Computing (2020) 102:2199–2228
https://doi.org/10.1007/s00607-020-00799-5

REGULAR PAPER

Performance analysis of the ubiquitous and emergent properties of an autonomic reflective middleware for smart cities

Jose Aguilar[1,2] · M. Jerez[1] · M. Mendonça[1,4] · M. Sánchez[1,3]

Received: 21 March 2019 / Accepted: 14 February 2020 / Published online: 30 March 2020

Abstract

One of the biggest challenges in a Smart City is how to describe and dispose of the enormous and multiple sources of information, and how to share and merge it into a single infrastructure, in a timely and correct manner. A Smart City requires computational platforms, which allow the interconnection of multiple and embedded systems, such that the technology is integrated with people, and can respond to unpredictable situations. The integration of information and communications technology in these spaces, allows exploiting the wealth of information and knowledge generated in a Smart City, and improving its planning and services offered to its citizens. In this way, the people are immersed in these spaces, which are aware of their presence (context-sensitive) and adapt to their needs. The context analysis in a smart city allows making available services and information, to support ubiquitously the activities of the individuals. This study aims to analyze the emerging and ubiquitous capabilities of an Autonomic Reflective Middleware. The Middleware is based on intelligent agents that can be adapted to the existing dynamism in a city for, ubiquitously, responding to the requirements of citizens. It uses emerging ontologies that allow, not only the adaptation to the context of the moment and in real-time but also responds to unforeseen situations.

Keywords Autonomic computing · Context-aware services · Emergent services · Middleware · Multi-agent systems · Emergent ontology · Smart city · Ubiquitous computing

✉ Jose Aguilar
aguilar@ula.ve

1 CEMISID, Universidad de Los Andes, Mérida, Venezuela

2 GIDITIC, Universidad EAFIT, Medellín, Colombia

3 Universidad Nacional Experimental del Táchira, San Cristóbal, Venezuela

4 Universidad Centroccidental Lisandro Alvarado, Barquisimeto, Venezuela

Mathematics Subject Classification 68Txx Artificial intelligence

1 Introduction

One of the main tasks of the smart cities is the integration of the existing computational decision systems, in order to exploit the wealth of information and knowledge generated, to improve their planning and public services offered to their citizens. The Information and Communications Technology (ICT) in Smart City can interconnect and automate virtually all the processes that occur in a city: infrastructure management, power consumption, communication system, traffic, health, among others, ubiquitously integrating all the elements of the city.

On the other hand, the computational systems in a smart city must consider that the context changes continually: situations, availability of services (based on ICT), as well as the requirements and preferences of the citizens. Therefore, they must react to this changing environment at runtime, considering that there are new needs and availability of services, emergent interactions, and situations, which cannot be predicted a priori. The continuous awareness of context and the gradual adaptation become the key to deliver value-added services based on ICT, to address the dynamics of continuously changing environments.

In general, the computational systems in a Smart City must exchange data, information, and knowledge permanently, which implies achieving semantic interoperability. This is some of the biggest challenges in a Smart City, in order to use the enormous and multiple sources of information correctly, with the adequate semantic representation, sharing them and merging them in a correct manner. The ontologies in a Smart City are an ideal tool to this end, making possible the communication between the different computational systems. However, the ontologies that participate in a Smart City must adapt to its changes, needs, and services. The objective of the ontological emergence is to generate emerging knowledge models [1]. The emerging ontologies will be a tool to address the needs of semantic interoperability in Smart Cities. Moreover, ontologies contain the knowledge needed to allow smart decisions oriented to improve the autonomy of the whole system. Essentially, the emerging ontologies will allow the system to cope with the next issues:

- Integrability and Interoperability of actors of the Smart City.
- The development of complex planning, where the negotiation is a main aspect.

In this work, we propose an architecture of Autonomic Reflective Middleware for Smart Cities (MiSCi), in order to allow making context-aware smart decisions in a smart city. This architecture was defined in a general way in [2]. In the present work is detailed the description of the MiSCi architecture, as well as is carried out one analysis of its Ubiquitous and Emergent capabilities, which are fundamental properties in a smart city [3]. Particularly, we evaluate the emergent and ubiquitous performance of the MiSCi platform in several case studies, in order to test its capacity to sense the environment, to reason, and execute a solution in an independent, proactive, context-aware and intelligent way, through emergent ontologies arising from the dynamics of the moment.

MiSCi is a multi-layer architecture based on the Multiagent System (MAS) paradigm, with the capabilities of such systems [4]: sociability, proactivity, adaptability, intelligence, etc. The autonomic reflective architecture is defined as a MAS, composed of agents that characterize individuals, devices and applications that coexist in the smart city, which provide services between them. Its architecture is based on web services, aware of context or not, which are consumed by the agents. The agents can create temporary or permanent emerging ontologies, which allow solving a particular situation, according to the context.

In specific, this study aims to analysis the emerging and ubiquitous capabilities of MiSCi, through intelligent agents that must adapt to the existing dynamism in a city for ubiquitously responding to the requirements of citizens, by using emerging ontologies that allow not only adapting to the context of the moment and in real-time, but also responding to unforeseen situations [3]. For example, in the vehicular system of a smart city, a meta-ontology can handle generic concepts about transport media, routes, etc. The ontologies that are handled by different applications of the vehicular system (such as, the public transport control system, the traffic management system, among others), can associate their concepts with meta-concepts in this meta-ontology. Then, if it is necessary to improve the transport in a situation of emergency, MiSCi must generate a temporary emergent ontology for the current situation. Principally, our contributions are:

1. An architecture in order to support context-aware smart decisions in Smart City contexts.
2. A meta-ontology that allows dealing with situations that are not considered in the knowledge bases (emergent situations).
3. The demonstration of the ubiquity and emergent capabilities of the proposed architecture, as well as the utility of the emerging ontologies.

This paper is organized as follows: the next section presents related works and a state of the art about architectures for smart cities; next, Sect. 3 describes MiSCi and its emergent behavior. Section 4 presents the scenarios used to test MiSCi. Later, in Sect. 5 the experiments and the result analysis are presented. Then, in Sect. 6 is presented a comparison with previous works, ending with some conclusions in Sect. 7.

2 Related work

Some works in the ontology area related to our work, are the following: in [5], the authors propose a semantic representation framework for unstructured data for smart cities. Also, they use fuzzy ontologies for the semantic recognition of information and services, solving the problem of ambiguous of the information from multiple data sources using a fuzzy representation. In [6] is presented a taxonomy for the smart cities, based on the next domains: natural resources and energy, transport and mobility, buildings, living, government and economy, and people. In addition, they propose policies and guidelines for the definition of strategies and planning actions in the Smart Cities, based on these domains. In [7], the authors discuss the impact of several applications of smart cities. This impact is analyzed according to their ontological and

technological features. Finally, they propose an Ontology for smart cities. In Pellicer et al. [8] provide a survey about the implementation and integration status of the main smart city applications around the world. These previous works highlight the use and application of ontologies in the implementation of various functionalities in smart cities, however, there are not concrete proposals about the use of dynamic and emerging ontologies in this context.

As well, some works have proposed architectures to build applications for smart cities [9, 10]. For example, [11, 12] present methodologies to define a smart city, which allows measuring its level of development. In Santana et al. [13] present the functional and no functional requirements for smart cities, and the new challenges in smart cities. Some of the functional requirements defined were: real-time applications, massive data processing, external data access, service management, among others. Some of the non-functional requirements defined were: interoperability, scalability, security, privacy, context awareness, adaptability, among others. Finally, some of the challenges defined were: energy management, definition of testbeds, platform maintenance, among others. Additionally, they propose a reference architecture for smart city platforms. These previous studies present requirements and challenges of the applications for smart cities, such as: real-time responses, processing of large amounts of data, interoperability, scalability, context awareness, adaptability, among others, which are some of the aspects that have been considered in our proposal.

In addition, different platforms have been proposed to manage the data, the services, and sensors, in a Smart City. For example, in [14] the authors propose a conceptual framework to address the issues that emerge from the citizens and public administration of smart cities and identify producers and consumers of data/service. The framework is built over a cloud infrastructure, for the development of contextual services for citizens, some of those services are Data, Analysis, Security, User Interaction, and Resource Management. In the infrastructure, the cloud approach allows dynamic storage and processing of data, necessary for smart city applications. According to the number of user queries in the cloud, the platform must provide a dynamic resource assignment to satisfy the QoS requirements. The architecture can be used as a Platform as Service (PaaS), or as a Software as Service (SaaS). In [9], Maciel et al. propose a middleware, called "Devices, Environments and Social networks Integration Architecture" (DESIA), which includes technologies such as social networks, cloud computing, and digital ecosystems. DESIA is divided into three layers: the individuality layer, the environment layer, and the cloud layer. The cloud layer manages the data integration from large groups of users and set of organizations. DESIA provides the next set of services: (1) collection, storage, and query of the context information; (2) situational inference; (3) user and environment interfaces; and (4) integration with external sources of data. These two previous works offer us approaches on how to manage the large amount of data for applications in smart cities, based on cloud services, without consider the applicative context, which is also part of our proposal.

In [10] was developed a Smart System Architecture for context-aware smart environments, which is composed of four layers: users, reasoning, ubiquity and physical. The physical layer contains the hardware parts of the system. The ubiquity layer acts as an intermediary to the system components and their utilization. The reasoning layer is where the logic resides. Finally, the user layer is where the information of the system

is presented to the users. In Roscia et al. [12] describe a MAS in a Smart City based on the Internet of Things (IoT), and show a case study in a Smart Grid. The proposal is based on a MAS in the context of an Intelligent Distributed Autonomous Smart City (IDASC). They use the ZEUS Framework to manage Knowledge and to implement the FIPA (Foundation for International Physical Agents) standard [4]. Both works present characteristics that serve as the basis for the proposal presented in this work, such as the definition of a multi-agent architecture, context sensitivity and ubiquity.

