain transactions (Martinez-Rendon et al., 2021) that does not need to modify the Fabric. We proposed to use atomic transactions and grouped validation techniques, which are applied on top of the blockchain platform and that are not dependent on any specific platform. Experiments results showed that the optimizations carried out on the transactions allowed to increase the performance and scalability of transactions validation in the blockchain platform, allowing high speed validation and storing the data consistently in near real-time, allowing the data to be available for query in the continuous validation of the contract.

2.3. Data management in IoT workflows

The typical data handling that is done in workflows is described in Fig. 1.

The taxonomy that is illustrated in the figure is based on the work presented in Siddiqa et al. (2016). In this taxonomy, there are three types of data processing:

- Treatment for decision-making. It basically consists of the processes applied to the data either for the extraction of knowledge and patterns in the data (decremental process), or to generate added value to them (incremental process) (He, Wang, & Akula, 2017).

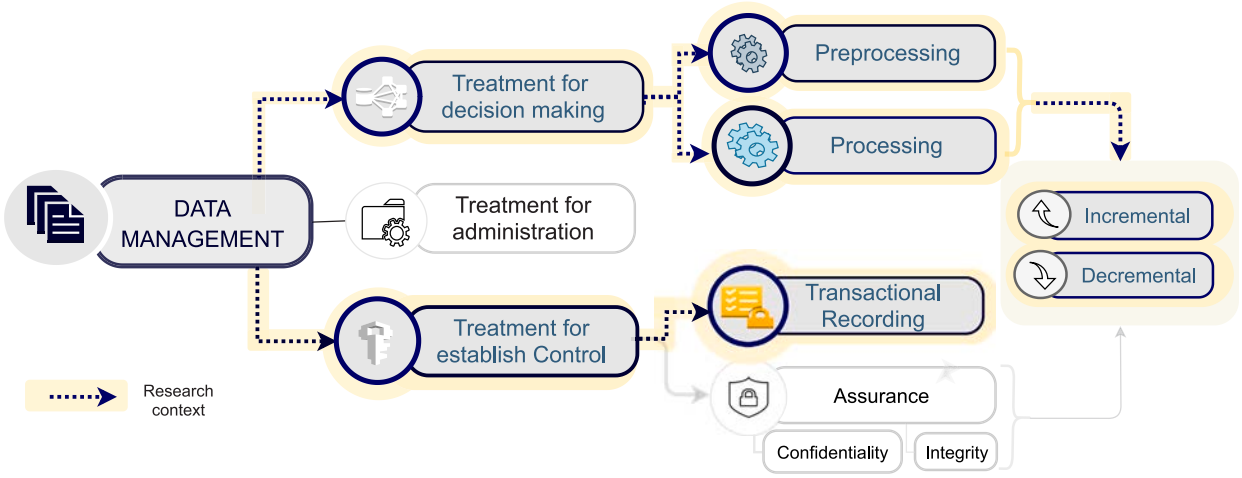

Fig. 1. Taxonomy of techniques of data management in IoT.

- Administration of meta-data. It refers to the management of the meta-data of the data, such as where the data is located, where it comes from, who it is, size, etc. (Pinoli, Ceri, Martinenghi, & Nanni, 2019).
- Procedures to establish control of the data. It basically corresponds to two tasks, the first is the traceability record of the operations on the data necessary to avoid problems such as non-repudiation (Lofstead, Baker, & Younge, 2019), and the second task is the assurance (for example, confidentiality and integrity) that is required in sensitive data now. They are transmitted during the different stages of the workflow and primarily in cloud environments (Goble et al., 2020).

In addition, in Diène, Rodrigues, Diallo, Ndoye, and Korotaev (2020) the authors present an insight into the data management techniques for IoT. The study includes the most relevant concepts of IoT data management and a comparison of surveys on the topic. The authors highlight new IoT data management methods, such as middleware or architecture-oriented solutions to facilitate efficient storage, the integration of generated data, and indexing methods of structured and unstructured data. In general, the authors classify these methods around three main principles: data collection, data management system design, and data processing. As future work, the authors propose mechanisms using future technologies such as Fog computing to improve data management.

The traditional solutions for verification of transactions are focused on a single piece of software with which the end-users are dealing with. This type of solution is not suitable for workflows used in IoT dataflows, as a workflow is composed by different pieces of software (applications).

Some solutions have been proposed to avoid Over-centralization of data. Ramachandran et al. (2020) describe different approaches for distributing the verification and data management aspect between Linked Data and the distributed ledger, keeping the integrity of the stored information intact through Blockchain-based verification. Helo presented in Helo and Shamsuzzoha (2020) a pilot system of a cloud-based portal for real-time tracking and tracing of logistics and supply chains using IoT and blockchain. The architecture of the proposed portal system is connected to transport companies, tracking devices, consolidation points and suppliers. Braun, Traue, Lingl, and Käfer (2021) have recently presented an approach for documenting the execution of inter-organizational workflows on a distributed ledger, with the possibility of adding selectively shared verifiable data to the workflow instances' documentation. It allows to record events, but the solutions is tailored to the specific use case. Shukla, Lin, and Seneviratne (2022) have presented this year a system called BlockIoT which connects personal Internet of Medical Things (IoMT) devices to Electronic Health Records using blockchain technology to provide a trustworthy and reliable method for aggregating IoMT device data and smart contracts that automatically provide relevant alerts to the healthcare providers.

Table 1 shows a summary of similar solutions from the state-of-the-art focused on the management, processing, and traceability of data through the edge–fog–cloud. We have classified these solutions according to the scope of the solution (Blockchain: use smart contracts and Workflow: structures for the processing of data), according to their environment (Multiple: if the solution is deployable in any of the edge, the fog, and the cloud), according to their deployment (Distributed: on multiple compute nodes and Container: encapsulated and interoperable solution), according to the data verification process (Real-Time: Receiving real-time data streaming, Continuous: issuing alerts, and traceability at any time of the data lifecycle), whether they include a visualization component (e.g., Dashboard to visualize the resulting data or analysis), according to the domain or use case (Multiple: solution available for different use cases, and IoT: If the data flow is coming from IoT devices), and according to the use of efficiency techniques to reduce the time required to manage and process the data (Parallelism: if data processing or verification processes in parallel and Scalable: when the solution allows increasing its nodes/services without ceasing to operate).

Solutions such as Jenkins (Armenise, 2015), Pegasus (Deelman et al., 2005), Slurm (Yoo et al., 2003), PuzzleMesh (Sanchez-Gallegos et al., 2022), DagOnStar (Montella, Di Luccio et al., 2018), Parsl (Babuji et al., 2019) among others, allow the construction of workflows for multiple use cases including IoT data management. Additionally, these solutions are intended to be distributed

Table 1
Summary of state of the art.

Work	Scope		Environment	Deployment		Verification		Visualization	Domains		Efficiency	
	Blockchain	Workflows	Multiple	Distributed	Container	Real-Time	Continuous	Dashboard	Multiple	IoT	Parallelism	Scalable
Jenkins (Armenise, 2015)	–	✓	–	✓	✓	–	–	–	✓	–	✓	✓
Pegasus (Deelman et al., 2005)	–	✓	–	✓	✓	–	–	–	✓	–	✓	✓[a]
Slurm (Yoo, Jette, & Grondona, 2003)	–	✓	–	✓	–	–	–	–	✓	–	✓	✓
PuzzleMesh (Sanchez-Gallegos et al., 2022)	–	✓	✓	✓	✓	–	–	–	✓	–	✓	✓
DagOnStar (Montella, Di Luccio et al., 2018)	–	✓	✓	✓	✓	–	–	–	✓	✓	✓	✓
Parsl (Babuji et al., 2019)	–	✓	–	✓	✓	–	–	–	✓	–	✓	✓[a]
Martinez (Martinez-Rendon et al., 2021)	✓	✓	–	✓	✓	✓	✓	✓	–	✓	–	✓
Sacbe (Gonzalez-Compean et al., 2018)	–	✓	–	✓	✓	–	–	–	–	–	✓	–
Nasir (Nasir et al., 2018)	✓	–	–	–	✓	✓	–	–	–	–	–	✓
Ramachandran (Ramachandran et al., 2020)	✓	–	–	✓	✓	✓	–	✓	–	–	–	–
Helo (Helo & Shamsuzzoha, 2020)	✓	–	–	✓	✓	✓	✓	✓	–	✓	–	✓
Braun (Braun et al., 2021)	✓	✓	–	✓	–	–	–	–	✓	–	–	–
Shukla (Shukla et al., 2022)	✓	✓	–	✓	–	✓	✓	✓	–	✓	–	✓
CD/CV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

[a]Means that the characteristic is provided by external tools.

and deployed in different infrastructures (e.g., in any of the edge, the fog, or the cloud), including parallelism patterns that allow them to be efficient and scalable. However, they lack the verification component to ensure the traceability of data or the correct execution of the processes that are part of these workflows. Ramachandran in Ramachandran et al. (2020) include the blockchain for a complete decentralized verification of data with confidentiality, however, such work is a conceptual solution that is in the process of development and testing. Implemented works such as that of Helo (Helo & Shamsuzzoha, 2020) and Shukla (Shukla et al., 2022) use the B = blockchain for process verification on IoT and real-time data but are designed and defined for a particular use case, the former for logistics/supply chains and the latter for Electronic Health Records. The latter, together with Braun's work (Braun et al., 2021), integrate blockchain technology with a workflow in a single solution. However, Braun's work does not indicate its applicability in an IoT data environment but focuses on the execution of an interorganizational workflow using the distributed ledger.

Different to the previous examples, the CD/CV schemes proposed in this paper create an intermediary between the applications of workflows components and the blockchain, which allows keeping the generalization of workflows (for organizations to change applications of workflows in a flexible manner). This intermediary facilitates the registration of application actions (by using an implicit ETL model) as well as the sequence of execution of applications (by using DAG structures). CD/CV also provides the automatic deployment of parallel patterns transparent to the application and consolidates transaction registers to improve the performance of the interaction with the workflows with blockchain, which can be reused by current available workflows to implement verifiability networks based on blockchain. Moreover, the CV components enable workflows engines and pipeline frameworks to create continuous verifiability for the transaction registration in distributed structures in edge–fog–cloud environments. Our solution is generic and can be applied to any IoT data management workflow without modifying the applications themselves. The workflow stages provide real-time information or data, while blockchain technology is used to provide a chain of immutable transactions.

3. CD/CV: continuous delivery/continuous verification schemes for traceable IoT dataflows

We propose a method to create (**C**ontinuous **D**elivery/**C**ontinuous **V**erifiability) (CD/CV) schemes allowing to integrate blockchain with workflows engines and/or pipeline builders.

Fig. 2 shows an example of the CD/CV scheme proposed. It shows two layers running in parallel but integrated: a Continuous Delivery system managing the IoT dataflows, and Continuous Verification system that allows to register the transactions performed by each stage of a IoT dataflow and verify them on-line using intelligent contracts.

The CD layer is based on techniques such as continuous delivery technique (Armenise, 2015; Gonzalez, Perez, Sosa-Sosa, Sanchez, & Bergua, 2015), ETL and DAG. Continuous delivery is used in software industry for organizations to split systems into small manageable software pieces; as a result, the pieces can be distributed to different teams that could deploy/execute them through multiple infrastructures and integrate them into a single solution by using control structures based on directed acyclic graph (*DAG*). In this paper we reuse this coupling technique to create edge–fog–cloud dataflows and to deploy the pieces on different infrastructures and to invoke the components of the second part (CV for continuous verification) of the CD/CV schemes. The ETL processing technique (extract, transform, and load) (Vassiliadis, 2009) traditionally is used in big data scenarios for data analysis and the transfer of data between different databases. It is used in our CD model for allowing end-users to provide information about the path of the *E*xtraction of the input data and contents that will be used by the stages (i.e., data produced by an IoT device), the *T*ransformation performed by the applications associated to this stage to the extracted data/content, and the *L*oading of the transformed data/content into another stage or a storage location (e.g., a sink deployed on the cloud). In this structure, the nodes represent the applications of the stages whereas the edges represents the I/O interfaces.

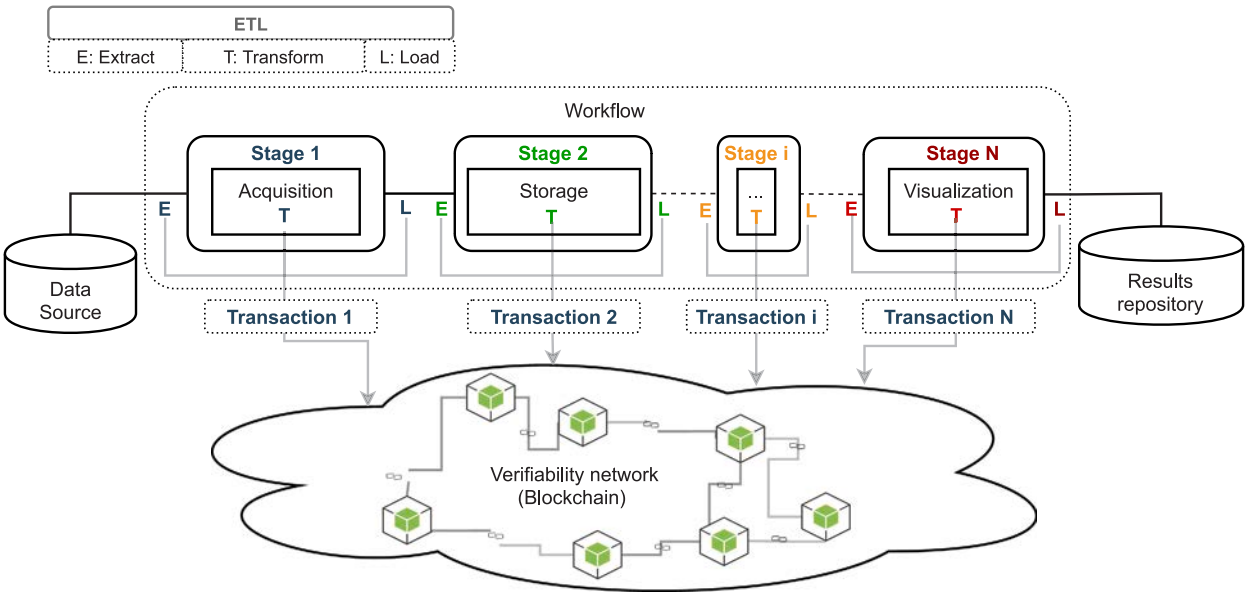

Fig. 2. CD/CV model applied to a workflow.

The CV layer registers the transactions performed by each stage of a IoT dataflow (CD model), thus allowing end-users to create traceability reports and contract verification for each processing phase in an IoT dataflow. The CD model automatically register the transactions performed by the applications by using ETL features and the *DAG* information to create trace routes of the information assets in the dataflow. The CV layer captures the operations performed by each software piece deployed on any of the edge, the fog, or the cloud infrastructures considered in an IoT dataflow. This produces a set of records in the blockchain network (Androulaki et al., 2018; Wood et al., 2014), one per stage in a workflow, concatenated into a single register created by following the *DAG* of the CD model. That enables organizations and end-users to yield a continuous verification of the transactions performed through the workflows by using intelligent contracts on each stage.

