Paper

# Semantic Knowledge Management and Blockchain-based Privacy for Internet of Things Applications

Manal Lamri[1] and Lyazid Sabri[1,2]

[1]*Faculty of Mathematics and Informatics, University of Mohamed El Bachir EL Ibrahimi, Algeria,*
[2]*Laboratory of Images, Signals and Intelligent Systems, University of Paris-Est, France*

https://doi.org/10.26636/jtit.2022.161522

**Abstract — Design of distributed complex systems raises several important challenges, such as: confidentiality, data authentication and integrity, semantic contextual knowledge sharing, as well as common and intelligible understanding of the environment. Among the many challenges are semantic heterogeneity that occurs during dynamic knowledge extraction and authorization decisions which need to be taken when a resource is accessed in an open, dynamic environment. Blockchain offers the tools to protect sensitive personal data and solve reliability issues by providing a secure communication architecture. However, setting-up blockchain-based applications comes with many challenges, including processing and fusing heterogeneous information from various sources. The ontology model explored in this paper relies on a unified knowledge representation method and thus is the backbone of a distributed system aiming to tackle semantic heterogeneity and to model decentralized management of access control authorizations. We intertwine the blockchain technology with an ontological model to enhance knowledge management processes for distributed systems. Therefore, rather than relying on the mediation of a third party, the approach enhances autonomous decision-making. The proposed approach collects data generated by sensors into higher-level abstraction using n-ary hierarchical structures to describe entities and actions. Moreover, the proposed semantic architecture relies on hyperledger fabric to ensure the checking and authentication of knowledge integrity while preserving privacy.**

***Keywords — hyperledger fabric, ontology, security of distributed system, spatio-temporal knowledge representation***

## 1. Introduction

The design of distributed complex systems capable of improving the perception of the user's context and making the interaction between humans and systems/robots more natural, requires data privacy protection, management and the sharing of a common and intelligible understanding of the environment [1]–[3]. Several technologies have emerged to ensure privacy and to identify when unauthorized users are trying to access the system [4]–[6]. As an emerging technology, blockchain offers tools that: protect sensitive personal data, solve reliability issues by supplying a secure communication architecture in a distributed application design (e.g. Industry 4.0), and eliminate the need for mediation of third-authority organizations. Moreover, blockchain technologies offer the concept of smart contracts (SM), e.g. rules and actions based on predefined scenarios. SM is self-executing using data spread within the blockchain network, thus providing a higher level of autonomy required in IoT applications.

In contrast, traditional network architectures and numerous traditional privacy protection schemes store data in a centralized server to ensure that data is not disclosed. Moreover, decentralized management models rely on encryption techniques (e.g. public and private keys), guaranteeing access control authorizations between several domains. However, as demonstrated in [7], due to computing power constraints associated with IoT devices, such a cryptography technique cannot be suitably managed in all the layers. Besides, the authors highlighted another main challenge that IoT application designers face, namely heterogeneity of devices and services. Thus, adequate performance at the communication and storage level is not really achievable.

Since devices are energy-constrained, the use of permissioned blockchain hyperledger fabric (HF) [8] seems to be more suitable for dealing with IoT requirements. HF allows access and process data in real time, instantaneously taking adequate decisions to tackle emergencies, such as malicious actions or fall detection. However, in the process of designing these blockchain-based applications, many issues need to be taken into consideration, e.g. heterogeneous data fusion.

Many works point out that [9]–[12] ontologies are one of the best solutions available today to share knowledge, as they ensure secure communication that preserves privacy, offers authentication mechanisms and supports semantic interoperability across heterogeneous information sources. Consequently, intertwining ontology with blockchain provides a good level of autonomy by sharing the related IoT data, addresses data heterogeneity issues, and allows reasoning about SM semantics. To fulfill these goals, we essentially need the following:

– **Post-processing sensor information**. An ontological model must consider both the static (physical objects) and the dynamic (events) layer. Here, the aim is to link entities (e.g. humans, robots, sensors) with ontological-grounding concepts.

– **Architecture for reasoning and decision-making systems**. It offers high interoperability and abstraction of the entities, with mechanisms for secure event/knowledge delivery, simultaneously ensuring their privacy and confidentiality. More precisely, it will be applied in integrated smart surveillance systems for physical and digital assets and personal safety.
– **Assuring data integrity**. Thanks to the HF platform, each entity has its own Blockchain identity, which ensures data integrity and authentication while preserving privacy.

Defined in this manner, this model allows a narrative representation of events and establishes implicit semantic relationships (causality, purpose, etc.) between events observed in the environment. More precisely, it models inter-dependencies between contexts and expresses knowledge of: who is the initiator (i.e. agent) of the event/action? The objective is to enrich the interpretation of the context to ensure better adaptation. This approach relies on two types kinds of ontologies of the narrative knowledge representation language (NKRL) [13]: a binary ontology known as HClass and an $n$-ary structure known as hierarchy temporal (HTemp).

The latter uses semantic predicates/roles to represent dynamic Knowledge pertaining to: what are the context-related events that have been observed? Has an action been performed? Which entity (beneficiary) derives a benefit from the completion of the event/action?

Let us consider a scenario devoted to monitoring an elderly person wearing a fall sensor. Depending on knowledge analysis, the robot accomplishes tasks under different environmental conditions and recognizes situations/contexts (e.g. falls). Distributed IoT devices assist the robot in localizing the elderly person. A concept defined in an HClass ontology becomes identifiable from the data provided by the sensors. Moreover, based on principles of cryptography, HF secures access to the robot's embedded camera to allow the hospital staff to evaluate the patient's health condition during the wait for the paramedics.