In [15] is proposed a middleware, called Civitas, to support the task of services for Smart City. They also analyze the main drawbacks of the traditional middleware oriented to services, and how Civitas overcomes them. In [16, 17] is proposed a middleware to offer urban open data to smart city applications. The middleware proposes mechanisms for the data integration for applications of smart cities. The GAMBAS middleware can support tasks such as efficient data acquisition, secure, privacy, data distribution, or data integration. In this way, it can reduce the development effort for smart city applications. In [18] discuss how the vision of the Future Internet (FI), and its particular components, IoT and Internet of Services (IoS), can help to define a unified urban-scale ICT platform for Smart Cities. They present a generic platform based on the Ubiquitous Sensor Network (USN) model, which fulfills basic principles of open, federated and trusted platforms (FOTs) at two different levels: at the infrastructure level (IoT, to support the complexity of heterogeneous sensors. deployed in urban spaces), and at the service level (IoS, as a suite of open and standardized enablers, to facilitate the composition of smart city services). These studies explain different strategies for implementing services and open data in smart city applications, as well as data integration, which is also part of the characteristics of our proposal.

On the other hand, some general works in smart cities, are the following: in [19] is introduced the concept of Cognitive Cities. They say that the smart cities only consider the efficiency of services to citizens, whilst that a Cognitive City must have three factors in the Urban Design: (1) Sustainability and resilience: the urban challenges, beyond efficiency, require learning and cognition; (2) Social Technology: citizens must be involved. The efficiency is achieved thanks to these actors; (3) Collective learning: especially learning about and by urban systems. In this way, they say that Cognitive Cities are a more appropriate concept. In [20], Fujiwara describes an example of a smart city, where the social public infrastructure is managed efficiently, using ICTs. Additionally, it shows trends in smart cities [21]. In [22], the authors present CityCoupling, an algorithm to determinate human mobility in a city, and predict what to in case of a disaster in the city. In Xu et al. [23] propose a destination prediction using a data-driven approach for a taxi ride, called DESTPRE. These are all new approaches, paradigms and challenges in the implementation of smart cities, which gives us ideas about the new requirements in these areas such as sustainability, resilience and collective learning, concepts that we consider can be supported, for example, in the ontological emergence.

Finally, other general works in areas as ubiquitous computing and context awareness, are the following: in Zambonelli et al. [24] describe the SAPERE platform, which supports emerging ubiquitous services using coordination mechanisms based on natural behavior. In Ghannen et al. [25] propose a methodology for the requirement engineering process for the "context-aware and adaptive systems", called "Require-

ment Engineering Context Awareness" (RE-CAWAR). The main contributions of this work are the methodology to develop the context models; and the ontological model that defines the context, which represents the typical scenarios where participants, activities, and operations, change over time.

The previous architectures presented in this section show some interesting particularities, nevertheless, they lack services to manage emergent situations, required in the smart cities. Also, they consider specific aspects, without a holistic view and integration of different aspects: emerging ontologies, real-time response, interoperability, scalability, adaptability, cloud services, context sensitivity and ubiquity, among others. Our MiSCi architecture mixes several paradigms, the MAS, the emergent systems, and the cloud computing, in order to integrate these aspects, and to allow ubiquitous not predetermined interactions between the different components of a smart city. Thus, some of the challenges presented in [26] are covered by MiSCi, mainly the heterogeneity and the context awareness, using emergent ontologies.

3 MiSCi architecture

In this section, the components of MiSCi are detailed, and its emergent behavior.

3.1 Multilayer architecture of MiSCi

MiSCi is a multi-layer architecture based on MAS. The agents of MiSCi perform an intelligent autonomic loop known as MAPE-K (Monitor-Analyze-Plan-Execute over a shared Knowledge) [27, 28], which allows perceiving the environment (monitoring it with the sensors available on it), thinking on an appropriate solution (services to be offered) according to the context, and planning and deploying the solution in the environment. In this way, agents can know the current state of the environment and take decisions to change it, in order to improve the quality of life and the comfort of the citizens. MiSCi takes advantage of the features of autonomic computing as the self-configuration using the MAPE loop, or of the reflexive systems as the introspection of the base level (see Fig. 1), in order to provide these services [2, 29].

The architecture of MiSCi consists of seven modules distributed through three levels. Also, it contains cross levels that bring support to the operations carried out by the agents of MiSCi. Each element of this architecture provides key features to the middleware, enabling the ubiquity, the context awareness, the ontological emergence, smart decisions, among other things. In general, the modules of MiSCi are:

1. *MAS Management Layer (MMAL)* This layer is an adaptation of the FIPA standard [4]. MMAL defines the rules that allow the coexistence and administration of a society of agents, encouraging the interoperability with other technologies. The Agents in this layer are: Agent Management Agent (AMA), Communication Control Agent (CCA) and Data Management Agent (DMA).
2. *Service Management Layer (SML)* This layer is a key feature in the MiSCi architecture, because it makes possible the integration between the MAS and SOA paradigm in a bidirectional way; that means that agents can register, discover, and

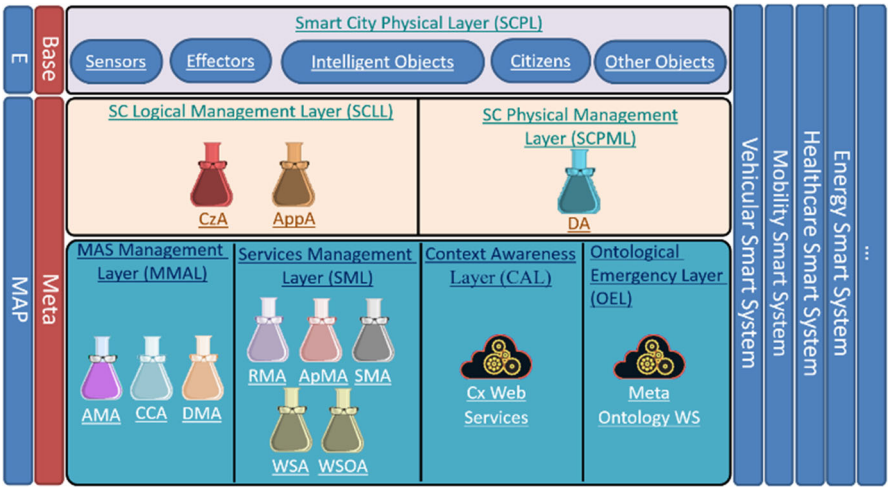

Fig. 1 MiSCi architecture [2]

consume web services in the cloud, and vice versa (agent's tasks are offered as web services). This feature makes possible to use the SaaS model of cloud computing in MiSCi, which is fundamental in a Smart City. The agents of this layer are: the Services Management Agent (SMA), the Web Service Agent (WSA), the Web Service-Oriented Agent (WSOA), the Resource Management Agent (RMA) and the Applications Management Agent (ApMA) (see [30, 31], for more details about these agents).

3. *Context-Awareness Layer (CAL)* The purpose of this layer is to offer context services, for the information management about the context. This information is managed in a cycle that is composed of: the discovering of the Context, modeling, reasoning, and the distribution of the context. It is also important to know the quality of the information about the context (quality of context), based on metrics as timeless, usability, and security level [26]. For this layer, we take as reference the services defined in [32], where it is proposed a Context-Aware Reflective Middleware based on Cloud Computing.
4. *Ontological Emergence Layer (OEL)* This layer is based on [1, 33], where are defined semantic services, such how the services for automatic updating of ontologies, to adapt them to the dynamics and the evolution of an environment, among others. More details about this layer are given in Sect. 3.2.
5. *Smart City Logical Management Layer (SCLL)* This layer is where all the applications (software, virtual objects, etc.) and persons present in the Smart City are characterized. Basically, each element/person is characterized by an agent (is an abstraction of it), which contains metadata that defines its properties. This layer contains agents like CzA (it characterizes each citizen in a smart city) and AppA (it characterizes the applications in a smart city, such as a Vehicular Smart System, a Healthcare Smart System, etc.). They are aware-context agents, which coordinate and cooperate with each other, to make decisions that are needed at a particular time, to help the people to perform their tasks in the smart city, encouraging the

comfort and improving the quality of life of the citizens. The CzA and AppA receives information from the sensors through the Device Agents (DA). They process this information to determine the different situations that arise in a smart city.

6. *Smart City Physical Management Layer (SCPML)* It allows managing the physical devices in a smart city. In this layer, all the physical elements of the environment are characterized, allowing their use and communication with each one of the other agents of MiSCi. Thus, each physical device (sensors, effectors, smart objects, etc.) is characterized by an agent (is an abstraction of it), which contains metadata that defines its properties. Some of these physical devices are intelligent (smart objects), so that the properties of learning, autonomy, reasoning, among others, are critical to characterizing them. In general, they can carry out tasks of data analysis, event identification, etc. This layer communicates with the real physical device that is in the next layer.
7. *Smart City Physical Layer (SCPL)* This layer is where is deployed the smart city itself. In this layer is where all the physical components of the smart city are deployed, such as: (a) Sensors, to capture the useful information for services and smart objects in the environment; (b) Effectors, to modify the physical conditions of the environment; (c) Smart Objects, which are components of the smart city that may adapt and respond to situations in the current context.

