An approach for semantic interoperability in autonomic distributed intelligent systems

Kamaleddin Yaghoobirafi | Ali Farahani

Abstract
The complexity and dynamism of distributed intelligent systems have motivated the utilization of autonomic computing in such systems. However, the interoperability of agents in order to attain self-managing properties is itself a great challenge that requires more attention. Although a number of semantic approaches exist for central autonomic systems, there are not many attempts to develop mechanisms for facilitating semantic interoperability to attain a distributed feedback loop. In this paper, a semantic approach is presented to overcome this challenge. The main component of this approach is AutonoML language that provides the shared semantics for autonomic computing in distributed environments. Besides this language, the method of its application for enhancing the interoperability of the distributed MAPE-K loop is presented. In order to evaluate this mechanism, three well-known metrics from the ontological assessment area are utilized, and the results are compared with an existing autonomic ontology that shows greater relational and attributes richness of the proposed semantic structures. Moreover, in order to evaluate the applicability of the mechanism, a case from NASA-ANTS project is studied, and the results show that the proposed mechanism is capable of facilitating the achievement of more stable self-adaptive solutions in less time and by exchanging fewer messages.

KEYWORDS
autonomic computing, AutonoML, distributed intelligent systems, MAPE-K, semantic, semantic interoperability

1 INTRODUCTION

The complexity of the information technology solutions has made their maintenance challenging in many situations. This has inspired the introduction of autonomic computing paradigm in the early years of the current century. This paradigm enables the system to process some environmental changes and react to them without human intervention. These capabilities that mainly include self-healing, self-protecting, self-optimizing, and self-configuring are generally named self-* properties. However, the growth of information technology solutions beyond the boundaries of one authority has motivated the emergence of distributed systems, which are not controlled by any central entity. In such systems, the occurrence of changes and incidents is considered a normal daily fact, and therefore, it is hardly possible to manage and maintain such systems by human experts. For this reason, applying autonomic computing and self-adaptation paradigms is mandatory in these environments. Therefore, a number

Kamaleddin Yaghoobirafi and Ali Farahani contributed equally to this study.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2022 The Authors. Journal of Software: Evolution and Process published by John Wiley & Sons Ltd.
of research studies have investigated the problem of applying these paradigms in distributed environments. These studies have resulted in some valuable reference models and various algorithms and approaches.\textsuperscript{4,5} However, the challenge of interoperability between agents participating in a distributed system in order to attain a self-* property is not considered deeply by researchers in this area. On the other hand, the need for semantic representation of the self-adaptive concepts, especially MAPE-K loop, is investigated in a number of research studies.\textsuperscript{6,7} These approaches that have generally led to an ontology or a semantic approach based on ontology have been proposed with central self-adaptive loop considerations. However, the main challenge of interoperability is related to distributed self-adaptive systems that need to organize agents in order to attain a self-* property. Moreover, in the context of distributed self-adaptive systems, the prospect system should be capable of exchanging information with semantic metadata between various agents and entities. In other words, it is not enough to provide a common semantic understanding of concepts in a distributed system, but also it is necessary to enable the participating agents and entities to exchange information items based on common semantic vocabulary.

Therefore, the pivotal objective of this paper is to present an approach that enables the definition of self-adaptive specifications of a distributed system in various schemas and enables the exchanging of information instances based on these schemas. All the taxonomies defined by this mechanism are distributed, and there is no single schema that can be considered as the main reference for semantic deductions. Moreover, this mechanism supports semantic interoperability in local organizations by defining specific taxonomies. At last, the ontologies that represent the common concepts of distributed systems, autonomic computing, etc. may be utilized as the reference of schemas and corresponding instance documents.

The major contribution of this paper is a language for exchanging the data between agents in order to achieve a self-* property. This language is not a certain taxonomy, ontology, or a common vocabulary. Instead, it defines a specification for developing, maintaining, and utilizing such contents in order to tackle interoperability barriers of decentralized autonomic mechanisms. As it can be observed in Figure 1, the contributions of this research can be categorized into semantic structures and the mechanism for enabling semantic interoperability in distributed autonomic environments using such structures. Moreover, the upper level metamodel for creating a common understanding and enabling semantic interoperability can be considered as the core component of the approach. Although the main component of this upper level metadata is a base taxonomy that should be extensible in order to cover various requirements, the mechanism recommends the utilization of other semantic metadata provided in form of ontologies. Consequently, some base ontologies are presented as other subcontribution of this paper. These contributions are evaluated with aid of semantic criteria that are used for a while in the domain of semantic and information science. At last, the other important contribution of this paper is the method and process of utilizing the proposed language in decentralized environments in order to enhance semantic interoperability in these environments. Due to the practical nature of this contribution, it is examined in a case study based on a classic mission defined based on NASA projects, called NASA-ANTS.