3.1. Methodology for building CD/CV schemes

Fig. 3 shows the methodology defined for building CD/CV schemes, consisting of three major steps: preparation, deployment, and operation.

The first phase is a *preparation* step in which all the stages (organizations) that want to participate in a IoT dataflow express ETL and DAG information.

In the *deployment* phase, a **Global Manager (GM)** receives the ETL and DAG information to ensure the effective materialization of this information in the form of traceable workflows by using the CD model. The (*DAG*) defining the dataflows is created to deploy them, components are inserted into the stages, using the ETL paradigm to extract data and events (transactions) from the CD components (stages, nodes, and edges) arising in IoT dataflows. The manager uses ETL components and the *DAG* to create and connect the different stages of the IoT dataflow, and it also establishes control structures for registering the transactions in the verifiability network. Finally, it checks that the continuous delivery (CD) and the continuous verification (CV) for the workflow are deployed. The CV model uses the information from the CD model to create two types of records in the blockchain: records of each service/app using the ETL information, and traceability records using the *DAG* scheme.

Finally, in the *operation* phase, software virtualization has been used to control the launching of both the workflow, and the verifiability network. Clones of the stages can be launched and managed as clusters in the form of parallel patterns (e.g., Manager/Worker pattern (Ortega-Arjona, 2010)), which improve stage performance and reduce the cost of the verifiability network.

3.2. Design of the CD/CV global manager

A manager is required to build and operate a CD/CV scheme. In this sense, Fig. 4 shows the Global Manager proposed, which is composed of a *Construction Manager* and an *Operation Manager*.

The Global CD/CV Manager uses an ETL Interconnection model to get the information from the end-users to create a IoT dataflow and the verifiability network. In this sense, the ETL interconnection model together with the *DAG* scheme interconnect the workflow components with the verifiability network.

The CD components of the model that describe the workflow components are as follows:

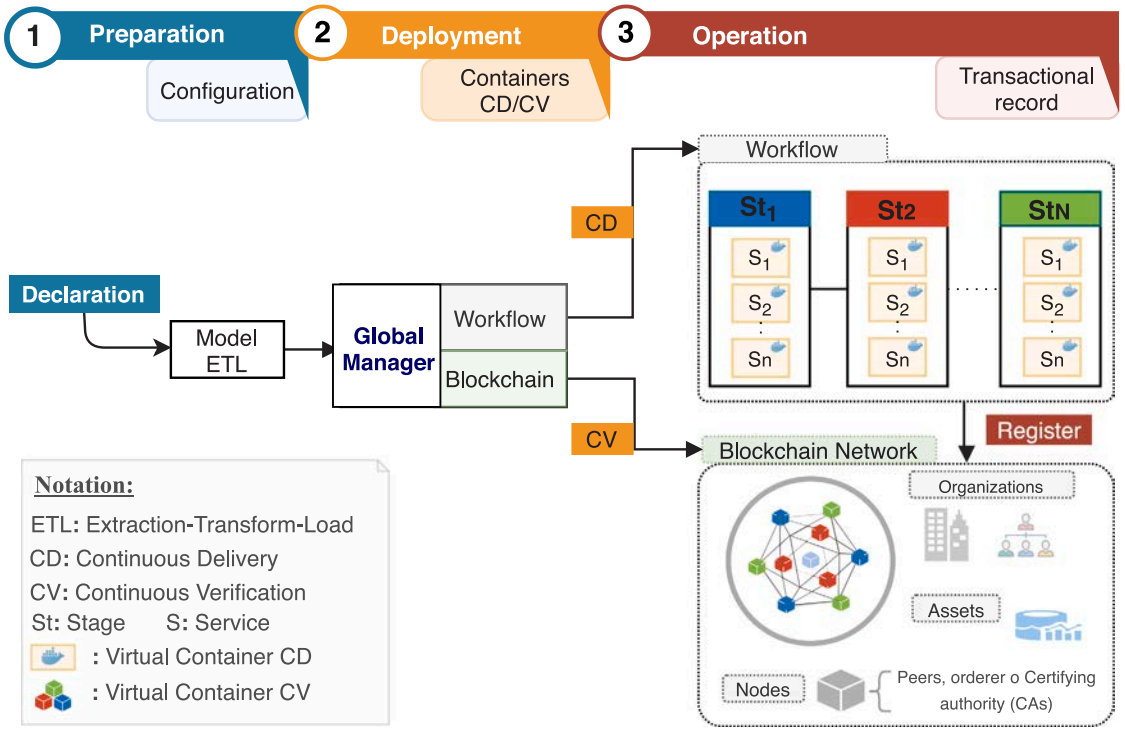

Fig. 3. Methodology proposed for building CD/CV schemes.

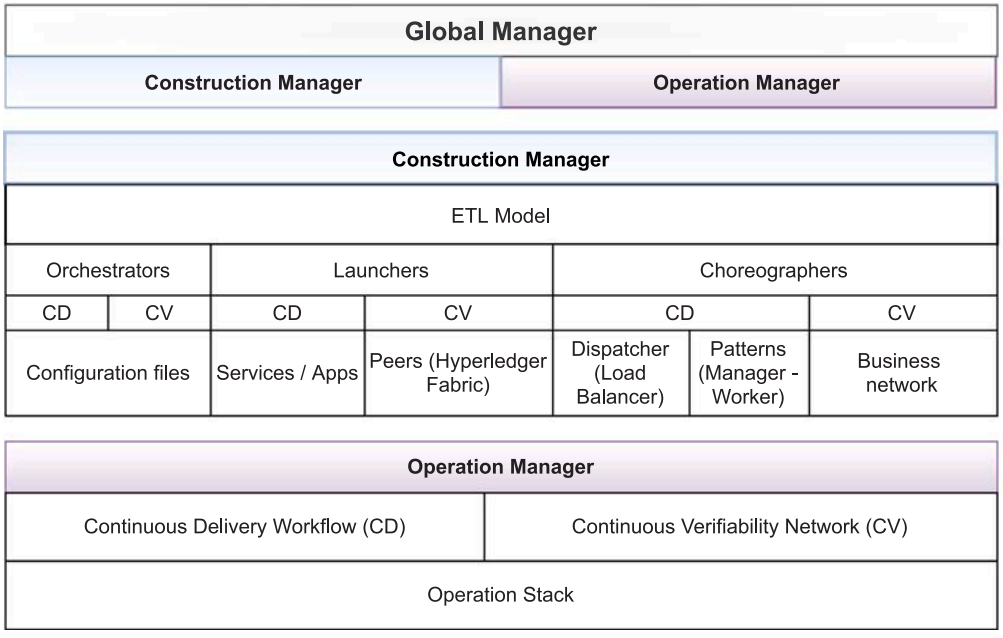

Fig. 4. Modules of the Global CD/CV Manager.

1. **Data Source** (DS): It represents the source of data to be processed through a workflow (W).
2. **Service** (S): It represents the user application that performs specific processing (e.g., S_1: data analysis, S_2 : data compression, S_3 : encryption, among others.)
3. **Stage** (St): It represents the set denoted as $St = \{S_1, S_2, S_3, \ldots, S_m\}$, where m is the number of services (S) including in a given stage (St).

4. **Workflow** (W): The workflow describes the set of stages (St) that collaboratively perform general processing to transform the original data source (DS) into useful information for entities or organizations that require to make decisions from the found information. A workflow is denoted as $W = \{St_1, St_2, \ldots, St_N\}$, where N is the number of processing stages, which are interconnecting ETL model and the *DAG* scheme.
5. **Extract, Transform, Load (ETL):** The ETL process allows a service (S_i) to obtain data from a given source (DS), applies given processing, and loads it in a specific storage system or space so that the result delivered by S_i is the input of the following service (S_{i+1}) to perform the same process, thus forming a workflow (W). This ETL process defined by each $S_i \in St \wedge St \in W$ consists of the following phases:

   (a) **Extract (E):** Indicates where to extract or acquire the data that will be the input of a S_i service. In this phase are considered two attributes $\langle type, path \rangle$. Where $type$ indicates the input type that receives S_i: this can be a folder or a file, and $path$ indicating where are the data to be extracted.
   (b) **Transform (T):** Indicates how S_i will process the data. In this phase, three attributes are considered: $\langle type, path, parameters \rangle$. Where $type$ is the method through which S_i implements the data transformation. It can be a service or an executable (e.g., a JAR). The second attribute is the $path$ where the transformation method is placed in. Finally, $parameters$ are the arguments that the transformation method receives for execution.
   (c) **Load (L):** Indicates where the S_i processing results are delivered. In this phase, two attributes are considered: $\langle type, path \rangle$. Where $type$ is the output type that provides S_i (folder, file), and $path$ where the processing results are loaded, and which are then the input to the service S_{i+1}.

6. **Patterns** (Pa): The pattern considered in this research work is the **Manager/Worker** (Pa_Ma/Wo) **pattern**. This pattern has a Manager in charge of distributing the workload (data to processed) among N Workers. The Manager divides the original data set and assigns it to each of the available workers to process the data source (DS) in parallel.

The elements of the interconnection model that describe the verifiability network are:

1. **Organization** (O): For the workflow component (W), an organization (O) represents an entity that deploys a set of services ($S_1, S_2, \ldots, S_N$). In the proposed model, these services are described as a stage ($St = \{S_1, S_2, \ldots, S_N\}$) that is part of the processing of the data in a workflow (W) in which participates multiple organizations. In this sense, an organization is an entity in charge of one or more stages that belong to a workflow ($St \in W$).
2. **Nodes** (N): in the verifiability network. From the verifiability network point of view, an organization is in charge of a set of Nodes (N) that will be used to create a blockchain network. Each node contains an API REST through which an organization records the transactions it performs in its stages (St) that belong to a workflow ($St \in W_i$). The Nodes can acquire any of the following roles:

   (a) **Peers nodes** (N_{peers}): Those are the nodes of the verifiability network that issue transactions.
   (b) **Order manager node** (N_{or}): establishes a consensus mechanism between the peer nodes when verifying transactions. The order manager node also verifies the sequence of transactions and records consistency.
   (c) **Certification Authority Node** (N_{ca}): manages the certificates and private keys delivered to all the peer nodes belonging to a specific organization.

3. **Transaction** (Tx): It represents an action performed by a service (S) in a workflow (W). The structure of a transaction (Tx) depends on the business network (BN) that was defined.
4. **Blockchain** (BC): is a *peer-to-peer* network or P2P, where are recorded the transactions performed by each service (S).
5. **Business Network** (BN): defines the business logic ruling the transactions performed by the participants in a workflow, as well as the abstraction of the entities and assets included in those transactions.
6. **Administrator** (Ad): manages the access keys associated required for a given organization (O) to make register in blockchain (BC).
7. **Peers by organization** (PxO): are the peer nodes (N_{peers}) will the system deploy on the IT infrastructure of the organization (O).

Once the declarative model of interconnection for the IoT dataflow and the verifiability network are described, the Global Manager, through a Construction Manager and an Operation Manager, can build and operate a CD/CV system.

3.2.1. Construction manager

This Manager has three main components: Orchestrators, Launchers, and Choreographers. They work together to build a CD/CV system from the ETL interconnection model and *DAG* scheme described in the previous section.

3.2.1.1. Construction of the CD/CV system. Two main phases are part of the construction of the CD/CV system: a preparation phase, and a deployment phase.

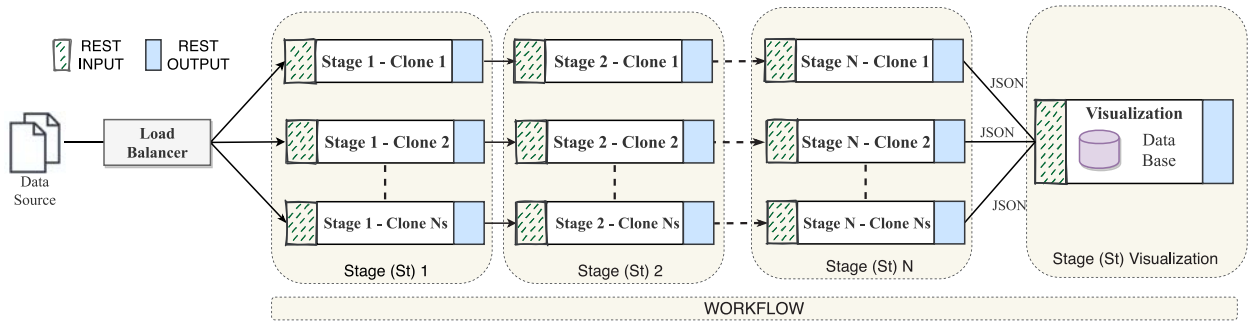

Fig. 5. The architectural pattern of the CD/CV Manager: an example of a service for the analysis/processing of data from IoT environments.

Preparation phase: Interconnection ETL model In the preparation phase, the Construction Manager creates the ETL configuration by using the information delivered by end-users about services at the stages of a workflow ($St = \{St_1, St_2, \ldots, St_N\} \in W$, where $N \rightarrow$ Number of stages). The user (administrator) defines the number of services that will be executed in each St_i as well as the sequence of each $St_i \in W$. The organizations also can define the parallelism degree (services per stage) to improve the stage and workflow performance, which is expected to be used to compensate for the overload produced by the registration of transactions performed by each stage. The number of services and applications are declared in the ETL interconnection model as:

$St_i = Ns_i \rightarrow$ Number of services in stage i, i = 1…N.

The CD/CV Construction Manager creates an architectural pattern using ETL information (see Fig. 5) to materialize the workflow (W).

The applications and services of each stage are encapsulated into containers for organizations to clone and deploy stages on IT infrastructure.

For simplicity, we can say that the CD/CV schemes include two parts: the CD focused on stages of the workflow, and the CV focused on nodes of the blockchain network. For this paper, in the CV part of the schemes, the number of organizations participating in a workflow corresponds to the number of stages of that workflow. Nevertheless, the CD schemes can create multiple associations (e.g., multiple stages compose one organization or multiple organizations compose one stage).

The CV part includes the following components: *(1)* A single order manager node that implements the consensus mechanism necessary to establish consistency in transactions; *(2)* A CA node for each organization to manage the cryptographic material for each stage to register transaction; and *(3)* The number of peer nodes N_{peers} defined for *P2P* of the blockchain. This parameter is assumed to be a variable for experimental purposes and of course in production depending on the IT-resources.

In this sense, the end-user delivers information about the CV scheme, such as the number of organizations and the number of peer nodes per organization in the verifiability network. These parameters are defined in the ETL interconnection model as follows:

$$PxO = N_{peers}$$

$$Orgs = \{Org_1, Org_2, \ldots, Org_N\}$$

where N_{peers} is the number of peer nodes for each organization, and $Orgs$ defines participating organizations. This research assumed that there is a correspondence between the number of workflow stages and the number of participating organizations.