3.2 Some remarks about the MiSCi architecture

The shaping of the layers allows reaching the middleware objectives due to the interaction between them. The SML and MMAL layers provide the next set of operational services to the SCLL and SCPML layers, which allow the agents of these two layers to meet the objectives for which they were designed:

- Know the services, agents, and applications available on the platform, grouped by levels, types, or any other classification (these tasks are performed by the AMA, RMA, APMA, DMA, and SMA).
- Know the agents providing service and where they are located (these tasks are performed by the AMA and RMA), as well as the web services available in the cloud that can be invoked by agents of the MAS (task carried out by SMA).
- Know the ways to access the services, and allow its invocation (WSA carried out this task).
- Control the mobility of the agents (task performed by the AMA).
- Allow the communication between the agents (task performed by the CCA).

Some of these features are provided by MMAL [33], and others by the agents of SML [30, 31]. The SML supports the SaaS model of cloud computing in MiSCi, enabling the MAS-SOA (Service Oriented Architecture) integration.

The CAL and OEL layers work together to provide contextual services (by making emerge the ontologies of context, domain, and components) to the agents of the SCLL and SCPML layers, so they can act intelligently and properly, to resolve the particular situations that arise in the city in a specific place and time. Thus, these two layers provide ubiquity's benefits to the middleware.

In the same way, MiSCi includes the main sub-systems of a smart city, such as: the Vehicular Smart System is responsible for the control of traffic, the Mobility Smart System that manages the mobility of citizens (public transport), the Smart Healthcare System in charge of facilitating the access to health services, among others. The agents of MiSCi can communicate with these systems, through the AppA agents, because they characterize the applications of the smart city in the architecture. In this way, the global systems can be coordinated with the local systems, to meet the needs of the citizens in a given moment.

The autonomous capability of MiSCi is based on the Autonomic Computing loop known as MAPE [27]. The MAPE loop is in charge of monitoring, analyzing, planning and executing the plan, adjusting the environment to the current needs, in order to detect and respond to the different situations in the Smart City. Each agent of MiSCi has a role in the processes of the MAPE loop. For example, the monitoring process is performed by agents DA, AppA and CzA. The AppA and CzA perform the processes of analysis and planning, in conjunction with the context-aware services and the meta-ontology. The execution of the plan is performed in the SCPL layer by using the effectors (DA).

On the other hand, the reflexive capability of MiSCi is in the Meta level of the middleware and specifically occurs in the SCLL, SCPML, CAL, and OEL layers, while the base level (the system functionality) corresponds to the SCPL layer. The reflexive property is based on the ontological emergence, which allows the emergence of a semantic representation to adapt it to the current requirements in the smart city. More details of the design of MiSCi can be found in [2], and a recent extension based on fog computing in [29].

3.3 Ontological emergence process

A Smart City requires the interconnection of multiple systems, which must be interconnected to support their activities and to respond to unpredictable situations. They must exchange data, information, and knowledge, permanently. For this, it is necessary that they can understand each other, know their context, manage a common language, among other things, which implies achieving semantic interoperability. The ontologies in a Smart City are an ideal tool for this, making possible the communication between different systems.

In general, the ontologies that participate in a smart city are distributed (several servers), heterogeneous (diverse data structures, languages and data types) and dynamics (that is, they must adapt to the changes, needs, and services of a smart city). In this case, the objective of the ontological emergence is to manage all information and knowledge that can be generated in a smart city, through processes of monitoring, analysis, and semantic enrichment. The emerging ontologies are a way to address the needs of the dynamic semantic interoperability.

The objective of the ontological emergence is to manage all the information and knowledge that can be generated or can enter the smart cities, creating new models of knowledge that allow to efficiently manage the needs of the city. Through processes of monitoring, organization, analysis and semantic enrichment of the different onto-

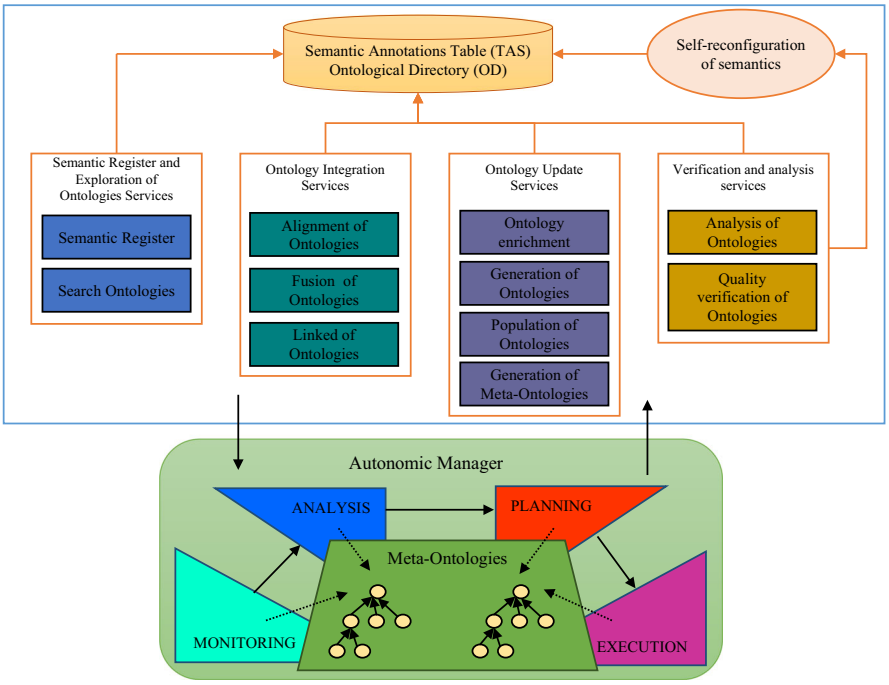

Fig. 2 Services and structure of the OEL

logical models that are handled, new emerging models can be generated that will be used by smart city services to be more efficient.

The ontological framework of MiSCi is composed of distributed ontologies that are managed by different agents and services involved in the Smart City, and a meta-ontological core. The ontologies are heterogeneous, since each source of knowledge has its own designers, and usually, they have different designs and structures. The agents of the Smart City use their local ontologies for their services, to interact with other agents, etc. When they detect the need to update their knowledge (e.g., the coming of a new service, changes in the behavior pattern of the components, context change, etc.), the agents request the services of the OEL. The general architecture of OEL is shown in Fig. 2.

The Meta-Ontology is managed by the OEL, which provides ontological services to the agents of the system to allow the ontological emergence. The Meta-Ontology is used to define the general model of the components and domains present in a smart city. In particular, this meta-ontology links and structures the domain and context ontologies, and organizes the semantic information from different data sources that arise in a smart city, due to the incorporation of new devices, agents, services, software system, etc.

The purpose of the OEL layer is to provide a set of services with very specific tasks for handling ontologies. These services are based on [1, 33]. Thus, the objectives of the services offered by the OEL are (for more details of the OEL services see [1]):

- Provide Services for self-management of ontologies,

- Manage the semantics of the smart city, through ontological mining services,
- Model new objects and behaviors,
- Maintain the consistency and semantic evolution of the smart city.

This layer carries out the self-management processes of the ontologies, for which, it is based on the meta-ontology that defines a generic model for the composition and updating of the ontologies, required by the different agents of the Smart City. Also, this layer allows monitoring the ontologies to adapt them to new requirements, defining the elements that must have the ontologies, and adjusting and self-managing the ontological framework of a Smart City. This layer contains an Autonomic Manager that executes a MAPE-K [1, 33] with the aim of discovering useful knowledge from the data generated in a Smart City and saving it as knowledge bases for populating the meta-ontology. This knowledge must be used by agents to make smart decisions. Particularly, MAPE-K paradigm allows the definition of autonomic cycles of data analysis tasks (ACoDAT), like the proposed in [34, 35], which are intelligent control loops for process supervision that allows reaching strategic objectives in a smart city (it is a MAPE loop). An ACoDATA integrates a set of data analysis tasks that act autonomously and collectively to achieve the strategic objectives pursued by it. Each task interacts with the others and has a specific role in the cycle [34, 35]: Observing the supervised process, analyzing and interpreting what happens in it, and making decisions that allow reaching the objective for which the cycle was designed. In this layer, the MAPE-K is used to supervise the context and to allow the ontological emergence. This loop is detailed below (see Fig. 2):

- *Monitoring* It detects the ontological service requests in the smart city. The monitoring phase identifies the semantic events, the new conceptual models, the ontologies entering or leaving the smart city, etc. It is implemented mainly through the "Semantic Register" service, which generates a semantic directory of the different conceptual models (documents, databases, ontologies) in the smart city.
- *Analysis* in this phase is analyzed the data in the semantic register, to detect the need to include new concepts, new categories, and new conceptual models. This process of analysis can be guided by meta-ontologies, which offer a general classification of categories of concepts. It is based on the "Ontological Analysis" service, which determines the type of ontological need, and the "Quality Verification" service, which detect any inconsistency or redundancy in the ontologies, in order to apply the necessary adaptations.
- *Planning* Here the actions to be followed are evaluated for the resolution of the situations detected in the analysis. It determines the services to be implemented to meet the ontological need of the smart city. Here are defined the tasks of data mining, ontological mining, etc., to perform.
- *Execution* In this phase are executed the different services planned, among which can be the services of "Integration of Ontologies" and "Ontologies Update".