In the remainder of this paper, first, the related research work is reviewed, and then the proposed mechanism and its constituent components would be presented in Section 3. Section 4 is dedicated to the evaluation of the proposed mechanism. The case studies and related scenarios, in addition to implementation details and results, are presented in this section. The last section provides the conclusion and future work.
2 | RELATED WORK

Autonomic computing is introduced to serve as an architectural solution for implementing context-aware systems where the scale or the complexity of the system makes it almost impossible or impractical to deal with all the changes beforehand.\cite{1,3} There were studies to adopt context-awareness in the system, and autonomic computing has been introduced to answer such need. A practical and yet well-defined approach to implement the goal of autonomic computing approach is to utilize a reference model called MAPE-K loop.\cite{6} MAPE-K is an architectural feedback loop that helps the system to monitor the environment for the needs of adaptation, analyze the situation, and plan accordingly to execute some action in order to respond to the changes in the environment or the needs of stakeholders. The whole approach is presented as an interplay between managed system as the whole system that is under adaptation and the managing element that is the one that takes decisions and applies those decisions for adaptation. The managed system is most of the time consists of an environment and the target system. In the monitor step, the data representing the managed element are monitored in order to signal situations that need an adaptation to be dealt with. In the presence of such incidents or situations, the analyze step tries to re-interpret the situation to extract possible adaptation choices for counteracting the current situation. These choices are then used in the plan step to get studied. The result will be a plan that is the most feasible and preferable one with respect to the current situation based on the constraints available in the system. The assembled plan is then decomposed into the actionable steps by the execute step and is executed later in the line.

However, having autonomic computing in intelligent distributed systems is not achievable through implementing only a single feedback loop in such systems, and this motivated many researchers to work on applying autonomic computing to this kind of system. These studies have led to many reference models and architectures that generally focus on communications and organizations of agents toward achieving a self-* property. Some earlier works in this area focus on self-organization paradigms.\cite{7,10} These research studies that are mainly inspired by swarm colonies focus on communications between agents and the emergence of global patterns from local interactions.\cite{11,13} However, other researchers have tried to combine the bottom-up paradigm of self-organization with top-down self-adaptive methods. In Weyns et al,\cite{5} a reference model is proposed for applying the self-adaptation paradigm in decentralized environments. This reference model suggests the use of coordination mechanisms to attain a distributed feedback loop.

Moreover, some research works have proposed more detailed solutions for distributed feedback loops.\cite{4,14,16} Furthermore, in order to satisfy the need for having appropriate self-adaptive organizations, MACCODO organization model and middleware are presented.\cite{17,18} In this model, the architecture of agent organizations, roles, and capabilities are considered in order to provide a dynamic architecture with the capability of self-adaptation. Later, the creation of self-adaptive organizations and the need for having autonomic specific roles motivated a new approach presented in Rafi et al.\cite{19}

A well-studied breakthrough for adaptation and management of distributed systems is using coordination mechanisms as a tool to dealing with challenges in distributed environments. Studies such as Rosa et al., Farahani et al., and Alti et al\cite{20-22} combined the usage of coordination mechanisms, models of adaptation, and characteristics of distributed environment management. Some other researcher put focuses on coordination mechanisms for specific solution in multiagent systems\cite{23-25} to harvest their potentials for performing adaption in such environment.