The total number of nodes in the verifiability network is represented by the following notation:

$$Nodes = \{Peers_{Org1}, Peers_{Org2}, \ldots, Peers_{OrgN}, Nodes_{CAs}, N_{or}\}$$

where:

$$Peers_{Org_i} = \{peer0_{Org_i}, peer1_{Org_i}, ..., peerN_{peers_{Org_i}}\} \in Orgi \text{ con } i \in 1, 2, .., N$$

$$Nodes_{CAs} = \{ca_{Org_1}, ca_{Org_2}, ..., ca_{Org_N}\}$$

$$N_{or} = \{orderer\} \rightarrow \text{Node to establish consensus.}$$

By default, an *administrator* for each organization is defined for each CD/CV scheme. Administrators are responsible for adding new participants to the CD/CV schemes and to associate stages to organizations.

Configuration for the CD/CV scheme deployment. The Construction Manager creates configuration files for the deployment of both components: continuous delivery (CD scheme part of the workflows) and the continuous verifiability network (CV scheme part

of the blockchain) on a given infrastructure. These files create the paths of Extraction (from data sources), Transformation (execution of applications and services), and Loading (sink) to a shared volume in the virtual container of the stages (CD) and nodes (CV). The definition of paths in each component of the CD/CV schemes and enables the Construction Manager to materialize the CD/CV schemes to realize workflows (managed by CD scheme part) and the corresponding blockchain network (managed by CV scheme part).

The following paths are added to the continuous verifiability (CV scheme part): *(1)* Path to Hyperledger Fabric, for the nodes to get the binaries required to create the cryptographic material (certificate, and public/private keys) as well as the binaries required to generate the building artifacts of the blockchain (peer communication channel, genesis block, and membership manager); and *(2)* Path to business module, which points to the configuration file containing the business model associated with the ETL model of all the stages of the workflow.

3.2.1.2. Using CD/CV schemes during executing time. This section describes three subcomponents such as *Orchestrators*, *Launchers*, and *Choreographers*, which are used by CD/CV Construction Manager to start the execution of the deployed stages of workflows (CD virtual containers) and nodes of blockchain network (CV virtual containers). These components ensure the establishment of controls over the registration of the transactions performed in stages of workflows in the blockchain nodes.

The *Orchestrators* enforce the ETL interconnection by creating configuration files that are used by *Launchers* to initiate all the deployed stages of workflows (CD scheme part) and blockchain nodes (CV scheme part). Finally, the *Choreographers* start the injection of workload (data for stages and transactions for nodes), and the execution of the applications of the stages and the connectors of these stages with the blockchain nodes.

In detail, the Orchestrators create the configuration files and artifacts needed to deploy the CD/CV schemes for workflows. Two Orchestrators were developed: the CD Orchestrator, in charge of continuous delivery, and the CV Orchestrator, designated for continuous verification of transactions.

The CD Orchestrator creates the necessary settings to synchronize the applications to communicate with each other. To do this, it receives the representation of the network from the Manager (i.e., a *DAG*) and sends orders to CD Launcher for coupling the stages following the *DAG* by configuring input and output ports in a given infrastructure for each stage.

The CV Orchestrator creates the logical artifacts needed to create the verifiability network for the workflow defined in the CD scheme. These artifacts implement the business model of each pair of stages in the verifiability network (e.g., the definition of participants and assets) for Launchers to create the smart contracts and to prepare the cryptographic components (keys for connectors in stages to make transaction registration) as well as to couple the *P2P* network with the CA (certification authority).

The CD Choreographs starts the injection over the stages following the *DAG* over the deployed virtual containers (stages) to guarantee the correct and valid coupling and interconnection of the applications deployed on the virtual containers of the workflow stages.

The CV Choreographs verify that all the connectors of the stages can reach the blockchain nodes, that the cryptographic components (keys for accessing to the blockchain network) are valid, that the smart contracts have been successfully created, and the transactions of the applications executed at the stages (workflows/pipelines) can be registered under the control of the contracts.

3.2.2. Operation manager

This module is responsible for managing the proper functioning of the IoT dataflow and the verifiability network. The Operation Manager must ensure the continuous delivery of the data in the workflow and also the continuous verification of the transactions performed. Fig. 6 illustrates the Management Framework once it is in operation.

Fig. 6 shows an example of a workflow including N stages containing n Virtual CD containers that will process a data source and extract, from it, knowledge, or information useful for decision-making in organizations or interested entities. At this point, the continuous verifiability component starts recording the transactions (Tx) performed by the stages of the IoT dataflow in the verifiability network. Summary functions (or commonly known as *hash* functions). The transactions are registered by using signature (hashes), which produces an identification for a content. A transaction includes the hashes of contents incoming to the stages and the resultant contents transformed by the applications executed at a stage, as well as the paths used to extract, transform, and loading contents.

This mechanism ensures that any alteration in the input contents and the resultant contents. This means, the contracts verify the hashes of the assets before and after to be transformed by each stage in a IoT dataflow; as a result, the end-users can verify the integrity of the data at each stage of the workflow. Moreover, the chaining of the ETL paths and hashes allows end-users to detect/prevent from stages processing unexpected contents by querying to the blockchain network.

3.3. Integration of GM-CD/CV with scientific workflows

The integration of GM-CD/CV to existent scientific workflows engines for processing data from IoT is feasible.

1. Workflow Stages: GM-CD/CV requires that each workflow stage is in the form of an executable service or application. This restriction is because the initial design of the Global Manager CD/CV delivers as a response from a i stage to a $i+1$ stage, the output of the server where the service is exposed (API REST response), or the logs of an executable application.
2. Declarative process: GM-CD/CV assumes that the user declares the ETL of each application.
3. Data source availability: GM-CD/CV requires the location of a data source (set of tasks to be processed) to start the continuous delivery process.

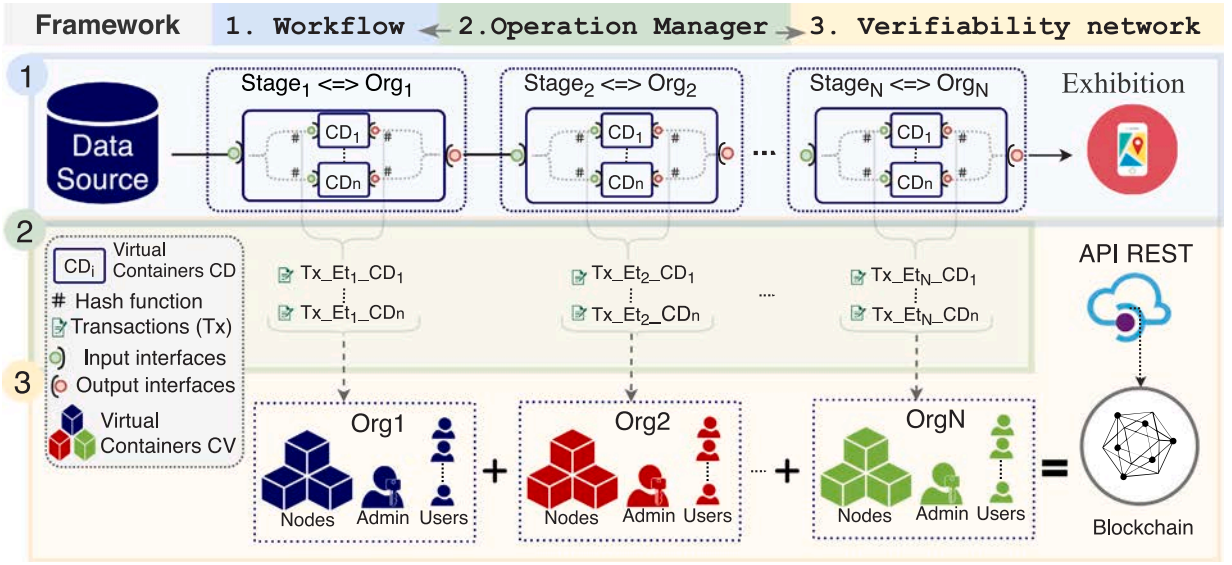

Fig. 6. Management framework CD/CV in operation.

4. Data Exchange: GM-CD/CV requires that the results exchanged between stages are packaged in a standard format such as JSON to ensure continuous delivery. We assumed that each organization that performs the exchange known the data placed in the exchange file.
5. Verifiability Network: GM-CD/CV already incorporates the blockchain manager. If the user needs to change this blockchain manager, he will have to adjust the GM-CD/CV deployment engine (i.e., use Ethereum or another mechanism to use cryptocurrency in transactions).

4. Case study based on IoT data, interest points and user mobility trajectories

For this research work, we conduct an experimental evaluation in the form of a case study based on the processing of IoT data about mobility, interest points, and trajectories of user.

This section thus describes the CD/CV schemes built for creating a traceable and verifiable workflow for the case study, which consists of user mobility data captured from GPS devices.

4.1. Workflow for discovering interest points and trajectories of user

The mobility data are acquired and processed by software that implements the algorithm proposed by Montoliu, Blom, and Gatica-Perez (2013). The algorithm allows the extraction of points of interest (POIs) by analyzing the location points of a user. We include this algorithm in a workflow, which implements stages such as acquisition, preprocessing, processing, storage, and exhibition.

4.1.1. Acquisition service and data source features

An Acquisition service extracts records from Geolife Database (Zheng, Fu, Xie, Ma, & Li, 2011), which contains information on the paths of 182 users collected over five years in Beijing, China. This data source includes 18,670 trajectories of an equal number of users corresponding to 1,292,951 kilometers. The acquisition service collects data such as latitude, longitude, and timestamp of each point collected by different GPS devices. The Geolife dataset is a popular dataset that has been used in many SOTA works for predicting future locations (Mathew, Raposo, & Martins, 2012), multimodal locomotion with mobile devices (Gjoreski et al., 2017), for classification of transport modes using ensembles (Xiao, Wang, Fu, & Wu, 2017), or for anonymization using machine learning (Shaham, Ding, Liu, Lin, & Li, 2019).

4.1.2. Operation of the workflow designed

Fig. 7 depicts a workflow for the extraction of points of interest (POIs) from the mobility data of users.

The first stage (St_1) of the workflow prepares the mobility data extracted by the acquisition service. This stage includes data preparation procedures such as removing anomalous values, unifying formats, and transforms them into a structured form of the data, such as JSON files.

The processing stage (St_2) includes tasks for data processing, such as the extraction of POIs and the inference of new data. Given a list of location points of different users, this stage extracts points of interest (POIs) for each user. Subsequently, from the extracted

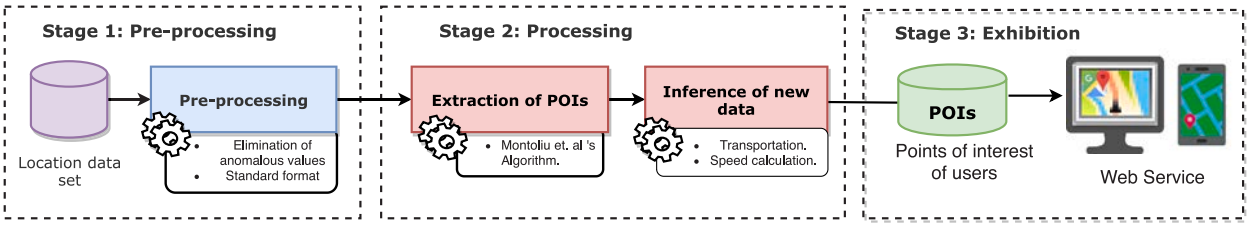

Fig. 7. Design of the workflow for mobility case study.

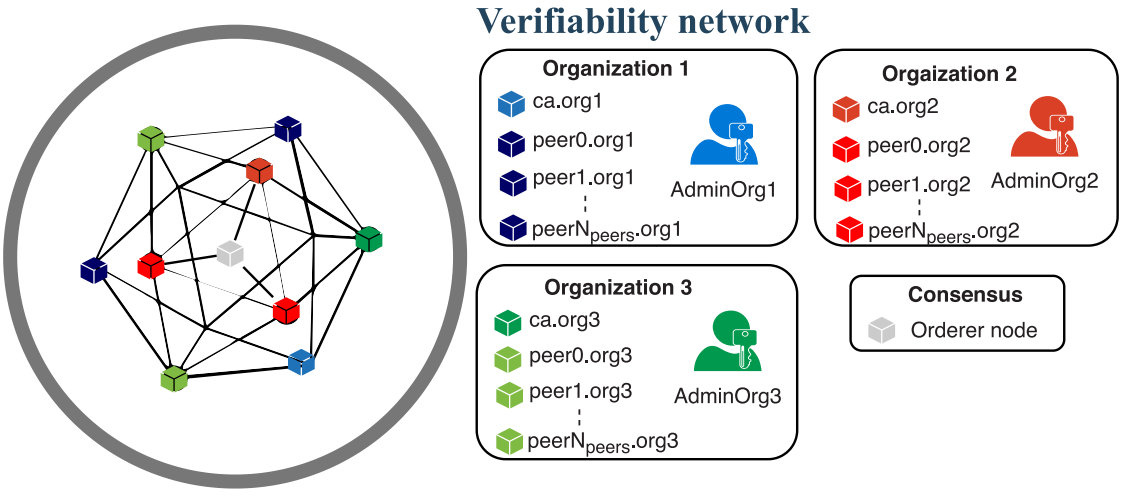

Fig. 8. Verifiability network for the mobility case study.

POIs, relevant information about the user's mobility is inferred, such as the mode of transportation (for example walking, by bus, or by airplane) between the POIs and the mobility speed of these users.

We store the calculated POIs in a database deployed on the third stage (St_3). Finally, an exhibition module based on a web service shows the POIs obtained per user and creates a route in a map created by GIS-geoportal. The end-users of this visualization web service can define search criteria (e.g., a given user or spatio-temporal parameters). The workflow described composes the CD/CV system displayed by the Global Manager (see Fig. 9).

It is important to note that the signature (hash) of each extraction, transformation, and load of these sensitive data performed by each stage (in the CD scheme part) are registered in the traceability network (CV scheme part) in automatic and transparent manners.

4.1.3. CD/CV schemes for building a traceable and verifiable workflow

The acquisition service invokes the workflow (W) defined by the following expressions:

$$W = \{St_1, St_2, St_3\}, \text{where: } St_1 \rightarrow \text{Preprocessing stage, } St_2 \rightarrow \text{Processing stage, and } St_3 \rightarrow \text{Exhibition stage, } NuSt_1 \rightarrow \text{N° of preprocessing services, } NuSt_2 \rightarrow \text{N° of processing services, } NuSt_3 \rightarrow \text{N° of exhibition services.}$$

The CD/CV Global Manager presents the above notation in the form of boxes to the end-users to retrieve from them the ETL model and the number of services to be deployed at each stage. The Global Manager uses this information to create an architectural pattern for the efficient analysis of GPS data.