In general, the services offered by the OEL are (for more details about these services, see [1] and Fig. 2):

1. *"Semantic Register" and "Search Ontologies" Services* These services allow knowing, publicizing and locating semantic information in a smart city. Through the "Semantic Register" service, the agents of a smart city record information about the conceptual models that handle. In this process, a "Semantic Directory" (SD) is generated, with semantic annotations of the sources of knowledge (at level metadata) used in the smart city. The SD consists of a "semantic annotations table" where the semantic events are recorded, and an "Ontological Directory" with the ontologies available in the smart city. The "Search Ontologies" service allows exploring the ontological directory.
2. *"Ontology Update" Services* These services allow creating new ontologies or enriching the already existing. The services are: Generation of new ontologies and meta-ontologies, meta-ontological population, and ontology enrichment. Through the "ontological population" service may be instanced in the ontological model with the current information about the context. In this ontological model are considered data on the user's profile, and activities or events that take place, considering the context in which they occur: the spatial domain (places, objects, etc.) and the functional domain (health, economy, energy, etc.). In each domain can also exist meta-ontologies to define generically a conceptual model, maintaining the integrity and consistency between the concepts, showing global information about how they should be structured and how they relate to each other. The "ontology enrichment" service allows updating an existing ontology with new instances. Finally, the "generation of new ontologies and meta-ontologies" service allows the emergence of new ontologies, following a procedure defined in [1].
3. *"Ontology Integration" Services* It is a set of ontological mining services that can be required during the emergence ontological, in order to align, link or merge ontologies.
4. *"Verification and Analysis" Services* These services check the quality of the ontologies. It determines, for example, the ontologies in the same domain, the correspondence between them, or new knowledge associated with a given domain, among other things, based on the semantic mining. To verify the quality of an ontology, are considered two criteria: Coherence and Redundancy. These services are the analysis of the ontologies and the quality verification of the ontologies.

In general, the agents involved in a smart city register information about the system events, and about the ontologies that handle, making semantic annotations of the sources of knowledge (in terms of meta-data) used. In this process of semantic registration, they relate the concepts and ontologies with some meta-concepts, through their description or properties. Before performing the "Register of Semantics" service, we can view the meta-concepts of the Meta-Ontology, through the "Query of Concepts" service. These concepts have the role of super-classes for concepts that are handled and emerge in the smart city. The process of ontological emergence occurs when are performing semantic mining tasks among registered ontologies and meta-ontologies.

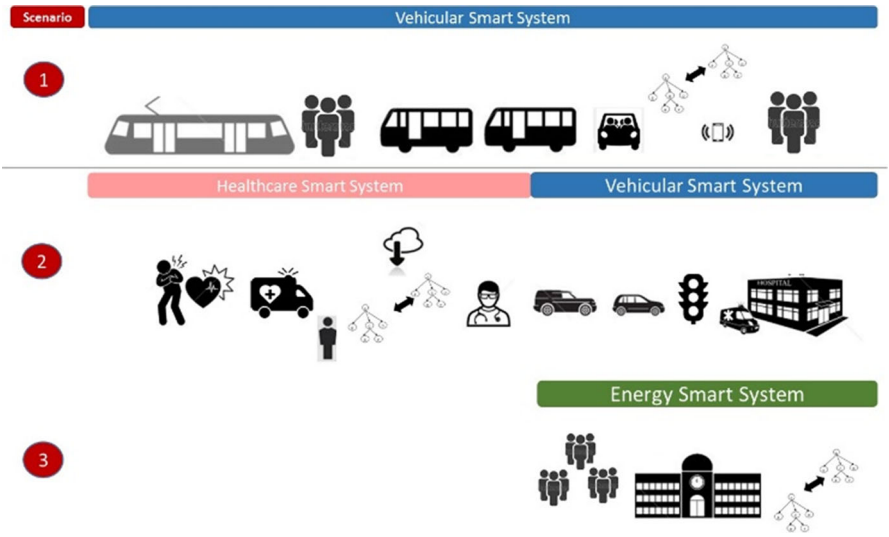

Fig. 3 Graphical description of the scenarios [29]

4 Performance evaluation of MiSCi

4.1 Definition of the test scenarios

In this section, we present three possible scenarios of use for checking the capabilities of ubiquity of MiSCi, as well as the ability of its agents to find solutions to the problems that arise in the city in a smart way, by using the elements of contextual awareness and ontological emergence of the middleware.

These scenarios are based on the same presented in [29]. The three scenarios happen in parallel (see Fig. 3), this will allow demonstrating the capabilities of MiSCi to provide ubiquitous services to the citizens.

The first scenario arises in a subway station, in which a system failure happens, making impossible to send trains to this place. The smart city will have the capability to deal with this kind of situation, and activate another mobility system, to allow citizens to reach their destination without major problems.

At the same time that the subway fails, a citizen located a few blocks away is suffering a heart attack (his body sensors activate the alert). The smart city should be able to provide priority services to this citizen and transport him to the closest hospital as soon as possible. However, this situation is complicated because citizens are leaving the subway to go to an alternative transport system, so, there is more traffic in this part of the city. This situation represents a difficulty to the ambulance that goes to pick up the patient.

The third scenario is given in a nearby shopping center, where some citizens who left the subway due to the failure, decide to stay in this mall for shopping. Because the shopping center is receiving a number of people greater than normal, it will activate some services to provide environmental and energy conditions necessary for citizens to feel comfortable in that place.

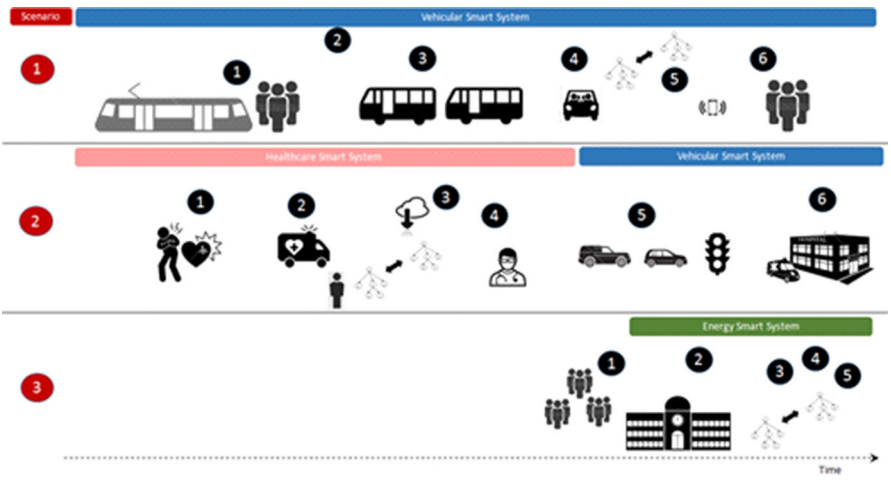

Fig. 4 Behavior of MiSCi in each scenario: describes the behavior of the components of MiSCi for each scenario

4.2 Description of each scenario

4.2.1 Scenario 1

The rail system in each station is monitoring and predicting possible events, through the connection with other agents. For example, a greater number of users of the rail system must generate more frequency of trains.

At the time of failure of the subway station, it is not possible to send trains to provide the service. MiSCi must activate other mobility systems for citizens to reach their destination. The next listing describes the situation (see Fig. 4 for more details).

1. Some DA (sensor) detects a failure on the segment of the road that connects to this subway station.
2. DA informs the failure of the AppA agent that characterizes the Vehicular Smart System (VSS).
3. The AppA of the VSS generates the "sending buses" service to the subway station to mobilize passengers, and further, request services of contexts to determine other mobility systems available.
4. The *services of context* information about the Carpooling services available and determine the need to activate a daemon application that can suggest transportation options for CzAs.
5. The daemon application requests the "ontologies search" service of carpooling, and a *strong fusion* is done with the transport system ontology, to generate a complete ontology (temporary emerging ontology) with all the available types of transportation, which is used by the application in order to suggest to the CzAs the available options.
6. The citizens will have the option to accept or reject the suggestions of the application.

4.2.2 Scenario 2

The agents of the citizens (CzA) have the personal preferences and physical conditions of each citizen and can send these data to the cloud. In addition, these agents can monitor their health data in real-time.

When a citizen suffers a heart attack near the subway station, the smart city must provide the best service to attend the citizen, e.g. organize vehicular traffic to move it quickly by ambulance to the medical center. MiSCi must activate its systems to reach this goal in the next way (see Fig. 4):

1. The CzA detects vital signs of the citizen and generates the alarm signal to the Healthcare Smart System (HSS), through the AppA that characterizes that system.
2. The HSS actives the ambulance service to treat the patient and move it to the closest medical center.
3. The HSS must obtain *context information* regarding vital signs, obtained by the sensors (blood pressure, respiration, temperature), as well as the information about the patient's history to know their background, previous illnesses, disease ancestors, current treatments, habits, allergies, etc. This information is provided by the CzA.
4. The HSS requests the service of *contextual information* about possible diagnosis and treatment suggestions that may provide the medical agent that comes in the ambulance. For this, it is requested "ontologies search" service of the health domain, concerning the pathology presented by the patient.
5. The "Ontological Emergence" Service determines the need *to align* the "patient's vital signs" ontology with the "diseases and symptoms" ontology, obtained by the previous service, to be used by a recommender system, which provides the information to the paramedics about of possible treatments or immediate steps to treat the patient.
6. The HSS requests the VSS the traffic management, for a rapid transfer of the ambulance to the Medical Center.
7. The VSS synchronizes the semaphores and alerts the vehicles and buses for clearing the way to the medical center.

4.2.3 Scenario 3

The Mall agent suggests possible actions to customer that access to the mall, and communicate with other agents, like the agents of the different stores in the Mall (e.g., restaurants), and especially, with the smart energy grid system, to give them information about the new users of the Mall (the group of citizens leaving the subway station that decide to enter into this shopping center). MiSCi acts in the next way:

1. The AppA that characterizes the Shopping Center (SC) detects the increase in the number of customers in the SC and requires adapting the environment to provide better and adequate services to the citizens: air conditioners, more lighting, advertising, etc.

2. The AppA (SC) determines the need to negotiate with the Energy Smart System (ESS) to get more energy to offer these services for. ESS activates the *context services* of the MiSCi, to deal with this trouble.
3. The MiSCi, through the *context services*, requests for the ESS information about the available energy suppliers, to carry out the negotiation for additional energy.
4. The "ontologies search" service is requested in the energy domain, to have information about the energy components.
5. The "Ontological Emergence" Service, considering that only part of the information is needed, in this case about the energy suppliers, makes a *weak fusion* (temporary) between the ontology about the components of energy and the AppA (SC) ontology, in order to generate the necessary knowledge about what would be the best options about the energy suppliers, and perform negotiations to request in a smart way the energy required.