Another approach that has gained attention in implementing autonomic computing in distributed intelligent systems is dealing with the problem of distributed yet integrated control through the concept of semantic mutual understanding. From the early years of autonomic computing, the semantic challenges of the autonomic approach especially for MAPE-K loop were investigated. In an early attempt, the IBM researchers have tried to utilize KOAN ontology in order to design a semantic correlation engine for MAPE-K loop.\cite{26} The semantic issues motivated more researchers to investigate the problem. In Lasierra et al,\cite{7} a semantic approach is presented to manage information exchange in an autonomic system. Although the main focus of this paper is on home applications, a general-purpose ontology is proposed, which covers components of MAPE-K loop. This approach is implemented in smart home scenarios. In addition, some other comprehensive approaches have been proposed for resolving semantic challenges of autonomic solutions.\cite{6,27} These approaches are mainly applicable to central autonomic solutions and usually utilize ontologies for this purpose.

In the context of distributed systems, many research studies have focused on semantic issues and relations between multiagent systems and ontologies.\cite{28} For this purpose, a number of ontologies are proposed for various applications of multiagent systems.\cite{29,31} These domain-specific ontologies can be utilized in our approach for the recognition of the concepts and attributes related to multiagent systems. In addition, several solutions are proposed for the problem of resolving inconsistencies among multiple ontologies in multiagent systems.\cite{32} Although some research studies have investigated the semantics of context-aware multiagent systems, the autonomic computing cycles are not investigated deeply in the literature of such systems.

And finally, few studies consider adaptation, the structure of a distributed system, coordination mechanism for the organization, and semantic interoperability in one package. Studies such as Rosa et al. and Farahani et al\cite{20,21} have combined the approach of having autonomic yet distributed approaches for organizing multiagent problems through some coordination mechanisms. Also, in Famaey et al. and Bagheri et al\cite{33,34} the automation and coordination mechanism is studied with the help of coordination mechanisms. However, as mentioned in Table 1, none of the aforementioned studies captured the essence of integrating an adaptation reference model with the data integration approaches inside
Table 1: Review of related work on distributed autonomic approaches, categorizing based on their coverage of automation, distribution, coordination mechanisms, and semantic interoperability

| Paper                 | Autonomic | Distributed | Coordination mechanism | Semantic interoperability |
|-----------------------|-----------|-------------|------------------------|--------------------------|
| Bagheri et al.        | ✓         | ✓           | ✓                      | ✓                        |
| Rosa et al.           | ✓         | ✓           | ✓                      | ✓                        |
| Weißbach et al.       | ✓         |             | ✓                      | ✓                        |
| Lalande et al.        | ✓         |             |                         |                          |
| Capodieci et al.      | ✓         |             |                         |                          |
| Melekhova et al.      | ✓         |             |                         |                          |
| Zambonelli et al.     | ✓         |             |                         |                          |
| Lasierra et al.       | ✓         |             |                         | ✓                        |
| Famaey et al.         | ✓         |             | ✓                      | ✓                        |
| Baresi et al.         | ✓         |             | ✓                      | ✓                        |
| Farinelli et al.      | ✓         |             | ✓                      | ✓                        |
| Farahani et al.       | ✓         | ✓           | ✓                      | ✓                        |
| Terán et al.          | ✓         |             |                         |                          |
| Alti et al.           | ✓         | ✓           | ✓                      | ✓                        |

distributed systems. Engineering a solution for interoperability in distributed environments alongside adaptation in such environments is the missing point in these studies.

In this paper, following the initiative presented in Yaghoobirafi and Nazemi, an approach is introduced to enable semantic interoperability and interactions among participants of self-adaptive MAPE-K loops. The main goal of the mechanism presented here is the enabling of participants to exchange semantically understandable information.

3 | Proposed Mechanism

In this section, a semantic approach is introduced and utilized for enhancing semantic interoperability between agents and entities of a distributed autonomic system. Due to the scope of the problem, the proposed approach has different aspects to cover the requirements as it can be observed in Figure 2. These aspects can be categorized into semantic structures and the mechanism for enabling semantic interoperability in distributed autonomic environments using such structures. The semantic structure is divided into two parts including base metadata and extended metadata. Base metadata consists of the DisMAPEK upper level schema along with some upper level domain-specific ontology, which will be described shortly. The second part addresses the detailed metadata that is the outcome of extending the upper levels and would be used to create final working instances. Of course, some of these semantic structures are not dependant on any metadata. Some examples of these semantic structures include functional and autonomic organization statements, role contracts, and detailed MAPE-K loop. Instance documents themselves include some kind of metadata to describe the nature and context of their values. In addition to semantic metadata, another important aspects covered in the approach are related to mechanisms of utilizing the semantic structures to attain self-* properties in decentralized environments through interoperability. Most important mechanisms and processes like event detection, creation of autonomic organization, definition, and assignment of autonomic roles will be described in Section 3.2.