The parameters of the number of organizations and the number of peer nodes per organization for this case study are defined as follows in the continuous verifiability component (CV scheme part):

$$PxO = N_{peers} \text{ and } Orgs = \{Org_1, Org_2, Org_3\}$$

where N_{peers} determines the number of peers for each organization, and $Orgs$ defines participating organizations. In this study, $Orgs$ is equal to three, which are the number of stages in the workflow.

The result of the established definitions automatically creates a network of verifiability that is displayed below (see Fig. 8).

By default, the CV scheme part is created by the *administrators* defined by each organization participating in a workflow.

When both CD (workflow) and CV (blockchain) have been established by administrators, the Orchestrators configure: the path to the Geolife database, the ETL routes for each service (pre-processing, processing, and exhibition) deployed at each stage, and the

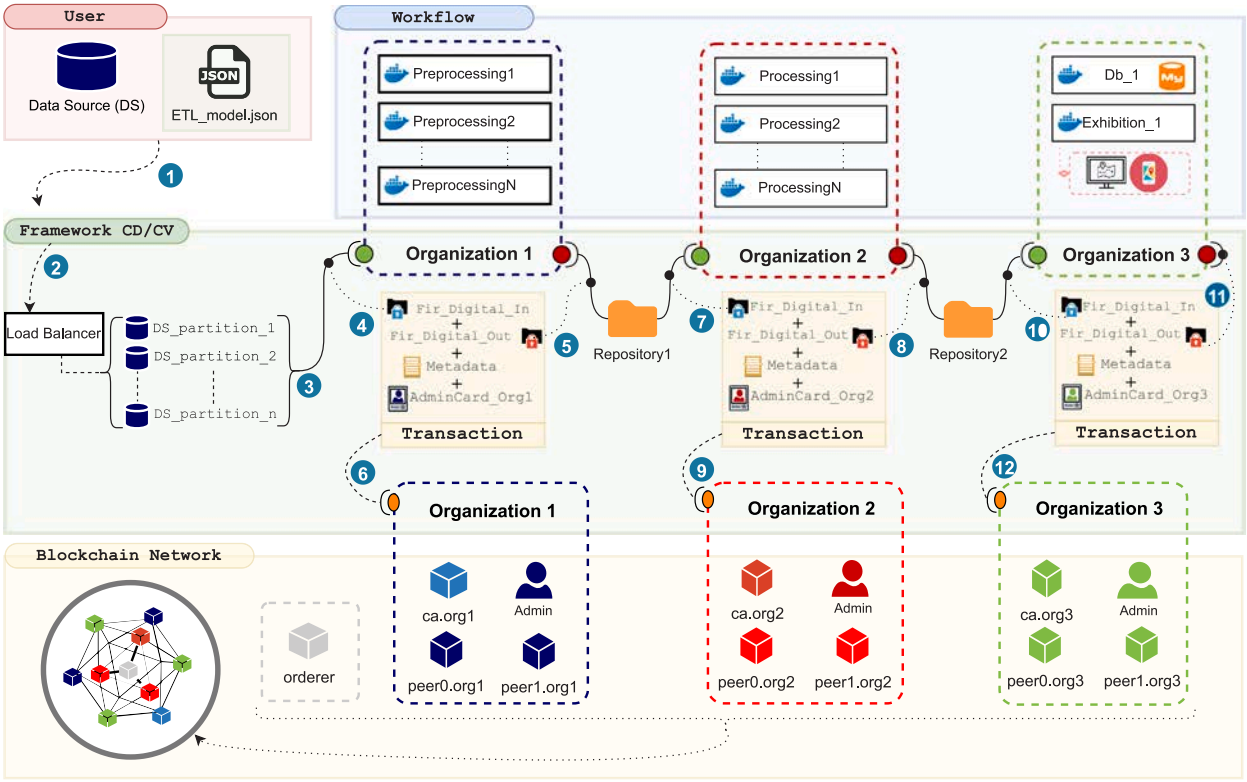

Fig. 9. Operation of the CD/CV system for the user mobility case study.

path that points to the Docker configuration for each virtual container image of the services considered. The Launchers deploy the virtual containers considered in CD/CV schemes. And the Choreographers establish the execution of the first stage of the workflow. This means that when end-users initiate the acquisition stage, the processing starts, and this culminates in the visualization, for end-users, of a map including the point of interests detected through the IoT dataflow.

At the same time, the Construction Manager with the CV scheme creates the business model and smart contracts for the stages of the workflow. As well as it creates the continuous verifiability (CV) network by using the Hyperledger Fabric. Additionally, it verifies the availability of this network for the stages of the workflow.

4.1.4. Deployment and implementation details of the workflow created by CD/CV

The stages of the workflow were encapsulated into virtual containers using the Docker platform. In this case study, the CD/CV manager prototype was deployed on an Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20 GHz with 64 GB of memory, two hard drives of 2.7 TB each one, and 12 cores.

The virtual containers also include software such as an operating system with installation type “Compute Node” version Linux CentOS 7 x64, Cloudera Manager, Docker, and Docker Compose higher than 17.06.2-ce, cURL Latest, Go 1.11.x, Python 2.7.x, Node.js Runtime, and NPM 8.9.x, and Hyperledger Fabric 1.2.

5. Experimental evaluation methodology

To evaluate the performance of the CD/CV schemes built to conduct the previously described case study, we performed an exploratory evaluation phase to identify the impact of critical parameters on CD and CV components performance as well as a direct performance comparison of CD/CV schemes with a state-of-the-art solution (DagOnStar Montella, Di Luccio et al., 2018).

We start defining a set of solutions, including multiple configurations of CD/CV, and a workflow engine available and studied in the literature.

5.1. Studied solutions

We studied *DagOnStar* (Montella, Di Luccio et al., 2018), a related solution found in the literature, which is a workflow engine for scientific and environmental processing. It is also studied the solution proposed in this research (**G**lobal**M**anager: GM-CD/CV), which not only creates the workflow but also the network of verifiability. We also evaluated a version of the workflow only using

the continuous delivery scheme (*GM-CD*). This evaluation compares the performance of GM-CD/CV with that produced by GM-CD to observe the overhead generated by the CV scheme part (registering transactions in the blockchain). In order to measure the impact of parallelism patterns on the reduction and even elimination of CV overhead, we performed a comparison of GM-CD/CV with DagOnStar, which also considers implicit parallelism at the stage level.

5.2. Metrics

The following metrics were defined and captured in the execution phases (preparation, deployment, and operation) of the workflow by the three solutions studied:

1. Response Time (RT): This metric represents the time that elapses since the user made requests to the IoT dataflow until user gets a response. For this research work, the response time is the time that the user applies a load to the workflow until user gets the results.
2. Deployment Time (DT): It represents the time required to deploy and have in service all the processing units or virtual containers. This time also includes runtime (Ru), which represents the time interval in which a deployment script runs on the operating system.
3. Overhead (Ov_d): This metric represents the extra time that a specific operation or process may require if the system adds some functionality. In this case, we seek to measure the overhead produced by adding continuous verification functionality to a traditional workflow such as DagOnStar or on our GM-CD proposal.
4. Throughput(Th): It corresponds to the volume of data per unit of time processed through the IoT dataflow.

5.3. Exploratory evaluation phase

In the exploratory experimental evaluation, the parameters of both the continuous delivery (CD) and continuous verifiability (CV) components are systematically varied.

The parameter variation is defined to explore the performance of the CD/CV schemes for different workloads. This allowed us to determine the set of parameters of the proposed solution that produce the best performance. This phase considers a heterogeneous distribution of data for the number of mobility routes of each user.

5.3.1. Experimental variation — Continuous delivery

The following parameters were varied in the exploratory phase of the CD scheme part: *(a)* number of services per phase or parallelism $\{1, 2, 4, 6, 8, 10, 12, 24\}$; *(b)* volume of data to be processed $\{1, 10, 100\}$ records; and *(c)* seed of the random number generator $\{1, 2\}$.

The Operation Manager selects a subset of data or users using two different seeds (each seed value generates a different subset of data). Considering all the possible combinations of the parameters (experimental *Variety* and *Volume*), there are a total of 24 configurations to measure the performance of the IoT dataflow using the CD scheme of GM-CD solution, with the same subset of data selected for each variation of data D_i for each S_i service. We ensured that each solution processes the same subset of source data. This subset of data is selected by setting a seed value for the random number generator and allows for replication of the experiments.

We performed two experiments to apply different workloads (a subset of data) to the workflow, E_1 and E_2 (with seed 1 and 2 respectively), each of them with 24 configurations, having a total of 48 evaluations.

We have executed each experiment 31 times. Results presented in the paper show the average and the standard deviation of the 31 executions. In most cases, the confidence interval is 5%.

5.3.2. Experimental variation — Continuous verifiability

For the CV scheme part, we considered varying the number of *Peers* in the blockchain for each organization: {1,2,4,6}. In this sense, for the GM-CD/CV solution, the total of configurations to be evaluated is given by the parameters of *Variety*, *Volume*, *Seed* (defined in the continuous delivery scheme), and the variety of Peers per organization.

This means, considering these four parameters, we performed 96 configurations of GM-CD/CV solution with the same workload for all data variations (D_1, D_{10}, and D_{100}) for each seed. In these configurations, D_i represents the amount (i) of data or users to be processed in the IoT dataflow.

In a similar fashion used for GM-CD configurations, we performed two experiments for GM-CD/CV, E_1 and E_2, each of them with the 96 configurations using the same seeds (1,2) selected for the GM-CD solution to process the same data but this time considering records in a verifiability network.

The result is a total of 192 evaluations to evaluate the GM-CD/CV solution, considering the component of continuous delivery and continuous verifiability in the exploratory stage of the experimental variation. In the comparative stage with the DagOnStar solution, we adjusted the parameters for GM-CD and GM-CD/CV according to the results presented in the exploratory phase. Section 5.4.2 describes this comparative stage.

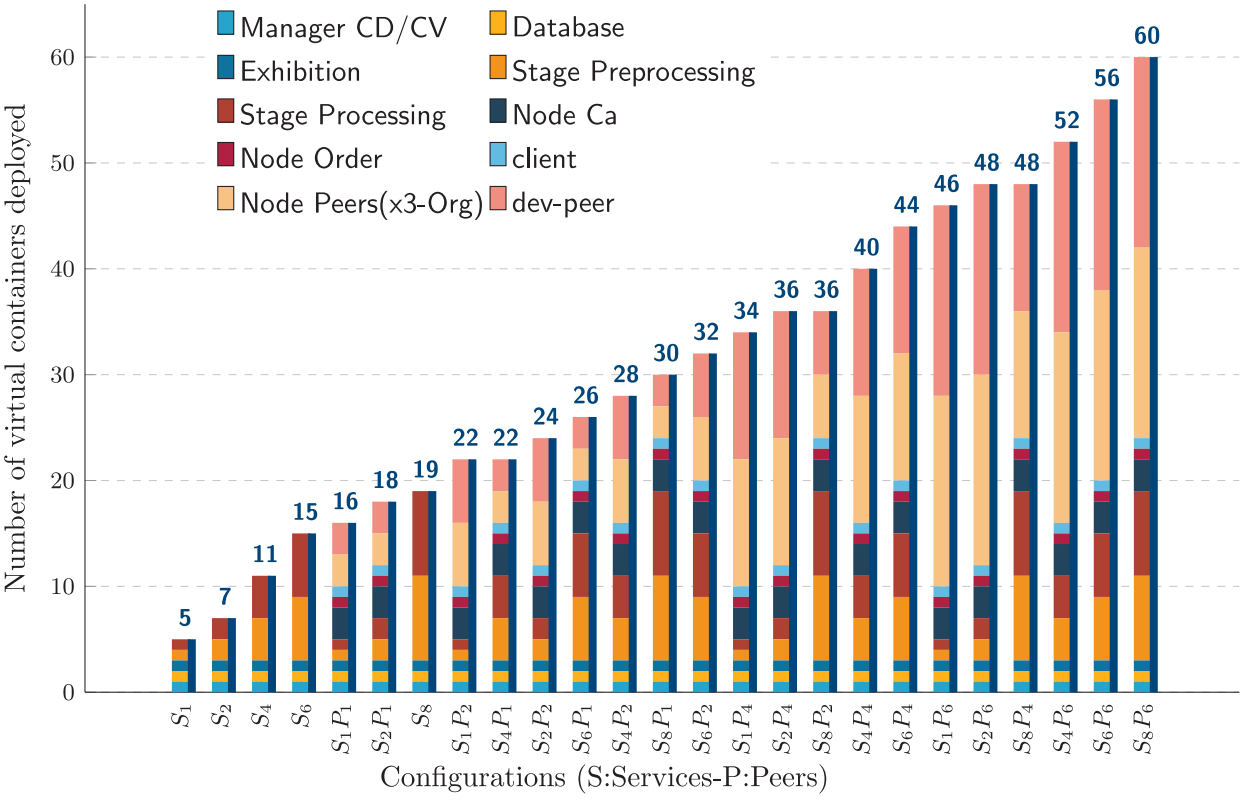

Fig. 10. The number of CD/CV Virtual Containers deployed for each configuration for the mobility scenario — Exploratory stage.

5.3.3. Results and discussion of the studied solutions

The first analysis carried out in the exploratory phase consisted of the number of virtual CD/CV containers to be deployed for processing the mobility data for the CD scheme part (*services per stage*) and CV (*peer nodes per organization*).

Fig. 10 shows the number of different types of containers deployed per each notation S_i-P_j, where i is the number of services per stage, and j is the number of peers per organization of a given configuration defined in the exploratory phase.

For this mobility case study, the CD containers that correspond to the workflow are *Database*, *Exhibition*, *Preprocessing Stage* and *Processing Stage*. The CV containers in the verifiability network are *Ca Nodes*, *Order Nodes*, *Client Nodes*, *Peer Nodes*, and *dev-Peer Nodes*. The container CD/CV Manager is the solution proposed in this work of research.

As an example, Fig. 10 shows that the notation S_8P_6 deploys 60 virtual containers on the cloud. In this example, eight correspond to the pre-processing stage and eight to the processing stage, given the notation S_8. For practical purposes for the development of this research work, all the configurations included one virtual container for the database and another one for the exhibition module. These 18 containers mentioned above are included in the CD scheme (for continuous delivery in the workflow). In the same example of S_8P_6 notation, the P_6 indicates that each organization (three for this case study) has deployed 6 peer nodes in the verifiability network, (i.e., the three organizations have deployed 18 virtual containers implementing peer nodes of a private blockchain). As previously established, the verifiability network includes two more containers for each organization: the first one implementing the order manager node and the last one implementing the certification authority (CA) node, which results in 6 virtual containers. A virtual container called *client* is deployed by each organization to perform operations such as creating the channel through which the peer nodes communicate, joining the peer nodes to this channel, among others. Also, for each Peer Node deployed, the Manager deploys another virtual container (dev-peer) to configure the business network in the Peer Node. In this sense, for the verifiability network, there would be 41 virtual containers CV. The last virtual container of the S_8P_6 configuration is the *CD/CV Manager*, which includes Orchestrator, Launcher, and Choreograph. It is important to note that both CD and CV virtual containers are automatically deployed by *CD/CV Manager* on the cloud and interconnected by using *CD/CV* scheme without the intervention of the end-users, which only provides the CD/CV manager with ETL information and business logic model (based on a DAG).