4.3 Simulations

In this section, some simulations are designed aiming to test the internal components of MiSCi, as well as to determine how some components, such as the Internet and local network speed, influence the performance of the platform, which is very important for the ubiquity property. We have designed three experiments, in which the following simulation variables can be changed:

- Avg. time for the generation of traffic information requests by devices: It is the mean used to generate traffic requests in the DA (Traffic Sensor), which uses an exponential distribution (request like routes, traffic data needed by HSS to coordinate the traffic, etc.).
- Avg. time for the generation of traffic information requests by the citizens: It is the time used by Agent CzA (Pedestrian) to simulate requests to the VSS (request of Uber, Carpool, etc.). These requests are generated using an exponential distribution.
- Avg. time for ACL message delivery: This parameter is used to simulate the local network speed, is used in the WSA and WSOA nodes to measure the time taken for a local message to be delivered. The time is simulated by using an exponential distribution.
- Avg. service time of the HSS: parameter used to simulate the time took by the HSS to process the data in the cloud, using a normal distribution (the deviation standard is also a parameter of the simulation model).
- Avg. service time of the VSS: parameter used to simulate the time took by the VSS to process the traffic data in the cloud, using a normal distribution (the deviation standard is also a parameter of the simulation model).
- Internet's upload latency (in units of time): Parameter used to simulate the time of sending requests from MiSCi's agents to the services in the cloud. An exponential distribution is used for this purpose.
- Internet's download latency (in units of time): Parameter used to simulate the time taken to send a response message from the services in the cloud to MiSCi's agents. An exponential distribution is used for this purpose.

- Avg. time for the generation of first aid requests: parameter used to simulate emergencies requested by the citizens through the CzA (Patient). An exponential distribution is used for this purpose.

We have used the GLIDER Simulator [36], to create and run the simulation model for MiSCi. GLIDER outputs statistical data for each simulation entity, which can be used to determine the behavior of the system under certain conditions. However, to understand the format of the output, we need to define the next variables:

- *Node.el (External List)* This represents the queue of the node (requests waiting to be served).
- *Node.il (Internal List)* The internal list, this represents the requests being served.
- *#Ent* On the external list indicates the number of requests that arrived in that node; on the internal list indicates the number of requests that were served by the node.
- *Lght* On the external list represents the queue length at the end of the simulation; on the internal list indicates if the node was serving a request when the simulation finished (regularly is 0 or 1).
- *Max* On the external list, it indicates the maximum length of the queue during the entire simulation time. On the internal list, it's 1 when at least 1 request was served by the node.
- *Avg* on the external list indicates the mean length of the queue; on the internal list indicates the workload percent of the node.
- *MaxSt* on the external list determines the maximum time lasted by a request in the queue, on the internal list it represents the maximum service time of the node.
- *MeanSt* on the internal list represents the avg. time taken by requests in the queue, on the internal list indicates the service time mean of the node.
- *TFree* For the external list indicates the time without queue on the node, whereas on the internal list it determines the idle time of the node.

4.3.1 First simulations

Goal The main goal of this experiment is to test the behavior of MiSCi with high first aids requests (ex: a catastrophe caused by some natural phenomenon), high traffic requests (ex: the failure in the train station system), high requests to the VSS by citizens (the failure in the train station produces that citizens request more information to the VSS), high resources of data processing for the HSS as well as for the VSS (low service times imply high data processing in the cloud and vice versa), and an acceptable internet and local network speed. Likewise, the scalability of the system will be tested against a large amount of traffic and health data generated. These simulations allow for emulating scenarios 1 and 2.

Parameters In the first experiment, we used the following values for the parameters of the simulation model:

- Internet's upload latency: acceptable, 200 ms.
- Internet's download latency: acceptable, 200 ms.
- Avg. time for ACL message delivery: regular, 200 ms.

- Avg. time for the generation of traffic information requests by devices: high, create 1 request every 0.7 s.
- Avg. time for the generation of traffic information requests by the citizens: high, creates 1 request every 0.95 s.
- Avg. time for the generation of first aid requests: high, creates 1 request every 5 s.
- Avg. service time of the HSS: low, serves 1 request every 0.9 s with a standard deviation of 0.1 s.
- Avg. service time of the VSS: low, serves 1 request every 0.8 s with a standard deviation of 0.2 s.

4.3.2 Second Simulations

Goal The main goal of this experiment is to test how a slow internet connection influences MiSCi's performance.

Parameters This experiment is a sub-case of the first one, and the same parameters are being used, but an extremely slow internet connection will be simulated, establishing a latency of 5000 ms (upload and download).

4.3.3 Third Simulations

Goal The goal of this experiment is aiming to measure the influence of the local network on MiSCi, as well as in the WSA and WSOA agents, against a high demand for requests of traffic and emergencies. In addition, there is high Internet speed and good resources of data processing in the cloud for both sub-systems (HSS and VSS). Again, these simulations allow emulating scenarios 1 and 2.

Parameters

- Internet's upload latency: low, 50 ms.
- Internet's download latency: low, 50 ms.
- Avg. time for ACL message delivery: high, 500 ms.
- Avg. time for the generation of traffic information requests by devices: high, create 1 request every 0.7 s.
- Avg. time for the generation of traffic information requests by citizens: high, create 1 request every 2 s.
- Avg. time for the generation of first aid requests: high, creates 1 request every 5 s.
- Avg. service time of the HSS: acceptable, serves 1 request every 5 s with a standard deviation of 0.8 s.
- Avg. service time of the VSS: low, serves 1 request every 0.1 s with a standard deviation of 0.08 s.

4.4 Analysis of the results

4.4.1 First simulations

The results of this experiment are shown in Table 1. The process was run in a time of 2 h (7200ss), during this time were generated 7166 traffic requests by Citizens and 1463 first aid requests. We can see in Table 1 that the node with the highest workload is the VSS (Endpoint), which only has served 8980 requests (vSSEndpoint.il's #Ent) of 17,982 received (vSSEndpoint.el's #Ent). Moreover, its queue reaches a maximum length of 9004 requests, with a max. time in queue close to 3600 s (almost 1 h) and free time in its internal list lower than 2 s., while its max time for service is 1.593 s. From the last 3 rows of Table 1, it can be noticed that the average time taken by MiSCi to solve a traffic request is 1769 s, however, this is not because of MiSCi. It can be noticed that VSS has a long queue of requests because the cloud resources that VSS is using are not enough to deal with the high quantity request received. We can conclude that because it is clear that VSS is in the cloud, and no other node is presenting issues, then it means that the issue is in the cloud, not in MiSCi. From this result, we can see that to properly handle all requests when a problem or failure occurs in the vehicular system, it is necessary to have good resources for data processing in the cloud, and to have algorithms that are able to get a solution in a very short time (see scenario 1). Also, we can see that the workload of the components of the MiSCi bridge (WSA) among the MAS and the cloud (WSOA) is good because they do not present any queue, and its mean time for service is less than 0.203 s (see rows 1–4, 13–16, of Table 1).

4.4.2 Second simulations

From Table 2, we can deduce that having a slow internet connection affects significantly the entire process as well as all the smart sub-systems since messages sent from agents to services in the cloud will take a long time to arrive, with an avg. time closest to 3380 s (internetUpload.el's MeanSt), and a maximum time of 6672 s (internetUpload.el's MaxSt). It is also clear that while most components of MiSCi remain idle (see the T.Free column), the internet node is busy all the time, which raises the total time of system's response 3400 s (dAEffTraffic's Mean) for traffic requests and 3725 (DAAmbulance's mean) for the health system. It can be a problem in scenario 2, where real-time response is required.

4.4.3 Third simulations

In this experiment, we can observe that when we set a high latency in the local network, it has an influence on the response time of MiSCi, as is shown in the last two rows of Table 3: the total avg. time to solve a traffic request took more than 8 min (503.082 s) while that the total time to solve a request for Ambulance, is close to 1 min (60.627 s). It means 10 min in total to get an ambulance and trace a route to the closest hospital, which can represent a high value in emergency situations. Also, there is congestion in the WSA (VSS). It means

Table 1 Results of the first experiment

Node.Lst	#Ent	Lgth	Max	Avg	MaxSt	MeanSt	T.Free
WSA(VSS).el	17,982	0	13	0.507	3.753	0.203	5412.237
WSA(VSS).il	17,982	0	1	0.494	2.858	0.198	3637.821
WSA(HSS).el	1463	0	2	0.001	0.850	0.008	7188.486
WSA(HSS).il	1463	0	1	0.040	1.524	0.196	6911.794
InternetUpload.el	19,445	0	15	0.642	3.240	0.237	5077.886
InternetUpload.il	19,445	0	1	0.540	1.832	0.200	3307.961
VSSEndpoint.el	17,982	9002	9004	4521.998	3595.781	1789.892	1.988
VSSEndpoint.il	8980	1	1	0.999	1.593	0.801	1.164
HSSEndpoint.el	1463	0	4	0.022	2.747	0.111	7051.322
HSSEndpoint.il	1463	0	1	0.183	1.211	0.901	5881.408
InternetDownload.el	8630	0	4	0.024	1.419	0.020	7038.397
InternetDownload.il	8630	0	1	0.238	1.776	0.199	5480.954
WSOA(EffTraffic).el	7167	0	3	0.011	1.526	0.011	7122.753
WSOA(EffTraffic).il	7167	1	1	0.198	2.425	0.199	5772.362
WSOA(Ambulance).el	1463	0	1	2.870E−4	0.517	0.001	7197.933
WSOA(Ambulance).il	1463	0	1	0.039	1.593	0.195	6913.748
SkippedData	1812			1769.070	3595.191		
DAEffTraffic	7166			1797.155	3596.984		
DAAmbulance°	1463			2.091	5.060		

that low-speed local network can generate issues in scenarios 1 and 2, in order to accomplish with the real-time and timely information requirements of Smart Cities.