The main part of this approach is AutonoML that is illustrated in Figure 3. The goal of this language is to enable agents to collaborate in a decentralized loop to exchange information and extract the necessary information and knowledge for accomplishing the autonomic tasks. For this purpose, as it can be observed, the upper level metadata is utilized to determine, discover, and communicate the recognized and actual events. Moreover, various structures of this language support organization specification and related role assignment.

The main reason that AutonoML is called a language, not a taxonomy or ontology, is its nature and mission that is not defining the detailed concepts and classes, but providing the base classes and the specification and recommendations to provide the metadata enabling semantic interoperability. For this purpose, modeling the semantic relations in IS-A hierarchies (which is usually done in ontology modeling) is not a requirement for such structures. The reason is that the prospect structures are considered to be extended easily according to specific domains and also the changing environmental variables. Therefore, adding the new concepts and classes into the complicated semantic hierarchies may be very confusing and time-consuming and also may limit the modeling capabilities of the structure. As a result, The proposed mechanism utilizes different schemas and instance documents to enhance the semantic interoperability between various agent organizations in order to attain self-*
Therefore, the main component of the AutonoML is DISMAPEK schema that should be extended according to each domain and instantiated to generate the essential documents in the life cycle of an autonomic loop. The schema components in this mechanism are responsible for the standardization of concepts and instances that intend to relate the actual values to those standard concepts. As the ultimate goal of the approach is to facilitate the emergence of self-* properties in distributed systems, a great number of standard concepts provided in the schema are largely derived from autonomic computing basic concepts and also reference models that represent the components of a distributed solution for autonomic computing. However, this mechanism is extensible, and any environmental and domain-specific concept can be added in the appropriate schema. Moreover, it is possible to utilize a multiagent general and domain-specific ontology to provide information about agents and their missions. For instance, there are several ontologies developed for multiagent systems in domains like manufacturing urban freight transportation and so on. The extensible nature of AutonoML provides the opportunity to utilize the semantically rich ontology to provide comprehensive meaning for the exchanged information. In order to reach this goal, a base upper level ontology is proposed as one of the metadata included in AutonoML specifications. As it can be observed in Figure 4, this upper level ontology aims at categorizing various fields and principles of self-adaptation and self-organization. In this manner, this ontology can be extended alongside the main taxonomy to cover the semantic requirements of various autonomic and self-adaptive disciplines.
Moreover, according to Figure 1, the contributions are not only limited to the AutonoML and related components but also include the mechanisms and approaches of utilizing this language to overcome semantic interoperability barriers in decentralized autonomic loops. These mechanisms that include event detection, creating autonomic organization, defining autonomic roles, role assignment, and coordinated self-adaptation are all discussed in the direction of providing and enhancing semantic interoperability. These mechanisms and approaches will be discussed shortly in the current section. Despite all of this, it can be said that the core component of this model is DisMAPE-K that provides common semantics for the concepts of distributed MAPE-K and is the base schema for producing and exchanging appropriate reports in form of instance documents. Moreover, as the events impact organizations’ structure and due to the need for the establishment of organizations responsible for the adaptation process, there should be some documents representing the organizational specifications. Of course, these statements should be created with the support of an appropriate schema. Also, it should be noted that these documents are created and transmitted in response to some events. In the next section, the overall structure and main components of DisMAPE-K schema will be presented.

3.1 Structure of DisMAPE-K schema

The core component of AutonoML is the schema covering the information elements that are needed to be exchanged between various agent organizations in order to attain self-* properties. The elements provided in this schema would be populated by participants of the system in the self-adaptation process in form of instance documents. As it can be observed in Figure 5, the main components of this schema are the components of MAPE-K loop customized for decentralized environments. However, this schema may be extended to cover various applications of self-adaptation. In addition, it is possible to split the schema into multiple ones.