The deployment of each virtual container, either CD or CV on a given infrastructure, implies the consumption of memory, disk space, and other computational resources. Fig. 10 shows that, as expected, those configurations with a higher number of peers (e.g., P_4 and P_6) consume more resources than those using few peers.

The notation P_0 is equivalent to the solution *GM-CD* as it does not consider peer nodes per organization. The configurations with the parameters P_1, P_2, P_4 and P_6 basically are used in *GM-CD/CV* configuration by varying the number of peer nodes per organization.

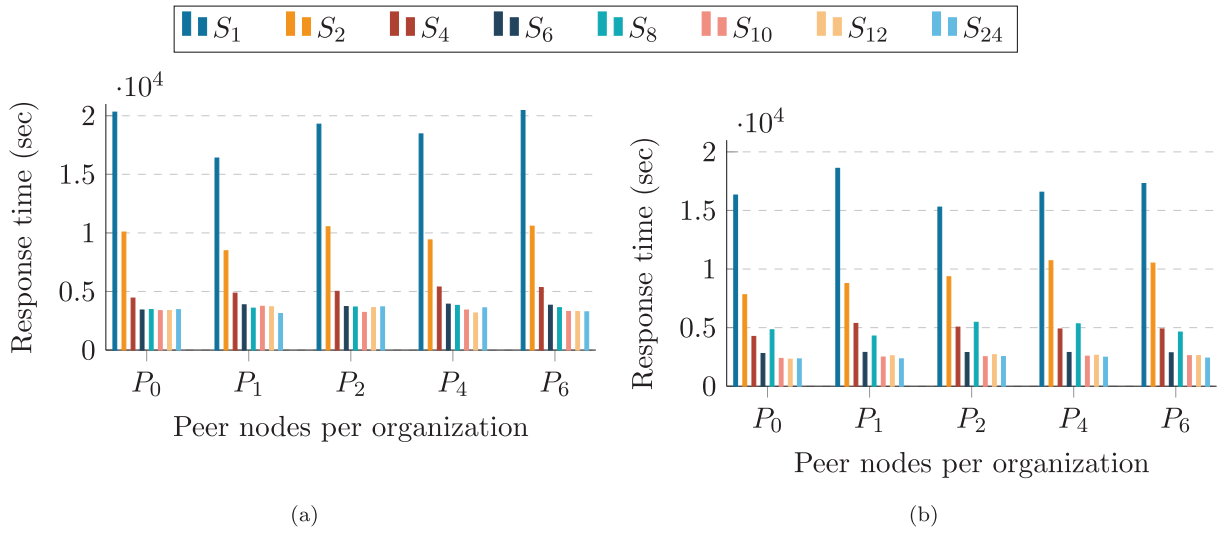

Fig. 11. Response times of the configurations in the exploratory stage.Workload of 100 selected user data chosen by seed 1 (graph a) and seed 2 (graph b).

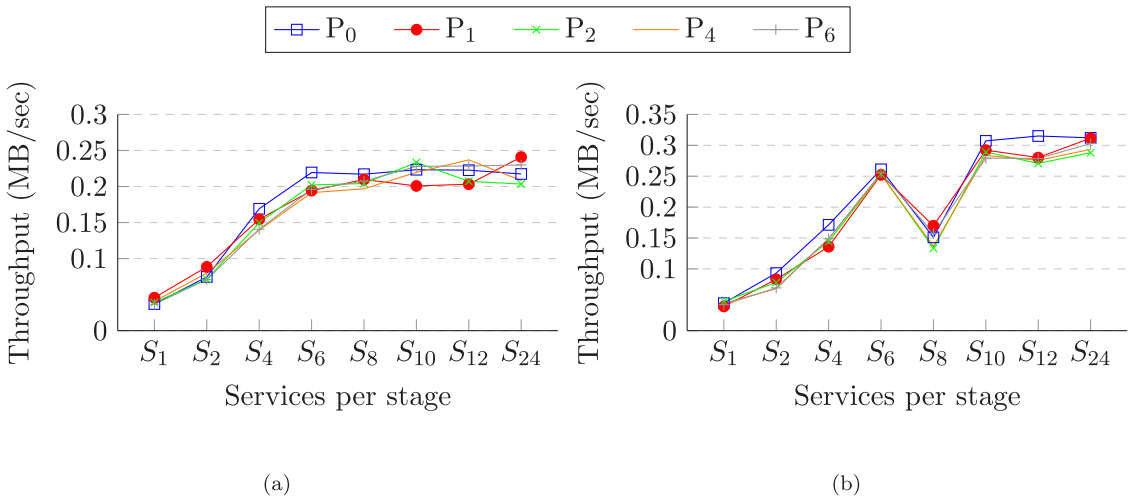

Fig. 12. Throughput of configurations defined in the exploratory stage. Workload of 100 selected user data chosen by seed 1 (graph a) and seed 2 (graph b).

Figs. 11(a) and 11(b) show that, independently of the workload or the number of peer nodes per organization, the configurations with the parameters S_{10}, S_{12} and S_{24} produce the shortest response times in comparison with the other variations.

The performance when varying the number of peer nodes per organization (P_1, P_2, P_4, and P_6) could be affected by latency in the case of organizations selecting multiple cloud resources to create their verifiability network.

Figs. 12(a) and 12(b) show the throughput of each *GM-CD* and *GM-CD/CV* configurations processing data of 100 selected user trajectories with seed 1 and 2 respectively.

As it can be seen in Figs. 12(a) and 12(b), the higher performance is achieved when parallelism schemes are applied to the CD/CV schemes (see configurations S_2 to S_{24}) as these configurations process data in concurrent manner using an implicit parallelism model embedded into the CD virtual containers, which also includes a load balancing algorithm (see P_1, $P_2, \ldots, P_6$ when each stage using four services). This model compensates and even eliminates the costs of the CV components (See S_{2-6} for all P_1, $P_2, \ldots, P_6$ CV components), which making feasible to create a traceable and verifiable critical workflows.

The results presented in this exploratory phase were analyzed to determine the parameters producing the best performance. In this context, it was observed that the choosing of the number of services per stage should be defined according to the number of cores available in the infrastructure where workflows based on CD/CV schemes were deployed on.

Specifically, in the infrastructure used in this exploratory phase, 4 and 6 peer nodes for each organization resulted in an overuse of resources (i.e., memory and disk) for CD/CV schemes, which directly affected the response time. Although the configuration of one peer node per organization produced acceptable response times, it might not be adequate as the organizations would be unable to access the network records of verifiability (blockchain) in events of node failures.

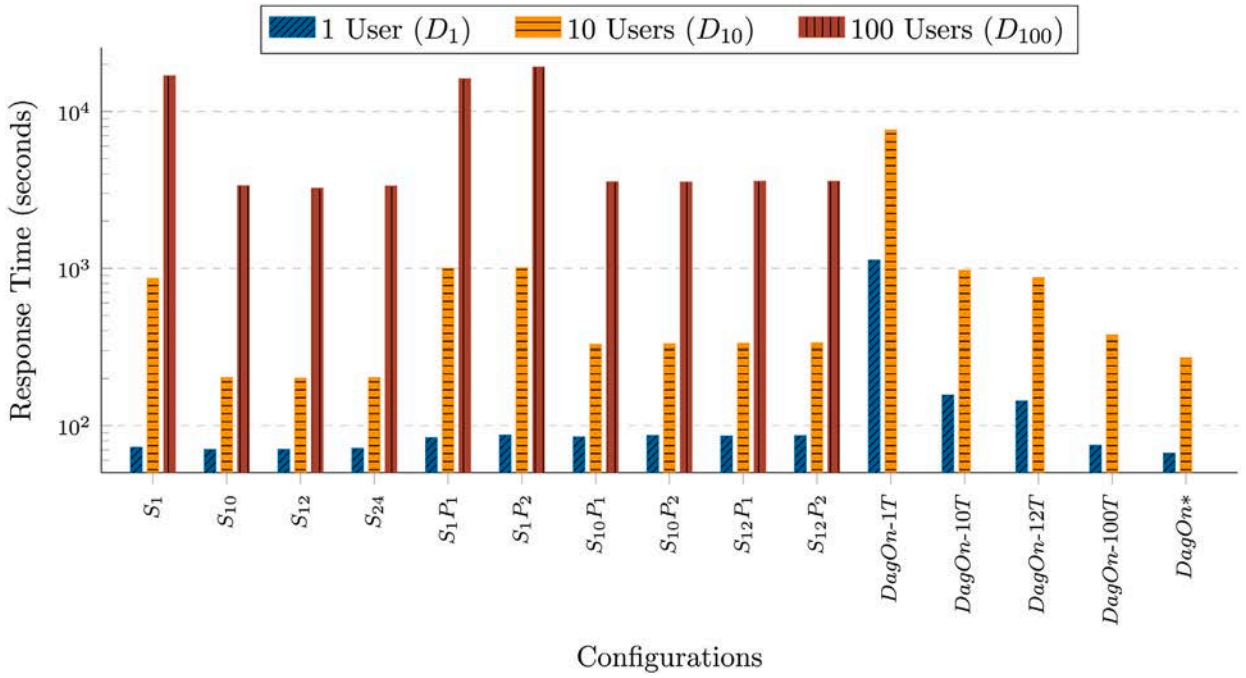

Fig. 13. Response time all solutions. **GM-CD** (S_1, S_{10}, S_{12}, S_{24}), **GM-CD/CV** (S_1P_1, S_1P_2, $S_{10}P_1$, $S_{10}P_2$, $S_{12}P_1$, $S_{12}P_2$), **DagOnStar** ($DagOnStar - 1T$, $DagOnStar - 10T$, $DagOnStar - 12T$, $DagOnStar - 100T$, $DagOn*$).

5.4. Direct performance comparison phase

With the results obtained in the first phase, adjustments were made to the given parameters to carry out, in a second phase, the direct performance comparison between a state-of-the-art solution (DagOnStar) and the solutions proposed in this research work (GM-CD, GM-CD/CV).

During this phase, the different solutions are configured in a comparable way (e.g., processing threads), the different response times are determined applying different workloads to both sequential and parallel versions. Additionally, the overhead/gain of a given solution is analyzed with respect to the remaining configurations.

5.4.1. Results and discussion of the solutions studied

The results of the exploratory phase allowed us to identify criteria to chose CD/CV operation parameters. For the continuous delivery component, the parameters to consider for the comparative stage were: 10, 12 and 24 services per stage (S_{10}, S_{12}, and S_{24}) with three different workloads ($D = 1, 10, 100$). Additionally, we included $S = 1$ to consider a sequential solution, generating a total of 12 configurations to evaluate.

5.4.1.1. Analyzing parameter criteria for GM-CD solution. Fig. 13 shows that S_{12} was the best selection of the number of services per stage for the mobility scenario workflow in GM-CD configuration as it processed the highest load in the shortest time. In a similar way to the exploratory phase, the best response times were obtained by the configurations using as many services as available cores in the infrastructure (12 cores). When configurations deploy more threads than available cores (e.g., S_{24}), the system ending up creating a queuing of tasks in the available cores of the infrastructure, which reduces the improvement in response times when deploying more services than available cores.

5.4.1.2. Analyzing parameter criteria for GM-CD/CV solution. For the CV components, the number of peer nodes $P = 1, 2$, produced the best performance results in the exploratory phase. We discarded the S_{24} configurations, as deploying more concurrent services than available cores (12 cores) does not produce performance improvements (−3.46% on average of all evaluations between S_{12} and S_{24}).

It was also observed that, from 10 services, the response times tend to become shorter and configurations do not yield a substantial improvement (see performance of configurations as S_{1-10}).

Fig. 13 shows that regardless of the number of peers selected (P_1 or P_2), configurations with S_{10} with a load of 100 data (D_{100}) produces similar average response times (0.32%).

5.4.1.3. Analyzing parameter criteria for DagOnStar solution. DagOnStar also performs task parallelism using threads. For the experimentation, we considered the number of threads $T = \{1, 10, 12, 100, *\}$ with the same workloads already evaluated (data of users $D = \{1, 10, 100\}$). The parameter $T = *$ means that DagOnStar manages all the available resources of the infrastructure, creating a thread for each input task to be processed. For this purpose, DagOnStar incorporates a component known as SLURM (Yoo et al., 2003), which is a cluster manager for scaling up thousands of processors.

Fig. 13 shows the response times obtained when considering a sequential version of DagOnStar ($DagOn-1T$) and how it improves with the number of threads enabled for processing ($DagOn-10T$, $DagOn-12T$, $DagOn-100T$). The DagOnStar configuration that allows using all the available resources ($DagOn*$) with the SLURM component was the one that obtained the best results, reducing by 96.43% the response times for the sequential version of DagOnStar for a load of 10 data and 94.09% for an of 1 data.

The DagOn* solution only performs continuous delivery (CD) tasks and does not include the verifiability component (blockchain network) as the solution proposed in this paper. This means the performance of DagOnStar is similar to GM-CD performance. The comparative results between these solutions are described in the next section.

The DagOnStar solution only shows results for one and ten workloads (D_1 and D_{10}) because the configurations processing 100 user data was could not be evaluated because of memory issues (threads consuming tens of Gigabytes). In particular, in the creation process of dependencies between tasks, DagOnStar creates ten sub-tasks for each data to be processed. Moreover, this configuration executes each sub-task to ensure continuous data delivery in the workflow. This process resulted in more than 10,000 tasks that the solution loaded into memory during runtime workflow, which producing overflows when DagOnStar processing workloads of 100 users' data in the infrastructure used for this experimental evaluation.

5.4.2. Comparative analysis and discussion of the solutions studied

The overhead produced by GM-CD/CV, and GM-CD (sequential version) configurations is evaluated and discussed in this section. The DagOnStar sequential version presents a high overhead (10x for 1 data and 7x for 10 data) compared to GM-CD sequential configuration.

Fig. 14 shows that GM-CD/CV configurations compared to GM-CD generate mean overheads between 12% and 35%. Additionally, there will be a higher overhead when considering a higher number of peer nodes and the workload increases.

In other words, the higher number of peer nodes in the verifiability network, the longer the time to accepting a transaction in the consensus protocol of the blockchain, which directly impacts the response times perceived by the user in the workflow.

To evaluate the efficacy of the parallelism features of the solutions to handle these overheads produced by the verification component in the workflow performance, we evaluated the parallel solutions of GM-CD/CV and DagOnStar with 10 (S_{10}) and 12 (S_{12}) threads each, which were the best configurations of both solutions.