The remainder of the paper is organized as follows. Section 2 presents the background of the approach. The ontological knowledge representation approach is presented and detailed in Sections 3 and 4. A functional and architectural description of the proposed framework is given in Section 5, where an exemplified case study is presented as well. Finally, experimental evaluations and conclusions are given in Sections 6 and 7.

## 2. Hyperledger Fabric Blockchain Basics

Blockchain structures data into blocks. Each block contains a transaction (several transactions) and all blocks are organized into a cryptographic chain. Each transaction is secured, authenticated, and added to a secure, immutable data chain. A consensus-based algorithm allows to add new blocks to the blockchain network. The consensus algorithm is responsible for data integrity in the blockchain network and prevents service attacks for double-spending [14]–[15].

Hyperledger fabric has the form of a permissioned network. It is limited to a set of users and the consensus is achieved through a selective endorsement process. Thus, HF offers the following advantages: it saves time, removes cost (overhead and cost components), reduces risks (tampering, fraud, and cybercrime), and increases trust level [16]. The blockchain network has a single view of the dataset, and each node (e.g. participant) stores its code. Below, we describe each

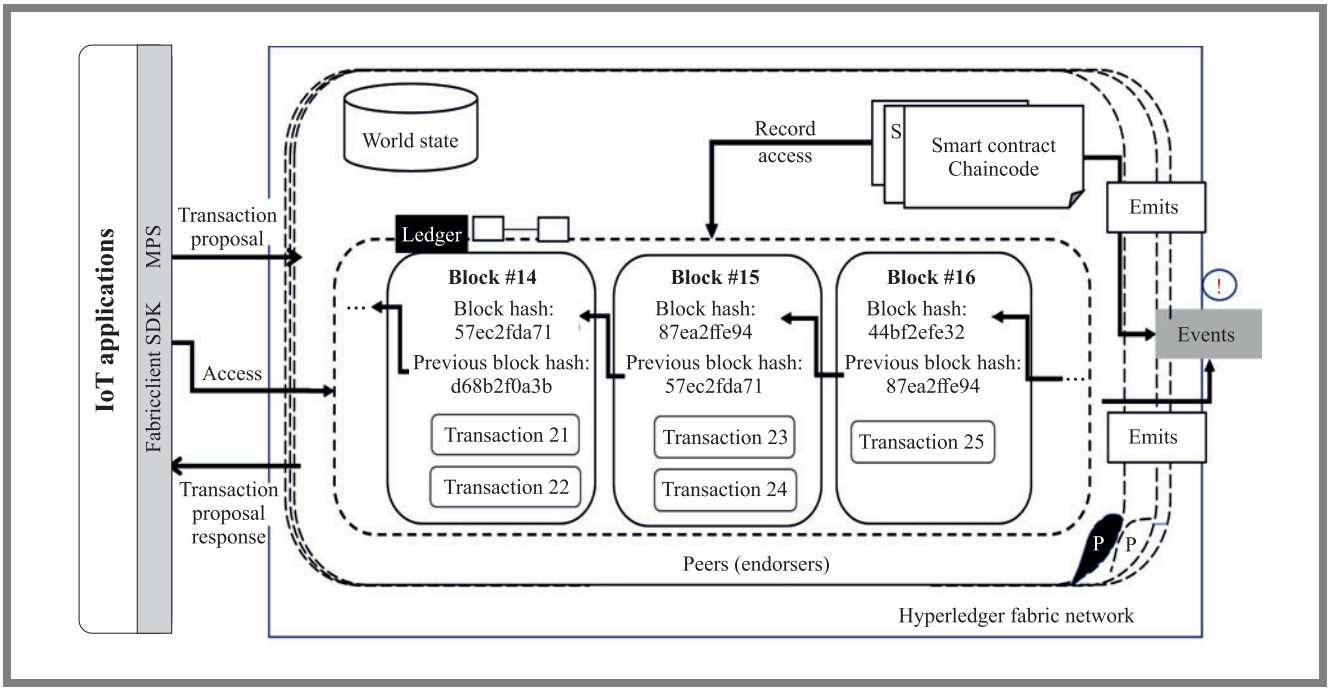



**Fig. 1.** Main components in a blockchain hyperledger fabric.

component (depicted in Fig. 1) and show how the cryptographic hash function secures the blockchain [17].

The ledger is a tamper-resistant of transitions. It stores the immutable blocks and the current state. The world state is seen as an ordinary database that stores and provides combined outputs of all transactions. Each participant within the blockchain network maintains a copy of the ledger. A cryptographic hash function accepts any binary data as input. A hash function (algorithm) is applied to the input data and generates an output whose length is determined in bytes (hash value), such as `57ec2fda71` (Fig. 1). The hash value serves as a digital fingerprint of that data.

A smart contract (i.e. chaincode) is signed encoded in the programming language. Unlike a traditional contract, SM defines the applicable rules in a blockchain. A chaincode is therefore a program that runs automatically in a blockchain to restrict actions or to transfer assets when specific conditions are met.

A peer network consists of the main component of a HF that belongs to a consortium. It stores the blockchain ledger, runs the SM, validates transactions and adds blocks to the ledger. Thereby, a peer network is seen as an overarching entity of the transactional flow.

Membership services authentication (MSP) is a process that allows to manage identities and authorize participants to access networks in a permissioned blockchain network. Since MSP handles all permissioned identities and knows all of the organization's members, it may recognize and trust each participant. That is why it is considered to be the backbone of the blockchain network.

Event handlers create notifications concerning significant blockchain operations and notifications related to smart contracts. All events include the transaction ID and allow the applications to take action when a transaction is concluded.