The base components of this schema would be driven from the main components of MAPE-K loop, namely, monitor, analyze, plan, and execute. These components comprise the main ontology elements presented in semantic approaches that are previously proposed for information exchange through the central MAPE-K loop. However, in the schema proposed here, the distributed loop is considered instead of the central one. This assumption imposes some considerations and additional concepts and attributes. The most important difference that should be considered in all steps of the cycle is partial realization and control of the environment. As the concepts provided for each step should be utilized and populated by various agents, each having a specific narrow perspective. The remainder of this section is dedicated to the introduction of the main components of DisMAPE-K schema.
Entity: All of the data provided as part of the self-adaptation process in a distributed system are described from the view of its reporter. Therefore, the authentication of that reporter entity is of great importance to the approach, and consequently, a complex element is included in DisMAPE-K to describe the specifications of that entity. This element should contain the identity of the reporter entity (which is usually an agent) and the organization that the entity belongs to. According to the model presented in Rafi et al., it is recommended that self-adaptive organizations that are responsible for self-adaptation processes be considered separately. As a result, entity type includes entity name, entity ID, organization ID, and self-adaptive organization ID as its elements. However, the self-adaptive organization is a temporal one that is not defined in all of the processes of self-adaptation. Moreover, in some approaches, there is no self-adaptive organization, and its responsibilities are taken by other entities. Therefore, in order to cover all of the autonomic approaches, the self-adaptive organization is considered as an optional element. The general definition of the entity element in DisMAPE-K schema is shown in the following code:

```
<element name="entity">
   <complexType>
      <sequence>
         <element name="entityID" minOccurs="1" maxOccurs="1"/>
         <element name="entityName" minOccurs="1" maxOccurs="1"/>
         <element name="organization" minOccurs="0"/>
         <element name="selfAdaptiveOrganization" minOccurs="0"/>
      </sequence>
   </complexType>
</element>
```

Monitor: Monitoring the distributed environments is a very challenging activity; as in such environments, every agent captures some data from the environment and perceives the events identically. In this way, recognition of the actual events is only possible by considering the data provided by many agents. Therefore, in the proposed schema, the monitor element has two main components: one for monitored data and the other one for specifying the event. Although these two elements seem identical to what previously was proposed for central MAPE-K, there is a fundamental difference and that is the partial nature of the data. Therefore, due to the lack of general control over the execution environment, there is no possibility of identifying the event completely by a single entity. Accordingly, the event should be perceived by each agent from its unique point of view. For this purpose, the monitoring element in AutonoML contains three parts: The first part specifies the date and time of the event occurrence. This information facilitates the analysis of events by considering the time and precedence of the events perceived by different agents. The next element is named “data,” which presents the data monitored by the entity. The last part of the monitor is “perceived event” that is a complex-type element. This element contains subelements clarifying the details of event occurrence including event location, affected entities, affected organizations, and confidence level that shows the amount of agent confidence in its perception. The overall definition of the monitor element is as follows:
Analyze: The main task in the analyze phase of a distributed self-adaptive system is to detect the main event from partial perceptions provided in the monitoring phase. It should be noted that the event recognized in such a way should be considered with the probability of some inaccuracies in event detection. Therefore, the probability of events should be considered as one of the important reported items of the recognized event. In addition, it is possible to recognize multiple events based on perceived events. In such situations, handling solutions should be considered for every important recognized event. In order to recognize important events, it is required to include an impact value, which shows the possible impact of the event if it is recognized correctly. Moreover, the recognized event includes elements like affected organizations, affected entities, and confidence levels as in the perceived event. After event recognition, solutions should be proposed to handle the recognized events with significant impact or probability. The solution statements should be clear enough in order to be planned and executed by the remaining components of the MAPE-K. Therefore, they should contain some subelements like solution description and solution process. The solution process itself comprises steps to execute the desired solution. These elements are summarized in the definition of analyzing element as it is shown in the following code:

Plan: After analyzing the events and proposing appropriate solutions, the required tasks should be planned for execution. The first element required for the identification of a good plan is the goal of the solution. There may be multiple goals for a plan, which translate to transparent tasks and roles. Therefore, the main purpose of the plan phase is to assign tasks to the appropriate roles in an organization. As a result, two
separate elements are considered to map the tasks to organizations and entities: organization contract and entity contract. Organization assignments relate the planned tasks to organizations and entity assignments do such mapping at the level of organization members. The following code shows the outline of such a planning element:

```xml
<element name="plan">
  <complexType>
    <sequence>
      <element name="organizationContract">
        <complexType>
          <sequence>
            <element name="organisationID">
              <element name="task"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <element name="entityContract">
        <complexType>
          <sequence>
            <element name="entityID">
              <element name="task"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </complexType>
  </element>
</element>
```