The four configurations of GM-CD/CV ($S_{10}P_1$, $S_{10}P_2$, $S_{12}P_1$ and $S_{12}P_2$) for 1 user have an overhead between 16.6% and 18.51% compared to the sequential solution GM-CD, while DagOnStar with 10 and 12 threads limit has an overhead between 97.43% and 115.39% with respect to GM-CD. These overhead percentages occur because there is only 1 User to be processed, and multiple threads are becoming unusable. However, when the load increases (D_{10} and D_{100}), the four GM-CD/CV configurations previously mentioned, reduce the overhead between 61.22% and 61.97% for a 10-data load and between 75.07% and 75.39% for a 100-data load with respect to GM-CD, while DagOnStar with 10 threads still has an overhead of 12.39% and 1.24% with 12 threads for 10 data.

Fig. 14 shows that the GM-CD/CV configurations including either 1 or 2 peer nodes with 10 and 12 threads do not produce overhead compared to the traditional sequential GM-CD workflow for 10 and 100 user workloads (D_{10} and D_{100}). The overhead (19.24%) of CV components is constant, but it is only reduced when using parallel patterns. The performance of the workflow (CD/CV) is even improved when using 10 and 12 concurrent services over GM-CD was approximately 61% for 10 data and 75% for 100 data.

When DagOnStar applies 10 and 12 threads, it still has a considerable overhead compared to the sequential GM-CD solution. However, the percentage was considerably reduced to 1.24% with 10 data when using 12 threads.

To find a competitive DagOnStar configuration, it was allowed to DagOnStar to launch as many threads as defined by SLURM (see DagOn*). Fig. 15 shows the response times of each of the studied solutions considering 10 and 12 threads and DagOn*. As it can be seen, DagOn* can significantly improve the performance of the workflow, but it also imposes a high consumption of computational resources producing task queuing, which also producing overflows when processing a large number of concurrent workloads (e.g., DagOn* produced an overflow when processing mobility routes of 100 users).

As it can be seen, for the 10-user-data workload, the GM-CD solutions with 10 and 12 threads obtained the best response times as these configurations do not to deal with the blockchain transactions registration overhead.

As expected, the GM-CD/CV configurations ($S_{10}-P_1$, $S_{10}-P_2$, $S_{12}-P_1$, and $S_{12}-P_2$) produced overhead in comparison with parallel configurations (GM-CD and DagOnStar), which are not registering transactions. Nevertheless, this overhead is reduced when managing high-workloads as the CV can consolidate the registering of transactions, which reduces the overhead produced by CV components in comparison with parallel GM-CD solutions. For instance, the overhead of GM-CD/CV solution was with the $S_{10}-P_2$ parameters (10 services per stage and two peer nodes per organization) was 8.80% compared to the best GM-CD solution, which was with 12 processing threads.

The transaction recording is based on the input and output batch of each of the processing stages of the workflow and not by the user's paths. This coarse-grained strategy allows the execution times of the transactions in the verifiability network not to be dependent on the number of trajectories of a user and to be practically constant, while for the continuous delivery component, the more paths to process, the longer the response time of the processing of these tasks. In this sense, we estimate that the higher the

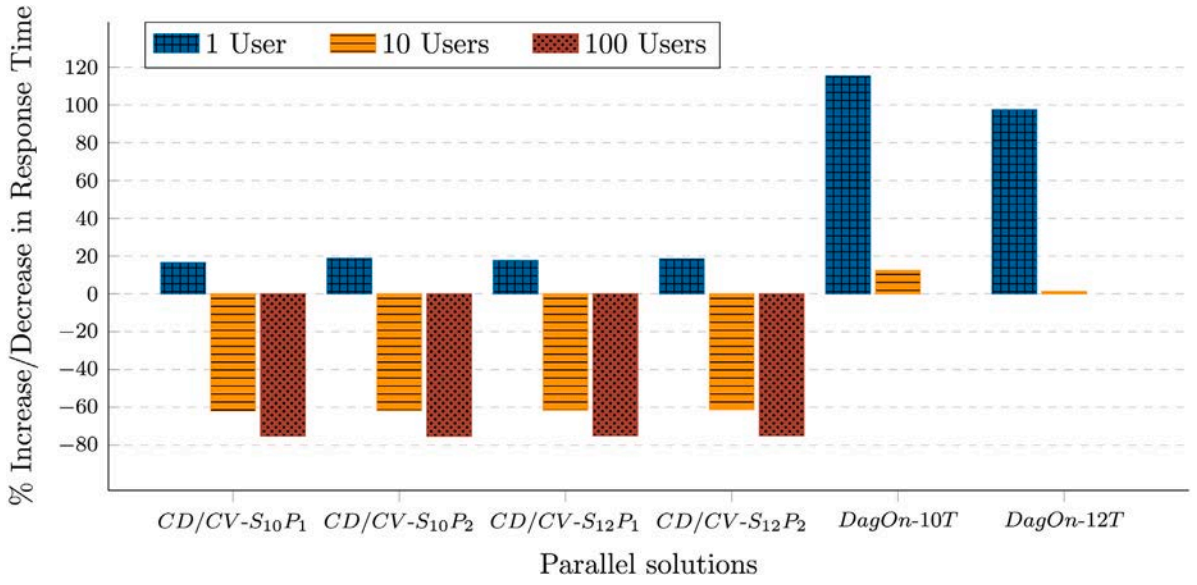

Fig. 14. Comparison of the performance overhead/gain of the studied solutions with $CD - S_1$ configuration.

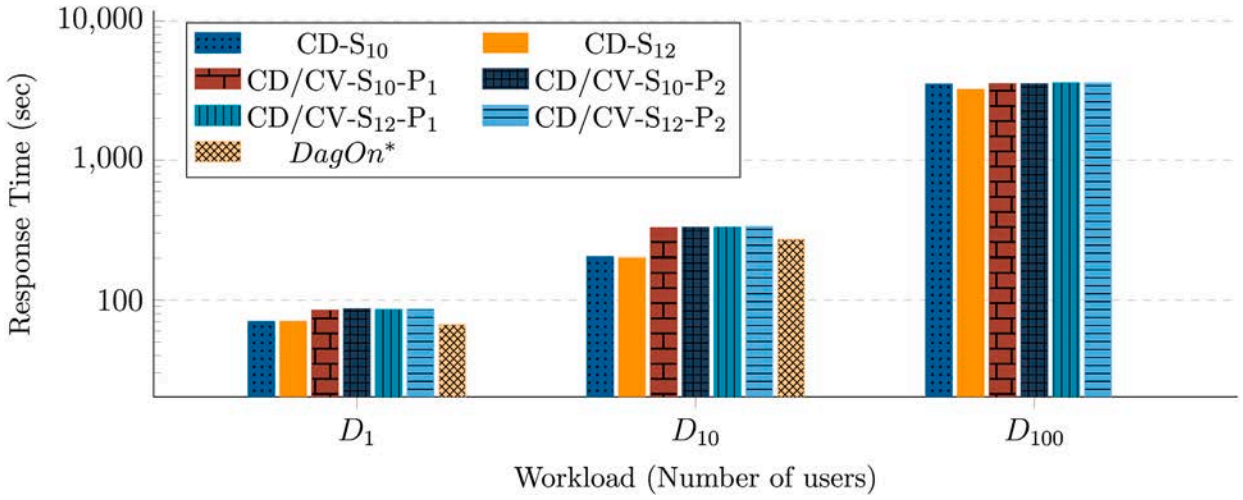

Fig. 15. Response times of the studied solutions considering task parallelism.

workload evaluated in the proposed GM-CD/CV solution, the lower the cost overhead of using a verifiable network compared to the GM-CD solution applying the same load.

We recall that DagOn* achieves better response times than configurations with the verifiability component (CV) for a 10-data load because DagOn* only performs continuous delivery (CD) tasks and not making registration in the blockchain.

The average response times and percentage overhead of all configurations of the studied configurations processing 10-data workload (D_{10}) is compared with the best DagOnStar configuration in Fig. 16.

As it can be seen, the $CD/CV - S_{12}P_2$ is the configuration producing less overhead in comparison with the parallel solution, which is still acceptable in comparison with the best performance configuration.

Table 2 shows a summary of the average response times produced by the studied configurations. It also includes deviation and the deviation % of the metrics for a 10-data load. The deviation percentage of the values obtained for the mean performance average of CD configurations in the range of [1.31 and 2.66%], [1.29 and 3.63%] for CD/CV, but reducing when increasing the number of services.

The average response times and percentage overhead of all configurations of the studied configurations processing 10-data workload (D_{10}) is compared with the best CD configuration in Fig. 17.

As it can be seen, the $CD/CV - S_{12}P_2$ is again the configuration producing less overhead in comparison with the parallel configuration producing the best performance (under 10%). It is important to note that the more the workload to be processed,

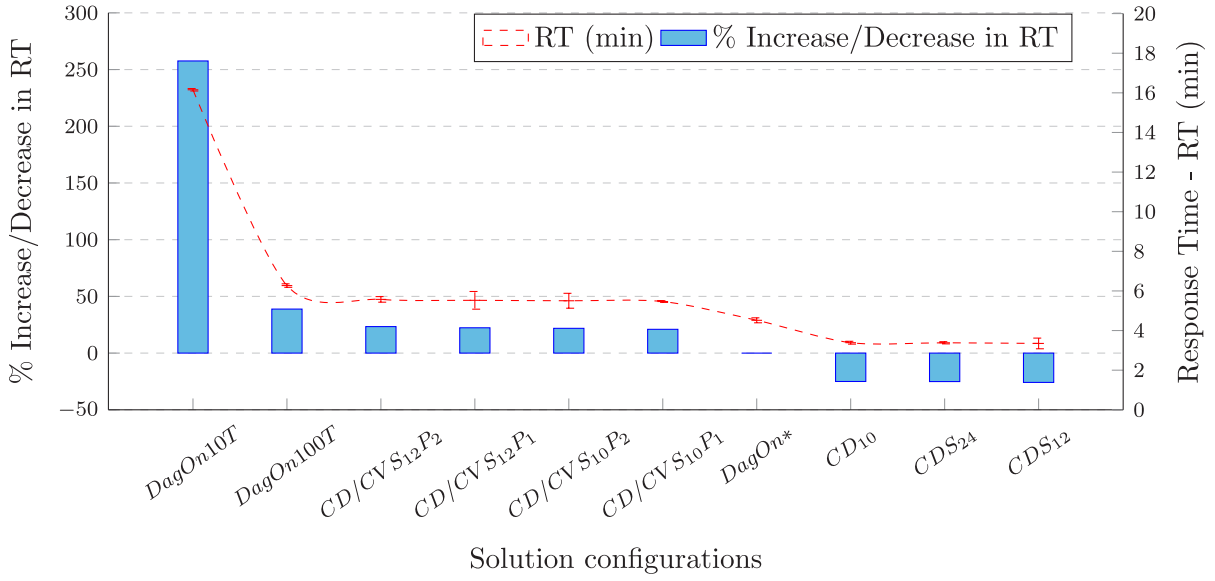

Fig. 16. Overhead/gain percentage of all configurations in comparison with DagOn* for workload of 10 user data.

Table 2
A summary of the mean response times produced by the studied configurations with 10-user-data workload.

	GM-CD				GM-CD/CV						DagOnStar			
	S_1	S_{10}	S_{12}	S_{24}	S_1-P_1	$S_{10}-P_1$	$S_{12}-P_1$	S_1-P_2	$S_{10}-P_2$	$S_{12}-P_2$	T_{10}	T_{12}	T_{100}	$T*$
Average	862.4	203.2	201.0	203.0	1001.7	327.9	331.6	1010.7	330.3	334.4	969.3	873.1	376.3	271.1
Standard deviation	22.91	3.76	2.64	3.34	36.39	11.58	7.54	23.76	5.33	4.32	8.47	37.68	26.81	22.58
% Deviation	2.66	1.85	1.31	1.65	3.63	3.53	2.27	2.35	1.61	1.29	0.87	4.32	7.12	8.33

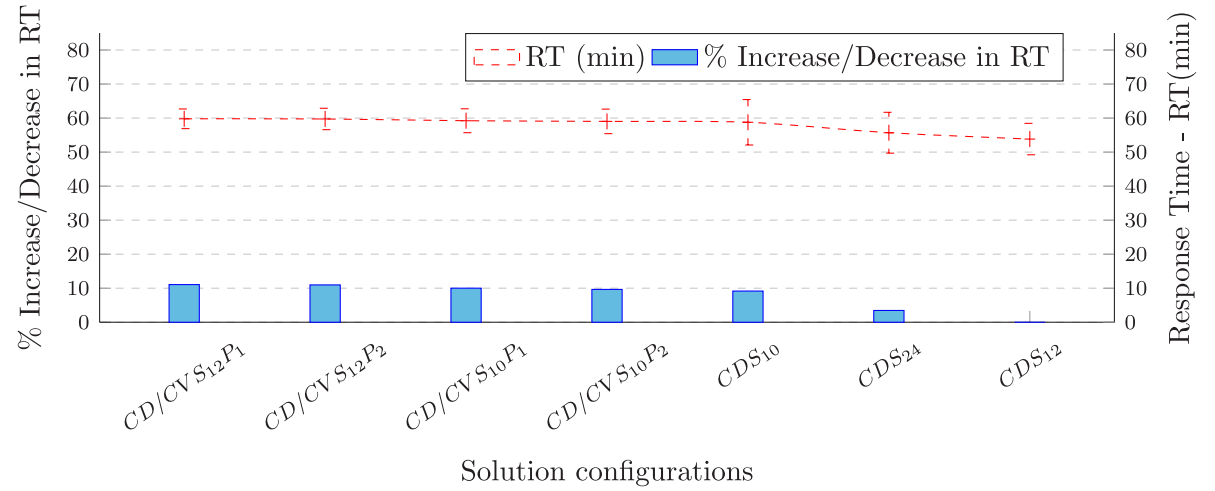

Fig. 17. Overhead percentage comparison of studied configurations with $CD-S_{12}$ for 100 user data.

the more the reduction of the overhead produced by the CV components. The reason for this behavior is due to the coarse and fine-grained strategies of GM-CD/CV. First, the coarse-grained method generates a record to the blockchain for each batch of incoming and outgoing information from each service, this is suitable for scenarios where data generation is constant. As a result, the costs of continuous verifiability are reduced by logging a consolidation of data transactions (a report based on the data flow), which reduces the number of logging processes in the verifiability network. Additionally, the fine-grained strategy is only incorporated into each ETL transformation process (for each transformed file) to generate a digital signature for each piece of data arriving and leaving each stage of the workflows.

Table 3
A summary of the mean response times produced by the studied configurations with 100-user-data workload.

	GM-CD				GM-CD/CV						DagOnStar			
	S_1	S_{10}	S_{12}	S_{24}	$S_1 - P_1$	$S_{10} - P_1$	$S_{12} - P_1$	$S_1 - P_2$	$S_{10} - P_2$	$S_{12} - P_2$	T_{10}	T_{12}	T_{100}	$T*$
Average	16 903.8	3357.3	3229.9	3341.7	16 340.7	3553.2	3587.3	19 196.9	3541.7	3584.3	!	!	!	!
Standard deviation	2929.2	231.2	277.1	360.2	551.3	210.7	173.6	1095.5	216.4	188.6	!	!	!	!
% Deviation	17.33	6.89	8.58	10.78	3.37	5.93	4.84	5.71	6.11	5.26	!	!	!	!