Fabric SDK allows the system to create, update and monitor blockchain components. The fabric client (made available through the fabric SDK) provides the `queryByChaincode()` API for developers to transmit a query request to a peer. Such a method is available on an instantiated channel object and takes two inputs. Besides using a chaincode query (queries implemented in a chaincode), a client application can send a request directly to the ledger, which is useful for retrieving metadata (for example instantiated chaincode) or for retrieving a specific transaction or block from the blockchain.

## 3. Ontological Knowledge Representation

The main difference when compared with a semantic web language, such as OWL, is that the proposed narrative knowledge representation language relies on the ontology of events called HTemp, in addition to the ontology of concepts. A template is an n-ary structure that enables a complex structured dynamic knowledge, e.g. "John falls and pushes the emergency button" to be represented. NKRL also defines the ontology of concept (HClass) that includes more than 3000 concepts. HClass is not different from frame-based or Protégé [18].

The conceptual representation of narrative knowledge is carried out through the HClass ontology of concepts. It is similar to traditional ontology languages, such as OWL or DAML+OIL. NKRL assimilates a concept to the notion of class in the semantic web. It is associated with a set of properties or attributes. NKRL distinguishes two categories of concepts: those that may be instantiated directly (`sortal_concept`) and those that cannot be instantiated directly (`non_sortal_concept`). Instantiable (i.e. instances) concepts, such as `Chair_125`, `Bed_2012`, `Tap_45`, etc., are created from the concepts `chair_`, `bed_`, `tap_` concepts.

The concept of color is an example of a concept that cannot be instantiated directly, as it cannot have a direct instance. Indeed, `Red_120` and `Yellow_1` cannot be considered instances, since they have no meaning if used separately. The solution provided by NKRL is to introduce the `color_appearance` concept, a specialization of the instantiable concept of `physical_appearance`. Thus, it is possible to associate a color with an instantiable concept, such as, for example, `Red_Table`, `Yellow_Cup`, etc.

In contrast, the ontology of events (i.e. HTemp ontology), as stated before, allows expressing knowledge concerning actions and events. We claim, therefore, that NKRL can enhance dynamic knowledge representation in the context of IoT applications. The conceptual narrative knowledge representation is structured into the following components:
**Concepts component** allows the representation of concepts. A concept is a binary representation of general notions (e.g. human-being, artifacts) or specific notions (doctor, sensor, chair). Formally, the knowledge representation of these notions is known as a concept. A concept is named using lower case symbolic labels with an "underscore", such as `human_being`, `artifact_`, `doctor_`, `sensor_`, `robot`.

**Individuals component** concerns the formal representation of instances (i.e. individuals) defined in the concepts component. Instances are created by instantiating the properties of concepts. Instances are labeled using the upper case, including the underscore symbol. `Motion_Sensor` and `Camera_1` are instances of the `sensor_concept`, and `Kitchen` is an instance of the `location_concept`.

**$n$-ary component** NKRL considers an elementary event as a spatial-temporal knowledge called a template or a predicative occurrence. Each template is expressed with the use of one predicate, specific roles and arguments. Figure 2 presents a hall structure of a predicative occurrence: `Predicate` that belongs to `MOVE`, `OWN`, `Exist`, `Produce`, `Receive`, `Experience`, `Behave`. Argument represents attributes that can be associated with each generic role, i.e. `subject(Subj)`, `object(OBJ)`, `Source`, `Modal`, `Topic`, `Context`, `Beneficiary(Benf)`.

**Factual components** allow to represent events/actions extracted within a narrative as instances of the $n$-ary component. Each event allows to describe, for example, a situation and/or a robot-human interaction (e.g. `JOHN_` and the `ROBOT_KOMPAI`). Figs. 3-4 show the `PRODUCE` and `OWN` templates. The `location_concept` indicates the action's lo-

```
PREDICATE
      SUBJ {< argument >: [location]}
      OBJ {< argument >: [location]}
      SOURCE {< argument >: [location]}
      BENF {< argument >: [location]}
      MODAL {< argument >}
      TOPIC {< argument >}
      CONTEXT {< argument >}
            [modulators]
            [temporal attributes]
```

**Fig. 2.** General structure of an NKRL template.

```
name: Produce:Entity
   PREDICATE: PRODUCE
        SUBJ var1: [(var2)]
        OBJ var3
        [SOURCE var4: [(var5)]]
        [BENF var6: [(var7)]]
        [MODAL var8]
        [TOPIC var9]
        [CONTEXT var10]
        {[modulators], !=abs}
   var1 = < artefact_ > | < human_being >
   var2 = < location_ >
   var3 = < information_content >
   var4 = < human_being >
   var5 = < location_ >
   var6 = < human_being >
   var7 = < location_ >
   var8 = < temporal_development >
   var9 = < situation_ >
   var10 = < situation_ >
```

**Fig. 3.** Produce template structure.

```
name: Own:SimpleProperty
   PREDICATE: OWN
        SUBJ var1: [(var2)]
        OBJ var3
        [SOURCE var4: [(var5)]]
        [(BENF) var4]
        [MODAL var6]
        TOPIC var7
        [CONTEXT var8]
        {[modulators], != abs}
var1 != < human_being|property_ >
var2 = < location_ >
var3 = < property_ >
var4 = < human_being >
var5 = < location_ >
var6 = < artefact_ > | < temporal_sequence >
var7 != < spatial_temporal_relation >
var8 = < situation_ > | < label_ >
```

**Fig. 4.** Own template structure.

cation. Temporal attributes represented by symbolic labels represent the start or endpoints of the authorized action or duration of the executed transaction. For this purpose, NKRL supplies two modulators: begin/end. The latter is a time stamp marking the beginning or end of an action/event. The `obs` modulator is used if no information about the beginning or the end of an action is given.