5 Evaluation of the emergent and ubiquitous behavior of MiSCi

5.1 Metrics

The semantic services provided by OEL, are required by CAL to update and adapt the context to the dynamic and the evolution of the environment. The contextual data collected from the environment, and in special, the produced data in the emergency event, can prone to imperfection due to limitations in sensors [26]. The "Quality of Context (QoC)" service is used to measure the quality of any information that is used as contextual information. In Shagarbi et al. [26] propose different QoC metrics: reliability, timeliness, significance, usability, and representation consistency, among others. In this work, we are interested in the timeliness and usability metrics, in order to evaluate the ubiquitous and emergent capabilities of MiSCi. The timeless metric allows determining the improvement in the response time of the services using EO, if the service is offered at the best possible time or at the right time. Thus, it measures the

Table 2 Results of the second experiment

Node.Lst	#Ent	Lgth	Max	Avg	MaxSt	MeanSt	T.Free
WSA(VSS).el	17,988	2	14	0.493	3.671	0.197	5438.910
WSA(VSS).il	17,986	1	1	0.497	1.864	0.199	3617.718
WSA(HSS).el	1376	0	2	0.001	0.614	0.007	7190.096
WSA(HSS).il	1376	0	1	0.039	1.384	0.206	6916.501
InternetUpload.el	19,361	17,928	17,928	9038.593	6672.949	3380.053	0.117
InternetUpload.il	1433	1	1	0.999	32.320	5.025	0.015
VSSEndpoint.el	1321	0	4	0.015	2.841	0.082	7101.320
VSSEndpoint.il	1321	0	1	0.147	1.472	0.805	6135.993
HSSEndpoint.el	111	0	0	0.0	0.0	0.0	7200.0
HSSEndpoint.il	111	0	1	0.013	1.190	0.896	7100.451
InternetDownload.el	1184	0	34	5.264	170.317	32.015	2034.904
InternetDownload.il	1184	1	1	0.850	39.156	5.177	1074.319
WSOA(EffTraffic).el	1072	0	2	0.001	0.703	0.007	7192.618
WSOA(EffTraffic).il	1072	0	1	0.031	1.541	0.208	6976.161
WSOA(Ambulance).el	111	0	1	1.169E−5	0.084	7.587E−4	7199.915
WSOA(Ambulance).il	111	0	1	0.003	0.898	0.207	7176.990
SkippedData	248			3327.302	6610.276		
DAEffTraffic	1072			3400.046	6676.265		
DAAmbulance	111			3725.710	6672.915		

temporal quality of the service given by MiSCi. The usability metric allows measuring the ease with which citizens use a service, and how useful it has been in order to achieve an objective. Thus, it measures the utility of the service given by MisCi.

$$\text{Timeliness:} \frac{\text{TR}_\text{E}}{\text{TR}}$$

Where TR_E: Response time of the agents using the emergent property of MisCI; TR: response time of the agents when they do not exploit the emergent property

If timeliness is less than one, it indicates that the emergence was relevant, at the right time, and it is convenient for the given situation.

$$\text{Usability:} \frac{\text{PL}_\text{E}}{\text{PL}}$$

Where PL_E: Number of people that arrived due to the emergent process; PL: Number of people that arrived without the emergent process.

If the usability is greater than one, then it indicates that the emergent process was useful, and it was suitable for the needs of the moment.

Table 3 Results of the third experiment

Node.Lst	#Ent	Lgth	Max	Avg	MaxSt	MeanSt	T.Free
WSA(VSS).el	14,119	1933	1933	978.218	995.699	499.732	0.015
WSA(VSS).il	12,186	1	1	1.0	5.173	0.590	0.0
WSA(HSS).el	1413	0	4	0.015	3.764	0.080	7104.679
WSA(HSS).il	1413	0	1	0.116	3.848	0.594	6360.322
InternetUpload.el	13,598	0	4	0.009	0.560	0.005	7138.138
InternetUpload.il	13,598	0	1	0.094	0.714	0.050	6518.196
VSSEndpoint.el	12,185	0	6	0.032	0.626	0.018	6998.694
VSSEndpoint.il	12,185	0	1	0.196	0.414	0.116	5784.422
HSSEndpoint.el	1413	23	32	10.611	160.861	54.233	658.418
HSSEndpoint.il	1390	1	1	0.968	7.606	5.016	225.777
InternetDownload.el	11,102	0	3	0.004	0.316	0.002	7170.470
InternetDownload.il	11,102	0	1	0.076	0.466	0.049	6648.729
WSOA(EffTraffic).el	9713	9	33	3.601	20.702	2.668	2406.517
WSOA(EffTraffic).il	9704	1	1	0.816	5.496	0.605	1321.813
WSOA(Ambulance).el	1389	0	1	1.021E−4	0.664	5.297E−4	7199.264
WSOA(Ambulance).il	1389	0	1	0.119	4.249	0.619	6339.623
SkippedData	2472			501.289	994.496		
DAEffTraffic	9703			503.082	1003.106		
DAAmbulance	1389			60.627	167.772		

5.2 Description of the emergent process

As was previously explained in Sect. 3.2, the emergent process seeks to unify the knowledge models of a smart city, and generate emerging ontologies that can be used by the different services that operate in the smart city. Next, it is explained how the ontological emergent process occurs in the first scenario presented in Sect. 4.

At the moment of the fault in the subway system, the goal of MiSCi is to quickly integrate information about the different types of transportation in the city, to offer the citizens all the possible options that allow them to get to their destination as soon as possible.

MiSCi generates a daemon application to activate the mobility system, which is one of the transport options suggested to citizens. However, as there is not a conceptual model that reflects all the types of transport and their characteristics, this application requests services from the OEL, for the generation of a temporary emergent ontology, with the types of transport available. To this end, a *strong merge* will be made between the registered ontologies of transport services and the meta-ontology of the transport domain, in order to generate the emerging ontology (see Fig. 5). A strong merge of ontologies includes all the aligned concepts of all the involved ontologies. As can see in Fig. 5, the Meta-Ontology has the meta-concepts MA, MB, and MC, then the *alignment* is made between the associated concepts of the recorded ontologies by the different transport services and the meta-concepts. Next, it is carried out a merge

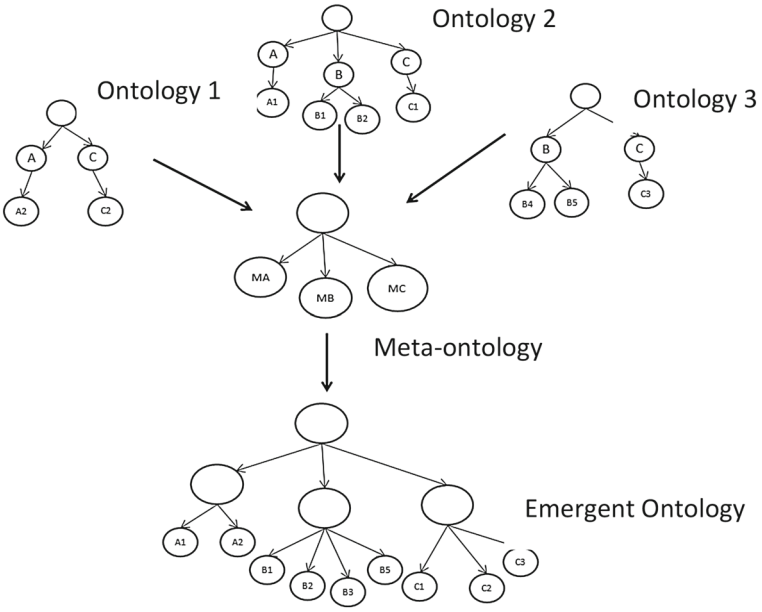

Fig. 5 Generation of the emergent ontology based on the merge of ontologies

that involves all the concepts and generates an emerging ontology that contains the meta-concepts and all the associated concepts. This emergent ontology will contain all the types of transport in the smart city. Then, the daemon application will be able to consult all the available services associated with the concepts (in this case, specific transports: Carpool, buses, bicycles, etc.), and process them (compare, filter, etc.), in order to make inferences about the need to send more transport units and more frequently, the possible mobility options according to their properties (location, capacity, availability, etc.), among other inferences. The agents that use the application will be able to reason using this information. For example, the vehicular system can decide to send a greater number of transport units to the site, sending notifications to the users about the vehicles in the zone, so that the citizens can make the most appropriate decision for their mobilization needs, in order to reach their destinations quickly in this unexpected situation.

5.3 Simulations

These experiments are aimed to check the emergent and ubiquitous properties of MiSCi and are based on the first scenario. We introduce some assumptions, to test only the emergent and ubiquitous properties of MiSCi. We compare the behavior of MiSCi without emergent processes vs the behavior of MiSCi with emergent processes, this will allow us assessing if these properties work properly. The assumptions are:

- Exist only three possible destinations for the passengers.