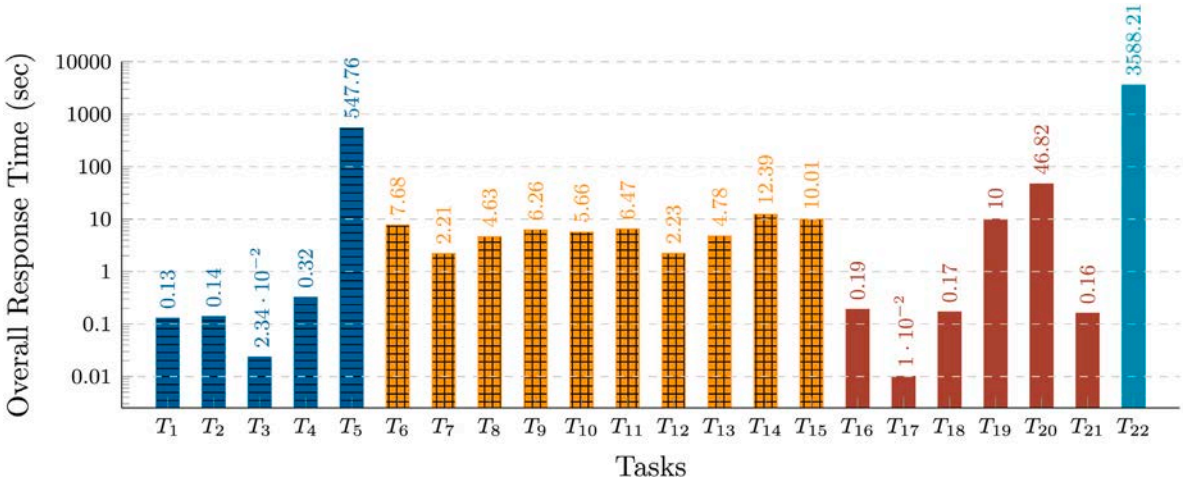

Fig. 18. GM-CD/CV module runtimes. T_1: creates configuration files, T_2: creates certificates, T_3: Inserts keys in configuration files, T_4: creates channel artifacts, T_5: Deploying CV Containers, T_6: Building connection profiles, T_7: Creating Verifiability Network Admin Cards, T_8: Importing Admin Cards, T_9: Installing BNA on HF, T_{10}: Obtaining Admin identities, T_{11}: Starting Business Network in Network Fabric, T_{12}: Creating Administrators per organization, T_{13}: Importing Administrator Cards, T_{14}: Verifying to Verifiability Network, T_{15}: Creating REST API, T_{16}: Configuring ETL Model, T_{17}: ETL Model Hash, T_{18}: Loading ETL Model, T_{19}: Deploy CD Containers, T_{20}: Service Registration (Participants), T_{21}: Task Assignment, T_{22}**: Overall Response Time**.

Table 3 shows a summary of the mean response times produced by the studied configurations in comparison. It also includes deviation and the deviation % of the metrics for a 100-data user workload. The deviation percentage of the values obtained for the mean performance in the solutions that include the CV model varies in the range of [3.37 and 6.11%].

5.4.3. Analysis of the verifiability component of the best solution GM-CD/CV

We analyze the best solution of GM-CD/CV to determine the construction times of the verifiability network. Taking the results previously described into account, we consider that the best configuration found for the GM-CD/CV including two peer nodes with ten services per stage (CD/CV-$S_{10} - P_2$).

Although the results between CD/CV-$S_{10} - P_2$ and CD/CV-$S_{10} - P_1$ are very close, it is advisable to have a larger number of peer nodes per organization for an eventual failure of a peer node. For example, in the configuration of 1 peer node (CD/CV-$S_{10} - P_1$), a breakdown in the communication with the only peer node of an organization could be critical since it could disable the access of that organization to the verifiability network and not be able to issue records to the network.

Fig. 18 shows the execution times of the processes executed in a sequential and phase-dependent manner when deploying our proposed solution.

In Fig. 18, the first seven tasks (T_{16}: T_{22}) illustrate the execution and service times of each process of phases such as declaration, deployment, and operation of the management of CD/CV components. The tasks of the preparation phase (T_{16}, T_{17}, T_{18}) correspond to the time to create the configuration files necessary to ensure continuous delivery (CD) in the workflow from the notation *DAG* given by the user by using the ETL model. Subsequently, in the deployment phase, the proposed solution executes the task T_{19}, which consists of the deployment or start-up of the services of each of the stages, which we call CD containers. When the solution correctly deployed the services on the infrastructure used in the experimentation, the manager registers these containers as participants of the verifiability network (T_{20}) and assigns them the workload to be executed by each one (T_{21}). Finally, the manager launches the workflow in the operation phase, where it is calculated the response time (T_{22}) of processing the 100 data user workloads through the workflow.

We divided into two phases the processes executed for the construction, deployment, and operation of the verifiability network: the first corresponds to the infrastructure of the verification network based on the creation of the blockchain, and the second corresponds to the business network that models the logic of the transactions that the solution recorded in the blockchain. We describe the times used in each of these phases below:

1. Continuous Verification Phase 1 — Creation and deployment of the verifiability network. In this phase, the verifiability network is built, deployed, and put into production employing the Hyperledger Fabric module. The times obtained in each of the processes performed in this phase correspond to tasks T_1, T_2, T_3, T_4, and T_5 shown in Fig. 18.
2. Continuous Verification Phase 2 — Business Network: In this phase, the business network is built, deployed, and installed in the verifiability network employing the Composer module. The times obtained in each of the processes performed in this phase correspond from task T_6 to task T_{15}, which we present in Fig. 18.

Among the results, the best configuration obtained in the comparative phase of the user mobility case study shows that the time required to deploy the CV component (tasks T_1 to T_5 of the Hyperledger Fabric component and T_6 to T_{15} for the Business module) is much greater than that required by the CD component (tasks T_{16} to T_{21}) to perform continuous delivery. A scenario that considers a higher number of peer nodes per organization than the one described by the best configuration will generate more deployment overhead for the CV component than the CD component. The process identified as T_{22} includes the processing time of the entire workload and the verifiability network registration times that contain asset registration and transaction registration.

6. Discussion

The GM-CD/CV (or **G**lobal **M**anager-**C**ontinuous **D**elivery /**C**ontinuous **V**erifiability) presented in this paper is an intermediary manager between workflow engines and blockchains for verification and validation of the execution of IoT dataflow in an established sequence and the integrity of digital assets. The CD/CV schemes proposed allow to perform continuous verifiability based on blockchain (CV model) of the transactions performed by applications included in workflows (CD model), that are recorded transparently and automatically in the blockchain system. These schemes incorporate parallel patterns encapsulated into virtual containers to reduce the impact of such verifiability on the performance of workflows when producing IoT dataflows.

We have built, deployed, and evaluated a prototype to create CD/CV schemes for processing data in an IoT dataflow from mobile devices and sensors to the cloud. These schemes, not only verify the execution and sequence of the processing stages (CD model), but also the transactions (data exchange) performed by participants when processing assets (e.g., IoT data).

We conducted a case study focused on the analysis of user mobility data, which revealed that GM-CD/CV not only registered the transactions in IoT dataflow, but its parallel patterns also reduced the overhead produced by the verifiability processes. A direct comparison with state-of-the-art solutions that produces continuous delivery (without a verifiability network) supports this claim.

GM-CD/CV is able to manage records in the verifiability network with coarse and fine grain strategies. The experimental evaluation showed that the coarse-grained method, applied to information batches, is suitable for scenarios where data generation is constant (sensors). On the other hand, the fine-grain strategy is used in each ETL transformation process (for each transformed file) to generate a digital signature for each data arriving and departing to/from each stage of the workflows.

The evaluation revealed that the coarse-grained strategy reduces the costs of continuous verifiability by recording a consolidation of data transactions (a dataflow-based report), which reduces the number of registration processes in the verifiability network. This strategy is more suitable for IoT dataflows producing data at constant rates (data produced by sensors in time series). In turn, the second strategy registers, in real-time, each transaction performed during a dataflow (stage-based report) in the blockchain, which however increases the costs of information recording. This strategy is more suitable for large data produced asynchronously (e.g., when managing files not streaming).

In performance details, the experimental evaluation also revealed that the number of peer nodes in the CV model deployed for each organization significantly impacts the performance of the CD model. Two peer nodes per organization were sufficient to perform transaction recording with service availability without imposing relevant costs on the CD model.

The experimental comparison of GM-CD/CV with a state-of-the-art solution based on parallel stages (DagOnStar), revealed that DagOnStar does not allow highly concurrent processing of workloads (it was not possible to process trajectories of 100 users in the mobility scenario) while GM-CD/CV did not show this limitation. DagOnStar is quite competitive when processing short tasks and is not limited to launching execution threads for stages (e.g., processing ten users). The best configuration of DagOnStar produced a 20% performance improvement compared to GM-CD/CV, but knowing that DagOnStar does not use the verifiability network (not performing blockchain records). In scenarios where GM-CD/CV and DagOnStar used the same number of resources, the overhead was not only eliminated, but even GM-CD/CV produced a better performance than DagOnStar (82.88% on average for 10 and 12 threads configurations with 1 and 10 user loads).

Our contributions are important for those companies and organizations requiring traceability of the processes executed along the several applications composing a workflow. Moreover, the continuous verifiability allows firing automatic processes to generate alarms, maintenance actions, or corrective processes to solve failures in the data-flow processing, which is important for supply chains, health workflows, business, and many other sectors. Thus, our solution has many potential applications, and applying it to several sectors with real-world problems will be part of our next steps.

7. Conclusions and future work

In this paper we presented GM-CD/CV (or **G**lobal **M**anager-**C**ontinuous **D**elivery /**C**ontinuous **V**erifiability), an intermediary manager between workflow engines and blockchains for verification and validation of the execution of IoT dataflow in an established sequence and the integrity of digital assets.

The main contributions of this papers over the current state of the art are the design and implementation of a Continuous Delivery/Continuous Verifiability model IoT–edge–fog–cloud system linking data-flow processing with blockchain registering of

significant events, and allowing the automatic deployment of smart contracts associated to any stage of the workflow to fire actions related to the continuous verifiability. Moreover, we have optimized the model to provide bear real-time recording of transactions in blockchain. Or experimental evaluation shows the feasibility and good performance of our solution applied to a mobility use case and compared with a state-of-the-art workflow manager, even if the last one does not include blockchain recording.

Given the growth of IoT devices and the exponential increase in the volume of data generated in this environment, GM-CD/CV results are quite useful for organizations to establish continuous verifiability in/during IoT processing dataflows.

However, even if the model proposed is general, we need to assess it with more datasets and business sectors uses cases to show its feasibility for multi-organization scenarios.

As future work, we propose the following aspects: (1) The incorporation of a high availability scheme for GM-CD/CV. A failure in a CD service could interrupt the continuous delivery process, or in the CV services (peer nodes), could interrupt or disable the records to the verifiability network; (2) Define and evaluate a geographically distributed multi-organization scenario. It is of interest to know the performance of GM-CD/CV in this scenario where is deployed each stage of each organization in a completely different geographical location, as well as the peer nodes of each organization, and that make up the verifiable network; (3) Hash function method for the construction of the transactional record, is necessary to preserve each resulting process between stages in the file system of the proposed infrastructure. This preservation of the results implies a reading and writing cost that directly affects the response times perceived by the user. To reduce costs, we propose the exchange of information between stages from memory. However, it is necessary to study how to guarantee continuous verifiability under this approach. (4) Security for CD/CV sharing secret for avoiding/mitigating man-in-the-middle attacksi (Berdik, Otoum, Schmidt, Porter, & Jararweh, 2021).

CRediT authorship contribution statement

Cristhian Martinez-Rendon: Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing – original draft, Writing – review & editing. **J.L. González-Compeán:** Conception and design of study, Analysis and/or interpretation of data, Writing – review & editing. **Dante D. Sánchez-Gallegos:** Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing – original draft. **Jesus Carretero:** Conception and design of study, Analysis and/or interpretation of data, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work has been partially supported by the project "CABAHLA-CM: Convergencia Big data-Hpc: de los sensores a las Aplicaciones" S2018/TCS-4423 from Madrid Regional Government, Spain and by the Spanish Ministry of Science and Innovation Project "New Data Intensive Computing Methods for High-End and Edge Computing Platforms (DECIDE)". Ref. PID2019-107858GB-I00; and by the project 41756 "Plataforma tecnológica para la gestión, aseguramiento, intercambio *y* preservación de grandes volúmenes de datos en salud *y* construcción de un repositorio nacional de servicios de análisis de datos de salud" by the PRONACES-CONACYT, Mexico. All authors approved the final version of the manuscript.

References

Ahmad, A., Cuomo, S., Wu, W., & Jeon, G. (2019). Intelligent algorithms and standards for interoperability in internet of things.

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., et al. (2018). Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth eurosys conference* (p. 30). ACM.

Armenise, V. (2015). Continuous delivery with Jenkins: Jenkins solutions to implement continuous delivery. In *Proceedings of the third international workshop on release engineering* (pp. 24–27). IEEE.

Babuji, Y., Woodard, A., Li, Z., Katz, D. S., Clifford, B., Kumar, R., et al. (2019). Parsl: Pervasive parallel programming in python. In *Proceedings of the 28th international symposium on high-performance parallel and distributed computing* (pp. 25–36).

Berdik, D., Otoum, S., Schmidt, N., Porter, D., & Jararweh, Y. (2021). A survey on blockchain for information systems management and security. *Information Processing & Management*, *58*(1), Article 102397. http://dx.doi.org/10.1016/j.ipm.2020.102397, URL https://www.sciencedirect.com/science/article/pii/S030645732030892X.

Bhushan, B., Sahoo, C., Sinha, P., & Khamparia, A. (2020). Unification of Blockchain and Internet of Things (BIoT): requirements, working model, challenges and future directions. *Wireless Networks*, 1–36.

Braun, C. H.-J., Traue, J., Lingl, B., & Käfer, T. (2021). Documenting the execution of semantically modelled inter-organisational workflows on a distributed ledger. In *2021 IEEE international conference on blockchain (Blockchain)* (pp. 280–286). http://dx.doi.org/10.1109/Blockchain53845.2021.00045.

Bumblauskas, D., Mann, A., Dugan, B., & Rittmer, J. (2020). A blockchain use case in food distribution: Do you know where your food has been? *International Journal of Information Management*, *52*, Article 102008.

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

Deelman, E., Singh, G., Su, M.-H., Blythe, J., Gil, Y., Kesselman, C., et al. (2005). Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming*, *13*(3), 219–237.

Deelman, E., Vahi, K., Rynge, M., Mayani, R., da Silva, R. F., Papadimitriou, G., et al. (2019). The evolution of the pegasus workflow management software. *Computing in Science & Engineering*, *21*(4), 22–36.