Figure 3 describes the produce template. A role or variable defined in square brackets is considered optional. The `SUBJ`, `OBJ` roles and the `var1` and `var3` variables are mandatory, while the `BENF`, `MODAL`, `SOURCE`, `TOPIC` and `CONTEXT` roles, and `var6` and `var7` variables, for example, are optional. Variables `var1`, ..., `var7` describe constraints that make it possible to check whether the values assigned to each variable when creating a predicate occurrence are specific to the terms (concept, instances) used in the Individuals component. Thus, the constraints defined in the templates of the HTemp ontology are associated with the concepts defined in the HClass ontology. Therefore, the knowledge consistency checking process relies on the HClass ontology to establish a hierarchy of concepts and instances based on the generalization/specialization principle.

## 4. $n$-ary Knowledge Representation

Formally, to create a predicative occurrence, the system evaluates the expression according to the $n$-ary structure described by: equation:

$$\text{Label ")" Predicat}_{i=1} \quad (\text{Roles}_{1\leqslant j\leqslant 7} \quad \text{args}), \tag{1}$$

where:

- label identifies a predicative occurrence. The label is a sequence of characters that matches the regular expression $[a-z][1-9]\backslash.[a-z][1-9]$;
- $\text{Predicat}_{i=1}$ is one of the conceptual predicates $\in \{$`MOVE`, `PRODUCE`, `RECEIVE`, `EXPERIENCE`, `BEHAVE`, `OWN`, `EXIST`$\}$;
- $\text{Roles}_{1\leqslant j\leqslant 7}$ is a conceptual role $\in \{$`BENF`, `MODAL`, `TOPIC`, `CONTEXT`, `SUBJ`, `OBJ`, `SOURCE`$\}$
- args – this attribute belongs to concepts and individuals components.

In terms of equivalence between the NKRL representation and the description logic [19], the individuals and concepts components correspond to the ABox and TBox components. However, there are no description language equivalents for the $n$-ary and factual components.

The OWN predicate allows to describe the type of the ownership notion between The entities or the state of an entity. Therefore, to express the fact that the front door was unlocked at `04/03/2022:9:56:15:362`, the predicative occurrence `aa11.c18` (Fig. 5) uses the `SUBJ` role with the `FRONT_DOOR_BUTTON` as an argument. The property unlocked (opened) is an argument of the `TOPIC` role, the `obs` modulator indicates that the starting time belongs to `date-1` and is associated with the "front door has been unlocked" action. We highlight that each event requires that its beginning and end be distinguished. However, it is hard to establish or infer the end of an event in many scenarios. Nevertheless, deter-

mining the start of an event, as a minimum, is mandatory for further reasoning.

```
aal1.c18) PREDICATE: OWN
        SUBJ: FRONT_DOOR_BUTTON
        OBJ: property_
        TOPIC: unlocked_
              { obs }
        date-1: 04/03/2022:9:56:15:362
        date-2:
Own:SimpleProperty
aal2.c3) PREDICATE: PRODUCE
        SUBJ: CAMERA_1
        OBJ: detection_: (LOCATION_1)
        TOPIC: activity_
              { obs }
        date-1: 04/03/2022:10:31:20:102
        date-2:
Produce:Assessment/trial
aal3.c6) PREDICATE: OWN
        SUBJ: LIGHT_BUTTON_1: HALL_1
        OBJ: property_
        MODAL: lighting_: (switch_off, switch_on)
              { obs }
        date-1: 04/03/2022:10:31:22:523
        date-2:
Own:SimpleProperty
aal1.c14) PREDICATE: EXPERIENCE
        SUBJ: JOHN_
        OBJ: respiratory_distress
        MODAL:SENSOR_DISTRESS_1
              { obs }
        date-1: 04/03/2022:17:57:35:105
        date-2:
Own:SimpleProperty
```

**Fig. 5.** Examples of predicative occurrences.

The knowledge representation in NKRL allows to specify the date-1 attribute only, while the temporal attribute `date-2` is empty. In turn, the predicative occurrence `aal2.c3` (Fig. 5) expresses that the camera denoted as `CAMERA_1` recorded some activity in `LOCATION_1`. The predicative occurrence `aal3.c6` expresses that the light button localized in the hall, as denoted by `HALL_1` changed its state from `switch_off` to `switch_on`.

While the `embedded_sensor` (subclass of `artifact_`) observes that a patient denoted by `JOHN_` (`aal1.c14`, Fig. 5) displays an acute respiratory deficiency, it does not provide any information about the duration or end of this particular event. Such knowledge is expressed in NKRL using the experience templates. They are mainly used to express the fact that an entity is affected by an action. For example, the system observes a decrease in light intensity in a given space, a human suffering from an illness or an accident (e.g. a fall). While the predicative occurrence `aal1.c14` (Fig. 5) expresses that the `JOHN_` symbol, representing a human, is used as an argument of the `SUBJ(ect)` role, the `respiratory_distress` property, being an argument of the `OBJ(ect)` role related to the `date-1` attribute, is used to describe the beginning time-stamp of the action. The sensor denoted by `SENSOR_DISTRESS_1` signals that John is suffering from a respiratory failure.