**Execution**: The main duty of the execution phase is to assign planned tasks to proper organizations and agents and to execute them. Therefore, the information necessary to be exchanged in the execution phase is mainly related to the log and reports of the operations executed and their results. Consequently, the main element here is the execution log that represents the list of performed operations and contains some subelements: the entity that represents the entity that has executed the plan, an activity that shows the tasks performed, date and time of the execution, and effect that represents the results of execution.

```xml
<element name="execute">
  <complexType>
    <sequence>
      <element name="ExecutionLog">
        <complexType>
          <sequence>
            <element name="entity"/>
            <element name="activity"/>
            <element name="dateTime"/>
            <element name="effect"/>
          </sequence>
        </complexType>
      </element>
    </sequence>
  </complexType>
</element>
```

### 3.2 Process of adaptation using AutonoML

In case of event occurrence, agents and organizations should work collaboratively with aid of information exchange to adapt to changes. AutonoML can be used to facilitate communications toward this goal. The process of utilizing AutonoML for this purpose is shown in Figure 6. As it can be observed, the process starts with monitoring some data by some agents. Each agent may perceive the event uniquely and therefore should prepare an instance document providing monitoring details and its unique perspective. Then, a self-adaptive organization would be created and roles, like analyze and plan, would be related to some entities. In the next step, the analyze parties would integrate the monitored data and perceived events and recognize the main event. Then, according to these events, appropriate solutions would be proposed by specifying their steps. This information then would be included in an instance document and would be sent to planning parties.

The planning parties would specify the tasks based on solution steps recognized in the analyze phase. Also, the goal of each plan should be defined based on corresponding solutions. The specified tasks later would be assigned to appropriate organizations and responsible entities in them. Then, planning parties specify all the plans in form of an instance document and send them to the execution parties, which are in fact the
organizations and entities responsible to carry out the planned activities. Finally, responsible entities execute the plan and report the execution logs and the results as an instance document.

The important fact in this process is the distributed nature of all steps. It means that in addition to interoperability considerations between various MAPE-K components, the interoperability of agents should be considered in order to accomplish each step. For instance, as there may be many agents contributing to the monitoring of an event, appropriate mechanisms should be in place to coordinate the local activities into an effective monitoring task. This may be the case for other steps of a self-adaptive process.

### 4 | Evaluation

This section is dedicated to the evaluation of the contributions described in the previous section. For this purpose, the evaluation approach should take different aspects of contribution into consideration. Such an approach is summarized in Figure 7. As it can be observed, evaluation of the proposed mechanism can be addressed from two different perspectives: The first one is the capability of this mechanism in enhancing the semantic interoperability in the distributed autonomic multiagent systems. Undeniably in this manner, other semantic interoperability solutions including any ontology-based, schema-matching solution, and other novel mechanisms should be considered and compared.

For the sake of comparing semantic structures, three well-known metrics are utilized in this paper:\[^{41}\]: Relationship richness that reflects the diversity of relations and placement of relations in the ontology as in Equation (1). As it is shown, this metric is defined as the ratio of the number of relationships, divided by the sum of the number of subclasses.

\[
RR = \frac{|P|}{|SC| + |P|}
\]  

(1)
The second metric is the attribute richness that considers the number of attributes defined for each class, and as it can be seen in Equation (2), it is formally calculated by dividing the number of attributes by the number of classes.\(^ {41}\)

\[
RR = \frac{|\text{att}|}{|C|}
\]  

(2)

Finally, inheritance richness that addresses the quality of the IS-A hierarchy in ontologies can be calculated using Equation (3). According to the corresponding researchers, the lower value of this metric indicates the more horizontal structure that is ideal for exchanging detailed data and metadata.\(^ {41}\)

\[
IR_\text{i} = \frac{\sum_{C_i,C_j \in C} |HF(C_i,C_j)|}{|C|}
\]  

(3)

Although these metrics are proposed for ontologies, they can be applied in any semantic structure that consists of equivalent components.