Diène, B., Rodrigues, J. J., Diallo, O., Ndoye, E. H. M., & Korotaev, V. V. (2020). Data management techniques for Internet of Things. *Mechanical Systems and Signal Processing*, *138*, Article 106564.

Esposito, S., Cafiero, A., Giannino, F., Mazzoleni, S., & 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.

Ethereum (2020). Ethereum. URL https://ethereum.org/.

Fernández-Caramés, T. M., & Fraga-Lamas, P. (2018). A review on the use of blockchain for the internet of things. *Ieee Access*, *6*, 32979–33001.

Firouzi, F., Farahani, B., & Marinšek, A. (2021). The convergence and interplay of edge, fog, and cloud in the AI-driven Internet of Things (IoT). *Information Systems*, Article 101840.

Fu, W., Wei, X., & Tong, S. (2021). An improved blockchain consensus algorithm based on raft. *Arabian Journal for Science and Engineering*, *46*(9), 8137–8149.

Gjoreski, H., Ciliberto, M., Morales, F. J. O., Roggen, D., Mekki, S., & Valentin, S. (2017). A versatile annotated dataset for multimodal locomotion analytics with mobile devices. In *Proceedings of the 15th ACM conference on embedded network sensor systems* (pp. 1–2).

Goble, C., Cohen-Boulakia, S., Soiland-Reyes, S., Garijo, D., Gil, Y., Crusoe, M. R., et al. (2020). FAIR computational workflows. *Data Intelligence*, *2*(1–2), 108–121.

Gonzalez, J. L., Perez, J. C., Sosa-Sosa, V. J., Sanchez, L. M., & Bergua, B. (2015). SkyCDS: A resilient content delivery service based on diversified cloud storage. *Simulation Modelling Practice and Theory*, *54*, 64–85.

Gonzalez-Compean, J., Sosa-Sosa, V., Diaz-Perez, A., Carretero, J., & Yanez-Sierra, J. (2018). Sacbe: A building block approach for constructing efficient and flexible end-to-end cloud storage. *Journal of Systems and Software*, *135*, 143–156.

Guegan, D. (2017). *Public blockchain versus private blockhain*: *Tech. rep.*, Centre d'Economie de la Sorbonne.

He, W., Wang, F.-K., & Akula, V. (2017). Managing extracted knowledge from big social media data for business decision making. *Journal of Knowledge Management*.

Helo, P., & Shamsuzzoha, A. (2020). Real-time supply chain—A blockchain architecture for project deliveries. *Robotics and Computer-Integrated Manufacturing*, *63*, Article 101909.

Hyperledger-Fabric (2020). Hyperledger fabric. URL https://www.hyperledger.org/projects/fabric.

Kousalya, G., Balakrishnan, P., & Raj, C. P. (2017). Workflow management systems. In *Automated workflow scheduling in self-adaptive clouds* (pp. 55–64). Springer.

Kurt Peker, Y., Rodriguez, X., Ericsson, J., Lee, S. J., & Perez, A. J. (2020). A cost analysis of internet of things sensor data storage on blockchain via smart contracts. *Electronics*, *9*(2), 244.

Li, Y., Jha, D. N., Aujla, G. S., Morgan, G., Zomaya, A. Y., & Ranjan, R. (2020). IoTWC: Analytic hierarchy process based internet of things workflow composition system. In *2020 IEEE international conference on cloud engineering (IC2E)* (pp. 1–10). IEEE.

Li, G., Ren, X., Wu, J., Ji, W., Yu, H., Cao, J., et al. (2021). Blockchain-based mobile edge computing system. *Information Sciences*, *561*, 70–80.

Lofstead, J., Baker, J., & Younge, A. (2019). Data pallets: containerizing storage for reproducibility and traceability. In *International conference on high performance computing* (pp. 36–45). Springer.

Louis, J., & Dunston, P. S. (2018). Integrating IoT into operational workflows for real-time and automated decision-making in repetitive construction operations. *Automation in Construction*, *94*, 317–327.

Martinez-Rendon, C., Camarmas-Alonso, D., Carretero, J., & Gonzalez-Compean, J. L. (2021). On the continuous contract verification using blockchain and real-time data. *Cluster Computing*, 1–23.

Mathew, W., Raposo, R., & Martins, B. (2012). Predicting future locations with hidden Markov models. In *Proceedings of the 2012 ACM conference on ubiquitous computing* (pp. 911–918).

Mazzei, D., Baldi, G., Fantoni, G., Montelisciani, G., Pitasi, A., Ricci, L., et al. (2020). A blockchain tokenizer for industrial IOT trustless applications. *Future Generation Computer Systems*, *105*, 432–445.

Medvedev, A., Fedchenkov, P., Zaslavsky, A., Anagnostopoulos, T., & 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* (pp. 104–115). Springer.

Meroni, G., Baresi, L., Montali, M., & Plebani, P. (2018). Multi-party business process compliance monitoring through IoT-enabled artifacts. *Information Systems*, *73*, 61–78.

Miles, A., Zaslavsky, A., & 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.

Mocanu, B., Pop, F., Mihaita, A., Dobre, C., & Castiglione, A. (2019). Data fusion technique in spider peer-to-peer networks in smart cities for security enhancements. *Information Sciences*, *479*, 607–621.

Montella, R., Di Luccio, D., & Kosta, S. (2018). DagOn*: Executing direct acyclic graphs as parallel jobs on anything. In *2018 IEEE/ACM workflows in support of large-scale science* (pp. 64–73). IEEE.

Montella, R., Ruggieri, M., & Kosta, S. (2018). A fast, secure, reliable, and resilient data transfer framework for pervasive IoT applications. In *IEEE INFOCOM 2018-IEEE conference on computer communications workshops (INFOCOM WKSHPS)* (pp. 710–715). IEEE.

Montoliu, R., Blom, J., & Gatica-Perez, D. (2013). Discovering places of interest in everyday life from smartphone data. *Multimedia Tools and Applications*, *62*(1), 179–207.

Mrozek, D., Koczur, A., & Małysiak-Mrozek, B. (2020). Fall detection in older adults with mobile IoT devices and machine learning in the cloud and on the edge. *Information Sciences*, *537*, 132–147.

Municio, E., Marquez-Barja, J., Latré, S., & Vissicchio, S. (2018). Whisper: Programmable and flexible control on industrial IoT networks. *Sensors*, *18*(11), 4048.

Nakamoto, S., et al. (2008). *Bitcoin: A peer-to-peer electronic cash system*: *Working Paper*.

Nardelli, M., Nastic, S., Dustdar, S., Villari, M., & Ranjan, R. (2017). Osmotic flow: Osmotic computing+ iot workflow. *IEEE Cloud Computing*, *4*(2), 68–75.

Nasir, Q., Qasse, I. A., Abu Talib, M., & Nassif, A. B. (2018). Performance analysis of hyperledger fabric platforms. *Security and Communication Networks*, *2018*.

Networking, C. V. (2018). *Cisco global cloud index: Forecast and methodology 2016–2021*: *White Paper*.

Nguyen, T. S. L., Jourjon, G., Potop-Butucaru, M., & Thai, K. L. (2019). Impact of network delays on hyperledger fabric. In *IEEE INFOCOM 2019-IEEE conference on computer communications workshops (INFOCOM WKSHPS)* (pp. 222–227). IEEE.

Ortega-Arjona, J. L. (2010). *Patterns for parallel software design* (1st ed.). Wiley Publishing.

Pahlajani, S., Kshirsagar, A., & Pachghare, V. (2019). Survey on private blockchain consensus algorithms. In *2019 1st international conference on innovations in information and communication technology* ICIICT, (pp. 1–6). IEEE.

Panarello, A., Tapas, N., Merlino, G., Longo, F., & Puliafito, A. (2018). Blockchain and iot integration: A systematic survey. *Sensors*, *18*(8), 2575.

Pinoli, P., Ceri, S., Martinenghi, D., & Nanni, L. (2019). Metadata management for scientific databases. *Information Systems*, *81*, 1–20.

Qasha, R., Cala, J., & Watson, P. (2016). Dynamic deployment of scientific workflows in the cloud using container virtualization. In *2016 IEEE international conference on cloud computing technology and science (CloudCom)* (pp. 269–276). IEEE.

Qin, X., Huang, Y., Yang, Z., & Li, X. (2021). LBAC: A lightweight blockchain-based access control scheme for the internet of things. *Information Sciences*, *554*, 222–235.

Ramachandran, M., Chowdhury, N., Third, A., Domingue, J., Quick, K., & Bachler, M. (2020). Towards complete decentralised verification of data with confidentiality: Different ways to connect solid pods and blockchain. In *Companion proceedings of the web conference 2020* WWW '20, (pp. 645–649). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3366424.3385759.

Reyna, A., Martín, C., Chen, J., Soler, E., & Díaz, M. (2018). On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems*, *88*, 173–190.

Ruan, J., & Shi, Y. (2016). Monitoring and assessing fruit freshness in IOT-based e-commerce delivery using scenario analysis and interval number approaches. *Information Sciences*, *373*, 557–570.

Sánchez-Gallegos, D. D., Carrizales-Espinoza, D., Reyes-Anastacio, H. G., Gonzalez-Compean, J., Carretero, J., Morales-Sandoval, M., et al. (2020). From the edge to the cloud: A continuous delivery and preparation model for processing big IoT data. *Simulation Modelling Practice and Theory*, *105*, Article 102136.

Sánchez-Gallegos, D. D., Di Luccio, D., Gonzalez-Compean, J. L., & Montella, R. (2019). Internet of Things orchestration using DagOn* workflow engine. In *2019 IEEE 5th world forum on internet of things (WF-IoT)* (pp. 95–100). IEEE.

Sánchez-Gallegos, D. D., Di Luccio, D., Kosta, S., Gonzalez-Compean, J., & Montella, R. (2021). An efficient pattern-based approach for workflow supporting large-scale science: The DagOnStar experience. *Future Generation Computer Systems*, *122*, 187–203.

Sánchez-Gallegos, D. D., Galaviz-Mosqueda, A., Gonzalez-Compean, J., Villarreal-Reyes, S., Perez-Ramos, A. E., Carrizales-Espinoza, D., et al. (2020). On the continuous processing of health data in edge-fog-cloud computing by using micro/nanoservice composition. *IEEE Access*, *8*, 120255–120281.

Sanchez-Gallegos, D. D., Gonzalez-Compean, J., Carretero, J., Marin, H., Tchernykh, A., & Montella, R. (2022). PuzzleMesh: A puzzle model to build mesh of agnostic services for edge-fog-cloud. *IEEE Transactions on Services Computing*.

Serhani, M. A., El-Kassabi, H. T., Shuaib, K., Navaz, A. N., Benatallah, B., & Beheshti, A. (2020). Self-adapting cloud services orchestration for fulfilling intensive sensory data-driven IoT workflows. *Future Generation Computer Systems*, *108*, 583–597.

Shaham, S., Ding, M., Liu, B., Lin, Z., & Li, J. (2019). Machine learning aided anonymization of spatiotemporal trajectory datasets. In *IEEE INFOCOM 2019-IEEE conference on computer communications workshops (INFOCOM WKSHPS)* (pp. 1–6). IEEE.

Shahhosseini, S., Anzanpour, A., Azimi, I., Labbaf, S., Seo, D., Lim, S.-S., et al. (2021). Exploring computation offloading in IoT systems. *Information Systems*, Article 101860.

Shao, Y., Li, C., & Tang, H. (2019). A data replica placement strategy for IoT workflows in collaborative edge and cloud environments. *Computer Networks*, *148*, 46–59.

Shukla, M., Lin, J., & Seneviratne, O. (2022). Blockchain and IoT enhanced clinical workflow. In *International conference on artificial intelligence in medicine* (pp. 407–411). Springer.

Siddiqa, A., Hashem, I. A. T., Yaqoob, I., Marjani, M., Shamshirband, S., Gani, A., et al. (2016). A survey of big data management: Taxonomy and state-of-the-art. *Journal of Network and Computer Applications*, *71*, 151–166.

Simpkin, C., Taylor, I., Harborne, D., Bent, G., Preece, A., & Ganti, R. K. (2020). Efficient orchestration of node-RED IoT workflows using a vector symbolic architecture. *Future Generation Computer Systems*, *111*, 117–131.

Singh, P., & Singh, N. (2020). Blockchain with IoT and AI: A review of agriculture and healthcare. *International Journal of Applied Evolutionary Computation (IJAEC)*, *11*(4), 13–27.

Taherkordi, A., & Herrmann, P. (2018). Pervasive smart contracts for blockchains in iot systems. In *Proceedings of the 2018 international conference on blockchain technology and application* (pp. 6–11).

Terstyanszky, G., Kukla, T., Kiss, T., Kacsuk, P., Balaskó, Á., & Farkas, Z. (2014). Enabling scientific workflow sharing through coarse-grained interoperability. *Future Generation Computer Systems*, *37*, 46–59.

Vassiliadis, P. (2009). A survey of extract–transform–load technology. *International Journal of Data Warehousing and Mining (IJDWM)*, *5*(3), 1–27.

Vujičić, D., Jagodić, D., & Randjić, S. (2018). Blockchain technology, bitcoin, and ethereum: A brief overview. In *2018 17th international symposium Infoteh-Jahorina (Infoteh)* (pp. 1–6). IEEE.

Wang, Y., Li, S., Xu, L., & Xu, L. (2021). Improved raft consensus algorithm in high real-time and highly adversarial environment. In *International conference on web information systems and applications* (pp. 718–726). Springer.

Wood, G., et al. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, *151*, 1–32.

Xiao, Z., Wang, Y., Fu, K., & Wu, F. (2017). Identifying different transportation modes from trajectory data using tree-based ensemble classifiers. *ISPRS International Journal of Geo-Information*, *6*(2), 57.

Xu, X., Sun, G., Luo, L., Cao, H., Yu, H., & Vasilakos, A. V. (2021). Latency performance modeling and analysis for hyperledger fabric blockchain network. *Information Processing & Management*, *58*(1), Article 102436. http://dx.doi.org/10.1016/j.ipm.2020.102436, URL https://www.sciencedirect.com/science/article/pii/S0306457320309298.

Yoo, A. B., Jette, M. A., & Grondona, M. (2003). Slurm: Simple linux utility for resource management. In *Workshop on job scheduling strategies for parallel processing* (pp. 44–60). Springer.

Yu, X., Yan, Z., & Vasilakos, A. V. (2017). A survey of verifiable computation. *Mobile Networks and Applications*, *22*(3), 438–453.

Zhao, Q., Chen, S., Liu, Z., Baker, T., & Zhang, Y. (2020). Blockchain-based privacy-preserving remote data integrity checking scheme for IoT information systems. *Information Processing & Management*, *57*(6), Article 102355. http://dx.doi.org/10.1016/j.ipm.2020.102355, URL https://www.sciencedirect.com/science/article/pii/S0306457320308505.

Zheng, Y., Fu, H., Xie, X., Ma, W.-Y., & 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.