## 5. NKRL and Blockchain

In order to provide an informal example of the paper's objectives, let us consider a scenario devoted to monitoring elderly persons at home. We assume that John is wearing a fall sensor used to detect the presence of an emergency alarm. Thus, the relevant contextual information considered in this use case includes the accurate location of John and his status (unconscious/conscious). The second piece of contextual information is not directly measurable, hence it is subjected to complex processes. Multiple events/actions must be correlated instantaneously to determine John's status and assess the current context/situation. The first goal consists in understanding what is happening after an alarm has been triggered. So, the robot moves towards the last location of John and tries to interact with him (i.e. check his status). The robot tries to establish a dialogue-based interaction Fig. 6. If John does not interact with the robot, he is considered unconscious, and then this non-observable context corresponds to an emergency. In this case, the monitoring function should be able to deduce the status. The second goal consists in ensuring secure communication that complies with privacy and authentication mechanisms. In fact, the doctor from the hospital will check the patient's health by observing the interaction between the robot and John. To do so, the doctor needs to remotely access the robot's embedded camera. Decision-making is followed by actions, such as allowing the hospital staff to remotely access the indoor home security camera or the robot embedded camera to evaluate the patient's condition.

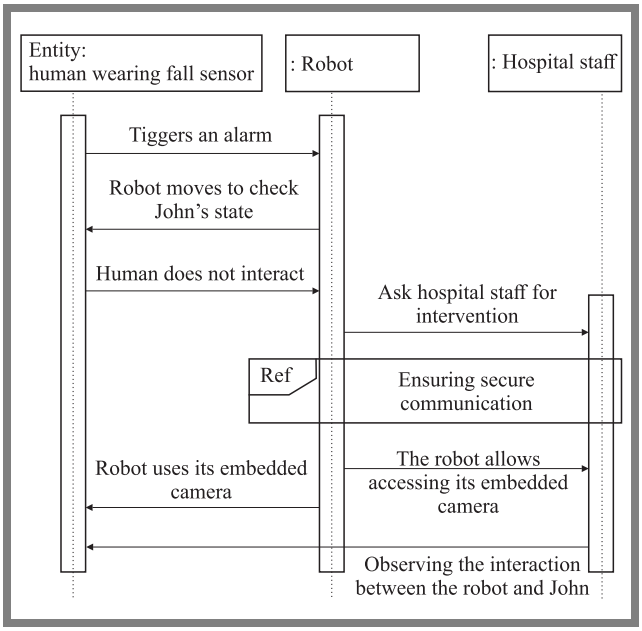



**Fig. 6.** Scenario sequence diagram.

### *5.1. HF and Distributed Complex Systems*

According to [20], due to the complexity of conditions pertaining to the network serving as a platform for communicating with IoT devices, a robot can modify its internal structure and activity patterns in the self-organization process. IoT devices generate enormous amounts of data and do not have the computational power required. That is why Bitcoin and Ethereum are not suitable for addressing (i.e. managing) actions/contexts described in the scenario above, where an instantaneous reaction is needed. Moreover, IoT devices should mine and create blocks according to the POW-based (proof-of-work) protocol that calls for considerable amounts of energy [14]. However, HF is an open-source distributed ledger platform based on the Linux architecture. It establishes decentralized trust in a network. Only the data we intend to share are shared among the relevant participants, i.e. advanced privacy controls are ensured.

HF is permissioned blockchain and empowered building a consortium, meaning that the participants are identified and may not trust each other. It provides pluggable consensus protocols allowing organizations (multiparty) to customize their consensus protocols. Each participant controls one or more peers (nodes in the chain) and should treat a chaincode as unreliable, since anyone can dynamically deploy a smart contract. Moreover, HF can rely on Byzantine-fault tolerant (BFT) [21] instead of POW consensus algorithms and the execute-order-validate architecture. Therefore, HF enables scalability, i.e., sharing information and permitting IoT devices to execute actions in real-time, since any transaction is endorsed before being added to the chain and validated. Additionally, HF ensures privacy, security, and confidentiality, making it more suitable for meeting the requirements of IoT applications.

Figure 7 shows an overview the layers of an architecture merging blockchain and model ontologies. It comprises three weakly coupled software layers: facade communication component, HF module, as well as HTemp and HClass ontologies. The facade communication component, seen as a set of interfaces, provides the unifying concept of service and a semantic description, i.e. hides the heterogeneity of the IoT devices. This layer works in a coherent and homogeneous semantic world linking the concepts defined in the HClass ontology with their real-world counterparts. The communication layer acts as an enterprise service bus (ESB), translating, each time, data generated by an authenticated IoT device to higher-level abstraction or creating commands, i.e. creating actions extracted from the smart contract.