Table 4 Example of results from the simulations

People	Time	Timeliness	Usability
122	21.626	Base	Base
210	20.542	0.94987515	1.72131148
226	21.126	0.97687968	1.85245902
226	21.126	0.97687968	1.85245902
226	21.126	0.97687968	1.85245902

- There is not much traffic in the area, this allows proving only the emergent capacities of MiSCi.
- Buses and Carpools always have all seats available.
- The Carpools take 20% less time than the buses for reaching the destination.
- Each bus has a capacity of 30 people and the Carpools of 4 people.
- Each test case runs for 35 min.
- Without emergence, the buses leave every 30 min.
- Without emergence, the users are not notified about possible Carpools.
- With emergence, the Vehicular Smart System sends more transport units to destinations (1 every 10 min).
- With emergence, notifications are sent to users with vehicles near the incident area, and to users with transport needs, such that this will cause more Carpools in the area (2 times the number of Carpools, with respect to the case without emergence).
- MiSCi agents use the ontology to reason and make emerge more Carpools and transport units in the area.

The variables of the simulations are:

1. Number of people with mobility needs (it takes values of 500, 1000, 1500, 2000).
2. Percentage of people to each destination (it takes values of (a) A: 50%, B: 25%, C: 25% (b) A: 45%, B: 45%, C: 10%, (c) A: 80%, B: 10%, C: 10%, (d) A: 33%, B: 33%, C: 34%).
3. Number of bus units available to send to the area (it takes values of 3, 7, 10).
4. Number of Carpools in the area (it takes values of 5, 10, 15, 20).
5. Travel time to destinations A, B and C of buses (take values of (a) A: between 10 and 15 min, B: between 10 and 15 min, C: between 10 and 15 min, (b) A: between 10 and 15 min, B: between 20 and 30 min, C: between 45 and 60 min, (c) A: between 45 and 60 min, B: between 20 and 30 min, C: between 10 and 15 min, (d) A: between 20 and 30 min, B: between 20 and 30 min, C: between 10 and 15 min).

After combining all those values, we have 768 test cases. We have introduced those values in the simulation model (with and without emergence), and we have obtained the time of mobility, as well as the number of people translated in the 35 min of each simulation. Table 4 shows an example of the results of the simulations.

The first row in Table 4 represents an example of base case (without emergence), and the next rows show the results of the same base case with emergence, modifying some of the variables of the simulation. For example, the first row of the Table 4 is the base case for the values People = 1000, percentage of People to each destination

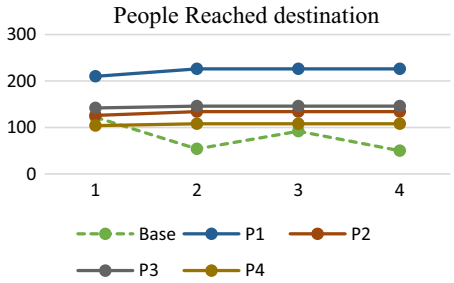

Fig. 6 People that reached destination. Cases of 1000 people, P1: Bus = 3, Carpools = 5, P2: Bus = 3, Carpools = 10, P3: bus = 3, Carpools = 15, P4: Bus = 3, Carpools = 20

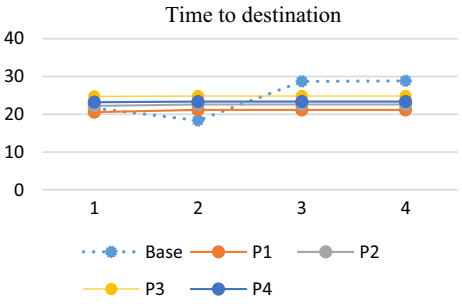

Fig. 7 Time to destination. Cases of 1000 people, P1: Bus = 3, Carpools = 5, P2: Bus = 3, Carpools = 10, P3: bus = 3, Carpool = 15, P4: Bus = 3, Carpools = 2

A: 33%, B: 33%, C: 34%., number of buses: 3, Carpools = 5, and Travel Time to destination A: between 10 and 15 min, B: between 20 and 30 min, C: between 45 and 60 min; the second row is the same case with emergence, the third row with emergence and Carpools = 10, and so forth.

In general, all the cases were not affected by the number of people (500, 1000, 1500, 2000), they give always the same result. Also, we can see that in all cases the Timeliness is minor than 1 and the Usability is major than 1, which shows that the emergent process generates knowledge on time and usable.

In Figs. 6 and 7, we compare several base cases with their emergent cases, where some variables of the simulations are modified (number of buses and Carpools). In Fig. 6, we can observe the behavior of people that reached a destination, and it is observed more people that reach the destination by using an emergent process. In Fig. 7, the emergent process was not useful only in one case, when the travel time to destinations A, B, and C of buses take values of A: between 10 and 15 min, B: between 10 and 15 min, C: between 10 and 15 min (case 2).

The values of the metrics of usability and timeliness determine the benefits of the emergence and ubiquity in MisCI. It can be observed in the simulations (Table 4 and Figs. 6 and 7) how timeliness takes values less than 1, which indicates that the use of the emerging ontologies, in order to reason and infer appropriate decisions about sending more transport units, improve the times with respect to the base case. Only in one case, this value is greater than one, and it is the case when the times of destination are very short so that the reasoning time is bigger than the time to reach the destination. This reasoning time includes the times of the different services, of the different MiSCi components to use the cloud, for the communications, etc. These times must be improved, in order to reduce the time of the emergent process. With

Table 5 Comparison with related works

Work	Ontolog	Interaction with smart city domains	Emergent behavior	MAPE	Web Services	MAS
[9]		X			X	X
[10]	X	X	X		X	
[32]	X			X	X	X
[15]		X			X	
MiSCi	X	X	X	X	X	X

respect to usability, it is greater than one always, which indicates that the emerging ontologies were useful and appropriate for the situations presented.

6 Comparison with previous works

This section presents a comparative analysis with previous studies that propose a solution in the area of contextual awareness, data processing, and service management, in a smart city.

In Table 5 it is presented a comparison of our proposal with previous works about middleware for smart cities. The criteria used for the comparison are: if they use ontologies to represent the knowledge about if the middleware can be adapted to a given domain, if it uses a MAPE loop for an autonomic behavior, if it is based on web services and MAS, and finally if it has an emergent behavior. All of them are important, in order to follow the dynamic of a smart city.

In [9] is proposed a middleware, which is divided into three layers. One of them is the cloud layer, which is key because manages data integration and provides inference services. In our proposal, MiSCi is composed of a multi-layer architecture based on 7 layers, and each layer is defined by a specific MAS. The agents can register, discover and consume web services in the cloud, and vice versa.

In [10], the authors propose an architecture for context-aware smart environments composed of four layers. In our case, MiSCi is deployed as a distributed middleware, this feature allows it to deal with scalability and performance troubles. Besides, it is coordinated with the core systems of the smart city, to find solutions to situations that arise in the city locally, based on its ubiquitous and emergent properties. In this way, the architecture of MiSCi considers the main sub-systems of a Smart City, which are responsible for managing the elements of the city in a global way, such as a Vehicular Smart System, Mobility Smart System, Smart Healthcare System, etc. The agents of MiSCi can communicate and coordinate with these systems.

[32] proposes a cloud context-aware service framework for the data storage and the processing needs of the smart city applications. In our proposal, in addition to a separate layer for the management of contextual awareness, it is extended with a layer to manage the ontological emergence in a smart city, allowing MiSCi to respond to emerging situations that may arise at a particular time. Our autonomic reflexive

architecture is based on MAS and web services, allowing its services to be consumed by applications (in our case, agents), aware of context or not. MiSCi is able to sense the environment, reason and execute a solution. Agents can create temporary or permanent emerging ontologies, which allow solving a particular situation, according to context.

In [15] is proposed Civitas, supports the deployment of services in a Smart City. Our architecture, additionally to manage the services in a smart city, has the capacity for reflection, self-adaptation, and emergence, since its design is based on the idea of an Autonomic Computing loop, known as MAPE.

The main advantages of our proposal are the ubiquitous and emergent properties of the architecture, through intelligent agents that can be adapted to the dynamism in a city, in a ubiquitous way, responding to the requirements of the citizens. For this, they use emerging ontologies, which allow not only adapted to the context of the moment and in real-time but also, respond to unforeseen situations.

This architecture allows the adaptation of the services of a smart city, according to its context, through the ontological contextual emergence, considering that the ontologies are a key component for the management of conceptual models of a smart city. In Alegre et al. [37] and Perera et al. [38] respectively, show the most common techniques used for context modeling. However, they expose that all techniques have disadvantages, but ontology-based modeling allows the support of semantic reasoning, expressiveness in context representation, strong validation, context sharing, and an application-independent approach. For these reasons ontologies are being taken as the standard in most applications. The ontologies allow producing and obtaining information with meaningful content and semantic characterization, allowing to carry out functions of reasoning and decision making. The use of ontologies in our framework allows integrating the service technology with the semantic web, adding the possibility of defining the semantics of the services, in order to give them a concrete and precise meaning. They also allow organizing the information and offer a metadata base for all the functionalities of a smart city, to describe its different components and facilitate the recovery of information. The use of ontologies also allows greater expressiveness in the representation of knowledge, standardization of the representation of information, specification of services and exchange of messages, among other things. The ontological emergence represents a dynamic way to create context ontologies, thus, it is a way to handle conceptual models, allowing update them with new concepts that emerge, and so have a much more accurate representation of the reality. In this way, it is able to generate ontologies dynamically through ontological mining processes, according to the needs of a smart city.

7 Conclusion

The MiSCi architecture has the capacity to respond to situations in real-time, through the use of emerging ontologies and contextual awareness, allowing providing responses adapted to the current situation. Similarly, the addition of agents as a base for making smart decisions allows MiSCi using the skills of reasoning, pro-activity, intelligence, etc., of this type of system, to improve the activities performed by users in the city. These three characteristics are combined in MiSCi, allowing that a smart

city can adapt and responds to the needs of its citizens in real-time, leading to an improvement in the development of their activities and their quality of life.

The intelligent agents can be adapted to the existing dynamism in a city, offering their services anywhere, anytime, to anyone, to ubiquitously responding in real-time to the requirements of the smart city, using emerging ontologies that describe unforeseen situations, according to the context of the moment.