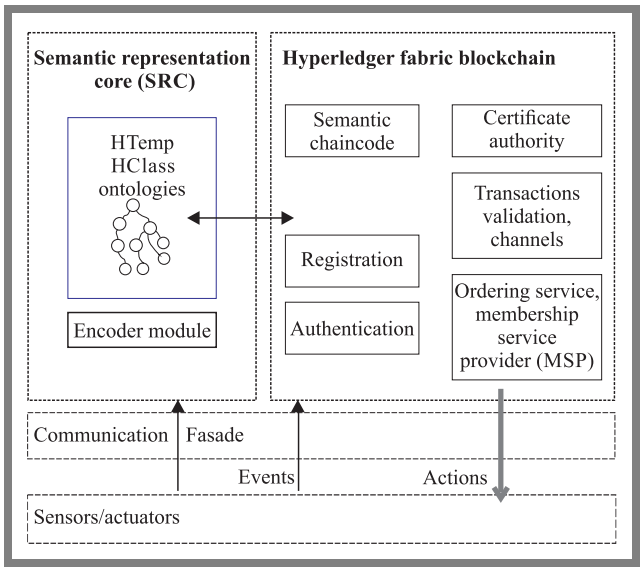



**Fig. 7.** Layers of an architecture merging blockchain and NKRL ontologies.

### *5.2. Development Execution Environment*

Thanks to the semantic abstraction level maintained by NKRL ontologies, the semantic representation core (SRC) module and the HF share the exact meaning of knowledge. The experiment conducted assumes that the consortium (participants) network consists of three organizations: robot, hospital, and home. Only the hospital cooperates with one peer (node) and one ordering node, while the robot and the home organization cooperate with two peers and two ordering nodes. The robot is responsible for setting up the blockchain network. It also has the privilege of creating channels and starts the ordering nodes. Only the channels between the robot and the home are private. Each channel has its ledger, which is replicated across other peers. These peers are integrated with the fabric network using certificate authority. Ordering peers receive blocks and generate validated transactions before committing a copy of the ledger to each peer. However, only the ordering peers within the robot and home organizations endorse peers, since they have the chaincode installed. The ordering and membership module (MSP) is the main components. Indeed, the MSP module maintains the cryptographic identities of all participants and links each IoT device to its identity. The ordering node allows the establishment of a consensus on transactions according to BFT algorithms. SMs deployed on channels running within the Docker generate an executable program.

The fabric SDK API is used to invoke the SMs from a client application. It allows to endorse transactions and to interact with the records on the blockchain ledger. The latter is composed of two components: the world state and the transaction log. The former represents a database of the ledger and is used to describe the state of the ledger at a given time. As for the transaction log component, it is the updated history for the world state.

The primary purpose of the authentication component is to ensure secure communication between network peers. This component relies on a public key infrastructure (PKI) to check the peers' cryptographic identities through authentication of a chain of trust and guarantee messages shared between peers involved in the interaction. By contrast, the MSP component uses the peer's public key to check each transaction that the peer should sign with its corresponding private key. Therefore, the identity checking mechanism enables, on the one hand, the node channels to establish MSPs to determine which IoT devices can perform actions. On

the other hand, it permits IoT devices to be trusted by each participant within the blockchain network.

Peers keep any data stored in the ledger. The IoT data are sent to endorsing peers and the facade communication component. After executing the SM, the peer responds to the application client if the transaction is endorsed (valid). The ordering service creates the block, and then the ledger is updated.

## 6. Implementation and Results

We implement the solution in three different environments of the same network. Fabric 2.3 was deployed as an underlying blockchain application, and we used node.js to write the chaincode and client applications. The proposed architecture ensures the homogeneity of the knowledge base. The relationships between real-world entities and their semantic representations are model-defined, meaning they allows for semantic matching. The model outputs the corresponding predicative occurrence. Therefore, the interface communication layer ensures a coherent representation of the real environment's states and the high-level abstraction. After executing the SM, and if the transaction is endorsed (valid), the peer responds to the application client. The ordering service creates the block, and then the ledger is updated.

After detection of the fall event, the corresponding chaincode installed on the robot peer is launched. Then, an authorization request to access the robot's camera is executed. Therefore, the communication layer enables converting the request into commands to access the robot's camera. Thanks to HF, the robot checks the hospital's identity using an MSP, abstracting all the cryptographic mechanisms, validating certificates and authenticating the user. After completing this process, the visual message action is endorsed, and the robot allows the hospital staff to access its embedded camera.

```
async writeData (ctx, key, time, sender,
           type, event, data) {
  const tmp =
     ID: key,
     SenderName: sender,
     SenderType: type,
     EventName: event,
     time: time,
     Data: data

 const buff = Buffer.from(JSON.stringify (tmp));
     await ctx.stub.putState (key, buff) ;
     return ctx.stub.setEvent (event, buff) ;
}

async readData (ctx,key)
      var response = await
ctx.stub.getState(key):
      return JSON.stringfy (response);
}
```

**Fig. 8.** Chainecode implementation.

HF is modular, pluggable, and allows different consensus algorithms (e.g. RAFT and byzantine fault tolerant) to be used. Furtheremore, HF relies, by default, on NoSql LevelDB to store public key values, and each entity has its own blockchain identity and registers only once. In the experiment, we used one ordering node only, because it can manage about 100 transactions per block.

In the experiments, we deployed the same chaincode business logic in the three channels connecting the endorsing peers of the network. The chaincodes implement mainly two functions to handle the reading and writing of data (Fig. 8).

### *6.1. Events Implementation*

To test the proposed implementation, we submitted 100 transactions from sensors to the ordering peer which batched and sent all of them to the anchoring peers. We measured the time between submitting a transaction, including the date write in the chaincode and its commitment to the ledger. Figure 7 shows the endorsing time of all the transactions.

The obtained results show that the transactions are endorsed in less than one second in most cases, and the mean duration to endorse a transaction is 819.77 ms. We repeated the same experiment with five sensors emitting 100 transactions at a throughput rate of five transactions per second. In Fig. 10,

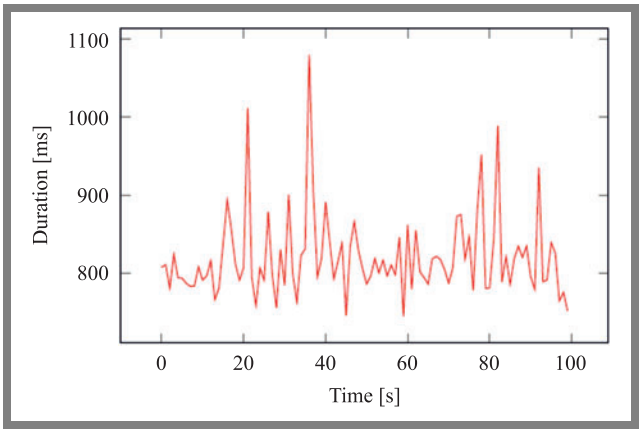



**Fig. 9.** Endorsing duration of 100 transactions from one sensor.

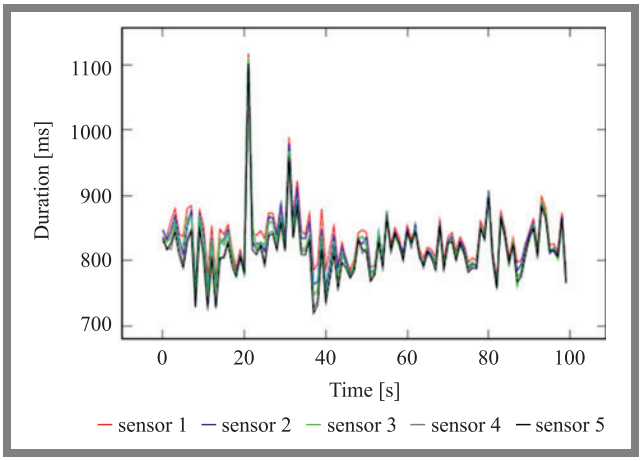



**Fig. 10.** Endorsing duration of 100 transactions from five sensors.

the resulting duration of the endorsement phase of this test is presented, with mean value duration equaling 824.37 ms. The broadcast chaincode events are captured by the applications connected to the channels using an event listener. The relevant action is inferred after parsing the payload data based on the event description and the sender type. The result of the inference is also submitted to the ledger by the creation of a transaction.

## 7. Conclusion and Discussion

IoT applications have to enforce security to control the data collected by scattered sensors. Privacy of data and monitoring of physical/virtual access to space or sensitive knowledge are among the many challenges faced by designers of distributed systems. Therefore, security and dynamic knowledge management are at the heart of the IoT application development process. The lack of interoperability and monitoring of physical/virtual access to space or sensitive data may endanger the adoption of the IoT paradigm. Nevertheless, it will be hard to ensure security and privacy for numerous IoT applications without providing a harvest ambient energy mechanism or reducing network latency (e.g. hyperledger fabric). Several studies show that qualitative and quantitative approaches have been adopted over the past years in connection with data processing, action and situation recognition [22]. Furthermore, many frameworks, projects, and techniques rely on a semantic mechanism to annotate and manage sensor data. The ontology web language (OWL) is de facto a solution that is most commonly used to express knowledge in IoT.

Several distributed applications, relying upon OWL, have been implemented. Even if these approaches offered some extensions and a built-in module to enrich the standard web semantic, their main weakness consisted in generating redundant knowledge descriptions. Many scenarios, such as those presented in this paper, need an $n$-ary structure that expresses actions/events and temporal properties. However, OWL and its variants have failed to address any proposals concerning the notion of $n$-relations. Thus, the entire semantic web language becomes de facto unsuitable for integrating heterogeneous IoT devices and addressing dynamic knowledge management requirements in IoT applications.

The work presented in this article aims to facilitate the implementation of access control for distributed systems. NKRL innovates by providing a hierarchy ontology of action. Indeed, the formalism we explore allows semantic descriptions of dynamic characteristics of the entities that frequently change overtime involved in IoT applications. Besides, we have proposed a semantic architecture relying on HF to ensure knowledge integrity and authentication while preserving privacy. Finally, the deployment of Fabric 2.3 as the framework's underlying blockchain did not negatively affect the response time. We have performed experiments to validate the time required for an action to take place and have evaluated the system's response time after observing the context, obtaining access to the camera request and executing the access command. The response time includes the processing time in the communication layer, as well as the time for generating the action and sending the command to an actuator device. This paper demonstrates how this approach ensures message integrity, verification, authentication, security and privacy, and how it allows semantic contextual knowledge to be shared in order to invoke one or more services. The use of a consortium blockchain in which not every peer has equal rights to endorse a proposed transaction is a potential disadvantage of the proposed architecture. Within the HF, only a few peers can validate transactions. Due to the fact that it is an emerging technology, we should explore the usefulness of the public decentralized blockchain principle.

## References


[1] A. Brunete, E. Gambao, M. Hernando, and R. Cedazo, "Smart Assistive Architecture for the Integration of IoT Devices, Robotic Systems, and Multimodal Interfaces in Healthcare Environments", *J. Sensors*, vol. 21, no. 6, 2021 (DOI: 10.3390/s21062212).

[2] J.M. Byeong, S.K. Sonya, and C. JongSuk, "Organizing the Internet of Robotic Things: The Effect of Organization Structure on Users' Evaluation and Compliance toward IoRT Service Platform", *IROS*, pp. 628–629, 2020 (DOI: 10.1109/IROS45743.2020.9340834).

[3] A. Kumari and S. Tanwar, "Secure data analytics for smart grid systems in a sustainable smart city: Challenges, solutions, and future directions", *J. Sustainable Computing: Informatics and Systems*, vol. 28, 2020 (DOI: 10.1016/j.suscom.2020.100427).

[4] B. Bhushan, P. Sinha, K.M. Sagayam, and J.A. Onesimu, "Untangling blockchain technology: A survey on state of the art, security threats, privacy services, applications and future research directions", *J. Computers Electrical Engineering*, vol. 90, 2021 (DOI: 10.1016/j.compeleceng.2020.106897).