By using ontological emergence, we have the possibility of creating temporary ontologies that are used to resolve a particular situation in the city, while permanent ontologies allow addressing repetitive situations, from which MiSCi may have learning processes to improve the responses of the smart city in similar situations.

Thus, one of the contributions of this paper is to show the utilization of the ontological emergence in conjunction with the context awareness and agents, in order to respond in an appropriate way to the unexpected situations in a smart city. Another important contribution of this work is the analysis of the performance of our middleware. In particular, we are interested in evaluating the behavior of the MiSCi agents in the context of the ontological emergence. The results, in different simulation contexts, are very promising, since the performance degradations in the system are due normally to other factors. Only when the requirements of real-time have very hard, it is necessary to improve the performance of the MiSCi components.

Thus, our main contributions have been shown in this paper: the architecture supports context-aware smart decisions in Smart Cities, the utilization of a meta-ontology to deal with emergent situations, and finally, the ubiquity and emergent properties of the architecture.

We are going to consider in future works, the analysis of the trust of the data collected in different sensors, and the incorporation of data analysis services. Additionally, we are going to design the main data analytical tasks required in a smart city, based on the idea of the autonomic cycle of data analysis tasks, to be included in MiSCi, in order to generate the knowledge required for a smart city [34, 35]. Also, as future work, the validation of our proposal in real cases is considered important.

One limitation regarding this research is its dependency on the Internet's latency. As it could be seen from the experiments section, a low internet connection could generate issues to get a response in real-time. There is an important work around this issue as future work, and we think that it can be fixed by integrating the fog-computing paradigm within our proposed architecture. Also, the computational cost of MiSCi must be evaluated, due to the different types of services that must be provided simultaneously, some of which are of context, ontological emergence, agent management (digital twins of the elements of a smart city), among others.

References

1. Mendonça M, Aguilar J (2016) Perozo N (2016) MiR-EO: Middleware Reflexivo para la Emergencia Ontológica en Ambientes Inteligentes. Latin Am J Comput 3(2):25–39
2. Aguilar J, Jerez M, Mendonca M, Sánchez M (2016) MiSCi: autonomic reflective middleware for smart cities. In: Valencia-García R, et al (eds) Technologies and Innovation (.), Communications computer and information science series, vol 658, Springer, Berlin, pp 241–253

3. Aguilar J (2016) Emergence and ubiquity in the smart cities. In: 6th IFIP world information technology forum, IFIP advances in information and communication technology series, vol 481. Springer, pp 235–244
4. Aguilar J, Ríos A, Hidrobo F, Cerrada M (2013) Sistemas Multiagentes y sus aplicaciones en Automatización Industrial, 2nd edn. Talleres Gráficos, Universidad de Los Andes, 2013. http://www.ing.ula.ve/~aguilar/publicaciones/objetos/author_libros/editor.pdf. Accessed Jan 2019
5. Tan Y, Zhang C, Mao Y, Qian G (2015) Semantic presentation and fusion framework of unstructured data in smart cites. In: 10th conference industrial electronics and applications (ICIEA)
6. Neirotti P, De Marco A, Cagliano AC, Mangano G, Scorrano F (2014) Current trends in smart city initiatives: some stylised facts. Cities 38:25–36
7. Komninos N, Bratsas C, Kakderi C, Tsarchopoulos P (2015) Smart city ontologies: improving the effectiveness of smart city applications. J Smart Cities 1(1):1–16
8. Pellicer S, Santa G, Bleda AL, Maestre R, Jara AJ, Skarmeta AG (2013) A global perspective of smart cities: a survey. In: Seventh international conference in innovative mobile and internet services in ubiquitous computing (IMIS)
9. Maciel C, De Souza PC, Viterbo J, Mendes F, Seghrouchni A (2015) A multi-agent architecture to support ubiquitous applications in smart environments. In: Agent technology for intelligent mobile services and smart societies, Springer, pp 106–116
10. Degeler V, Lazovik A (2013) Architecture pattern for context-aware smart environments. In: Creating personal, social and urban awareness through pervasive computing, pp 108–130
11. Giffinger R, Gudrun H (2010) Smart cities ranking: an effective instrument for the positioning of the cities? ACE: Archit City Environ 4(12):7–26
12. Roscia M, Longo M, Lazaroiu GC (2013) Smart city by multi-agent systems. In: International conference on renewable energy research and applications (ICRERA), pp 371–376
13. Santana EFZ, Chaves AP, Gerosa MA, Kon F, Milojicic D (2017) Software platforms for smart cities: concepts, requirements, challenges, and a unified reference architecture. ACM Comput Surv 50(6):1–37
14. Khan Z, Kiani SL, Soomro K (2014) A framework for cloud-based context-aware information services for citizens in smart cities. Journal of Cloud Computing 3(1):1
15. Villanueva F, Santofimia M, Villa D, Barba J, López J (2013) Civitas: the smart city middleware, from sensors to big data. In: Seventh international conference on innovative mobile and internet services in ubiquitous computing, pp 445–450
16. Romualdo L, Finkelstein A, Gann D (2013) A middleware framework for urban data management. UbiComp'13, pp 1359–1362
17. Handte M; Foell S, Schiele G, Iqbal U, Apolinarski W, Parreira X, Marrón P, Kortuem G (2014) The GAMBAS middleware for smart city applications. In: Internet of things success stories (Cousin, Philippe ed.). Internet of things European research cluster (IERC), pp 74–81
18. Hernández-Muñoz J, Vercher J, Muñoz L, Galache J, Presser M, Hernández-Gómez L, Pettersson J (2011) Smart cities at the forefront of the future Internet. LNCS 6656:447–462
19. Portmann E, Finger M (2016) Towards cognitive cities: advances in cognitive computing and its application to the governance of large urban systems, vol 63. Springer, Berlin
20. Fujiwara Y, Yamada K, Tabata K, Oda M, Hashimoto K, Suganuma T, Georgakopoulos A (2015) Context aware services: a novel trend in IoT based research in smart city project. In: 39th IEEE annual international computers, software & applications conference, pp 479–480
21. An EU/Japan collaboration to deliver Intelligent Knowledge-as-a-Service. http://iKaaS.com. Accessed 28 June 2016
22. Fan Z, Song X, Shibasaki R, Li T, Kaneda H (2016) CityCoupling: bridging intercity human mobility. In: Proceedings of the 2016 ACM international joint conference on pervasive and ubiquitous computing, pp 718–728
23. Xu M, Wang D, Li J (2016) DESTPRE: a data-driven approach to destination prediction for taxi rides. In: Proceedings ACM international joint conference on pervasive and ubiquitous computing, pp 729–739
24. Zambonelli F, Omicini A, Anzengruber B, Castelli G, De Angelis F, Serugendo G, Mariani S (2015) Developing pervasive multi-agent systems with nature-inspired coordination. Pervasive Mobile Comput 17:236–252
25. Ghannem A, Hamdi MS, Abdelmoez W, Ammar H (2015) A context model development process for smart city operations. In: IEEE international conference on service operations and logistics, and informatics (SOLI)

26. Al-Shargabi AA, Siewe F, Zahary A (2017) Quality of context in context-aware systems. EAI Endorsed Trans Context Aware Syst Appl 4(12):1–25
27. Vizcarrondo J, Aguilar J, Exposito E, Subias A (2012) ARMISCOM: Autonomic reflective middleware for management service composition. In: Global information infrastructure and networking symposium (GIIS), pp 1–8
28. Vizcarrondo J, Aguilar J, Exposito E, Subias A (2017) MAPE-K as a service-oriented architecture. IEEE Latin Am Trans 15(6):1163–1175
29. Aguilar J, Jerez M, Mendonca M, Sánchez M (2017) An extension of the MiSCi Middleware for smart cities based on fog computing. J Inf Technol Res (JITR) 10(4):23–41
30. Sánchez M, Aguilar J, Cordero J, Valdiviezo P, Luis B-G, Chamba-Eras L (2016) Cloud computing in smart educational environments: application in learning analytics as service. In: New advances in information systems and technologies, vol 444. Springer, pp 993–1002
31. Sánchez M, Aguilar J, Cordero J, Valdiviezo P (2015) A smart learning environment based on cloud learning. Int J Adv Inf Sci Technol (IJAIST) 39(39):39–52
32. Aguilar J, Jerez M, Exposito E, Villemur T (2015) CARMiCLOC: context awareness middleware in cloud computing. In: Latin American computing conference (CLEI), pp 532–541
33. Mendonça M, Aguilar J, Perozo N (2014) Middleware reflexivo semántico para ambientes inteligentes. In: Segunda Conferencia Nacional de Computación, Informática y Sistemas (CoNCISa), pp 24–32. http://www.concisa.net.ve/2014/ponencias/
34. Aguilar J, Sanchez M, Cordero J, Valdiviezo-Díaz P, Barba-Guamán L, Chamba-Eras L (2018) Learning Analytics Tasks as Services in Smart Classroom'. Univ Access Inf Soc J 17(4):693–709
35. Aguilar J (2019) A multi-HVAC system autonomic management architecture for smart buildings. In: Garces-Jimenez A, Gallego-Salvador N, Gutiérrez de Mesa J, Gómez-Pulido J, García-Tejedor A (eds) Coautores, IEEE Access, vol 7, pp 123402–123415
36. GLIDER: http://www.faces.ula.ve/glider/
37. Alegre U, Augusto J, Clark T (2016) Engineering context-aware systems and applications: a survey. J Syst Softw 117:55–83
38. Perera C, Zaslavsky A, Christen P, Georgakopoulos D (2014) Context-aware computing for the internet of things: a survey. Commun Surv Tutorials 16(1):414–454