[5] S. Pal, A. Dorri, and E. Jurdak, "Blockchain for IoT Access Control: Recent Trends and Future Research Directions", *CoRR*, 2021 (DOI: 10.1016/j.jnca.2022.103371).

[6] H.C. Chen, "Collaboration IoT–Based RBAC with Trust Evaluation Algorithm Model for Massive IoT Integrated Application", *J. Mob. Networks Appl*, vol. 24, pp. 839–852, 2021 (DOI: 10.1007/s11036-018-1085-0).

[7] S. Christos, E.P. Kostas, K. Byung-Gyu, and G. Brij, "Secure integration of IoT and Cloud Computing", *J. Future Generation Computer System*, vol. 78, pp. 964–975, 2018 (DOI: 10.1016/j.future.2016.11.031).

[8] –, (https://www.hyperledger.org/use/fabric)

[9] C. Îozgü and D. Yilmazer, "Improving privacy in health care with an ontology–based provenance management system", *J. Expert Systems, Special Issue: eHealth and Staying Smarter*, vol. 37, 2020 (DOI: 10.1111/exsy.12427).

[10] P. Gonzalez-Gil, J.A. Martínez, and A.F. Skarmeta, "Lightweight Data-Security Ontology for IoT", *J. Sensors*, vol. 20, 2020 (DOI: 10.3390/s20030801).

[11] L. Sabri and A. Boubetra, "Narrative Knowledge Representation and Blockchain: A Symbiotic Relationship", *Advanced Information Networking and Applications Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA–2020)*, pp. 320–332, 2020 (DOI: 10.1007/978-3-030-44041-1_30).

[12] B. Sejdiu, I. Florije, and L. Ahmedi, "A Management Model of Real-time Integrated Semantic Annotations to the Sensor Stream Data for the IoT", *WEBIST*, pp. 59–66, 2020 (DOI: 10.5220/0010111500590066).

[13] G.P. Zarri, "A knowledge representation tool for encoding the 'meaning' of complex narrative texts", *Natural Language Engineering*, vol. 3, pp. 231–253, 1997 (DOI: 10.1017/S1351324997001794).

[14] A. Reyna, M. Cristian, J. Chen, E. Soler, and M. Díaz, "On blockchain

and its integration with IoT. Challenges and opportunities", *J. Future Generation Computer Systems*, vol. 388, pp. 173–190, 2018 (DOI: 10.1016/j.future.2018.05.046).

[15] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: a technical survey on decentralized digital currencies", *J. IEEE Communications Surveys & Tutorials*, vol. 18, pp. 2084–2123, 2016 (DOI: 10.1109/COMST.2016.2535718).

[16] D.D. Fiergbor, "Blockchain Technology in Fund Management", *Communications in Computer and Information Science; Springer*, vol. 899, pp. 310–319, 2018 (DOI: 10.1007/978-981-13-2035-4_27).

[17] –, (https://www.ibm.com/topics/hyperledger)

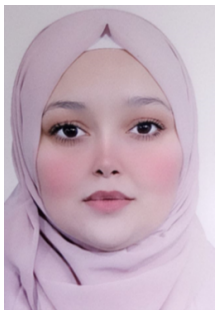
[18] –, (https://protege.stanford.edu/).

[19] F. Baader, D.L. McGuinness, D. Nardi, and P.F. Patel–Schneider, "The Description Logic Handbook: Theory, implementation, and applications", *Cambridge University Press*, 2010 (DOI: 10.1017/CBO9780511711787).

[20] J. Kwapień and S. Drożdż, "Physical approach to complex systems", *Physics Reports*, vol. 515, no. 34, pp. 115–226, 2012 (DOI: 10.1016/j.physrep.2012.01.007).

[21] Y. Li, L. Qiao, and Z. Lv, "An Optimized Byzantine Fault Tolerance Algorithm for Consortium Blockchain", *Peer-to-Peer Netw. Appl*, vol. 18, pp. 2826–2839, 2021 (dpi: 10.1007/s12083-021-01103-8).

[22] D. Hooda and R. Rani, "Ontology driven human activity recognition in heterogeneous sensor measurements", *J. Ambient Intelligence and Humanized Computing*, vol. 11, pp. 5947–5960, 2020 (DOI: 10.1007/s12652-020-01835-0).



**Manal Lamri** is a Ph.D. Student at the Faculty of Mathematics and Informatics, University of Mohamed El Bachir El Ibrahimi, Bordj Bou Arreridj, 34030, Algeria. Her research interests are in ontology, knowledge representation, reasoning, the Internet of Things paradigm, and blockchain technology.

E-mail: manal.lamri@univ-bba.dz
Faculty of Mathematics and Informatics, University of Mohamed El Bachir EL Ibrahimi, Algeria

**Lyazid Sabri** is an Assistant Professor at the Bachir Ibrahimi University. He holds a Ph.D. in Computer Science, Images, Signals and Intelligent Systems from University Paris-Est Créteil (UPEC). He is a Senior Consultant on Artificial Intelligence, Information Systems Architectures and Cybersecurity. While at UPEC, he worked on several collaborative research projects (e.g. SembySem, Web of Objects, Predykot, A2nets) dealing with IoT and ambient assisted living, robotics, artificial intelligence, and access management. His research interests are in ontology, knowledge representation and reasoning, robotics, artificial intelligence, deep learning, cognitive psychology, the Internet of Things paradigm, and blockchain technology.
E-mail: sabrilyazid@gmail.com, lyazid.sabri@univ-bba.dz, sabri@lissi.fr
Faculty of Mathematics and Informatics, University of Mohamed El Bachir EL Ibrahimi, Algeria
Laboratory of Images, Signals and Intelligent Systems, University of Paris-Est, France