For the purpose of evaluation, it is tried to evaluate the proposed semantic structures based on these three metrics and compare them with the state of the art. In this direction, the work done in Lasierra et al.\(^7\) is selected, as it proposes an ontology-based on MAPE-K loop and can be compared with the proposed semantic structures. However, it should be noted that the proposed semantic structures in this paper are only modeled at an abstract level, as they are designed for the sake of extension due to the specifications of various domains. Therefore, the mentioned metrics and their measurement technique should change slightly in order to cover the evaluation requirements. First of all, the relationship richness should be evaluated considering the relations between various base categories. Moreover, due to the flat structure of the proposed taxonomy at the base level, the number of classes is considered as the value of \(|S|\), with the assumption that a single level of inheritance is used in extended taxonomy. With these assumptions, as it can be observed in Figure 4, four of five classes are related to the entity and also are related to each other based on the MAPE-K loop logic. In this manner, the total of the seven non-inheritance relations is detected, and therefore the populated relationship-richness formula is calculated as follows:

\[
\frac{7}{5+7} = \frac{7}{12}
\]  

(4)

In the case of the research in Lasierra et al.,\(^7\) only the base level of the proposed ontology is considered. With this consideration, there are six non-inheritance relationships indicated in the provided big picture. As there are five main classes named management profile, monitoring task, analysis task, planning task, and execution task, the populated relationship-richness would be as follows:

\[
\frac{6}{5+6} = \frac{6}{11}
\]  

(5)

However, from the attribute richness perspective, as the ovals in the figure indicate, a total of 12 attributes are considered for all of the five entities in the DISMAPEK schema. As a result, the calculated attribute richness value would be “12.” However, in Lasierra et al.\(^7\) the attributes of the main five classes excluding the ID attributes is totally 7. At last, from the inheritance richness perspective, as it was discussed in the previous section, the purpose of the DISMAPEK schema and other components of the AutonoML is to provide the base common understanding through the distributed agents who collaborate in an autonomic feedback loop. Therefore, the vertical inheritance richness is not considered for this contribution, and therefore, the value of this metric is “1.” However, in Lasierra et al.\(^7\) although the accurate value cannot be recognized from the provided figures, the investigation shows that the main classes and subclasses have more than average of two subclasses, and therefore this value is bigger than “2” for this approach.

The summary of these comparisons and evaluations is available in Table 2. The results indicate acceptable relational and attribute richness and fewer inheritance richness for the proposed semantic structures. According to the previous discussions, as the purpose of these structures is extending and instantiating the concepts by various agents, a horizontal structure is ideal for this solution according to the researchers who proposed the metrics.\(^ {41}\)

On the other hand, in order to evaluate the proposed semantic adaptation mechanism, which is the second part of the evaluation, some scenarios from a distributed autonomic system should be selected to examine the applicability of the process on the scenario. Moreover, necessary instance documents based on DisMAPE-K should be generated. For this purpose, we use examples and scenarios inspired by the new class of NASA space missions especially NASA ANTS.\(^ {42,43}\) In this project, thousands of tiny spacecraft weighing less than 1.5 kg will work cooperatively

| Relation richness | Attribute richness | Inheritance richness |
|-------------------|-------------------|---------------------|
| Lasierra et al.\(^7\) | \(\frac{6}{11}\) | \(\frac{7}{5}\) | \(\leq 2\) |
| AutonoML | \(\frac{7}{12}\) | \(\frac{12}{5}\) | 1 |
to explore the asteroid belt. Three kinds of spacecraft cooperate in order to achieve the mission targets: workers, rulers, and messengers. Workers that constitute 80% of all spacecraft carry the necessary instruments to gather data. As the instruments they carry differ from each other, every worker gathers a specific data type. Rulers define and update the rules about the operations and team formation. Messengers have the responsibility of communications between workers, rulers, and stations on the earth. All of these spacecraft have a solar sail to obtain power from the sun and also onboard computing and necessary AI capabilities. Autonomic computing is obviously essential for the success of this mission. The self-* properties in this project are reached at the individual, team, and swarm levels.\textsuperscript{42,43}

In this section, a case inspired by NASA ANTS is considered, which has been previously simulated in a discrete event simulator developed in self-* lab.\textsuperscript{19} In this scenario, worker agents are considered in teams that have a leader agent each. Also, several rulers are considered, which permanently collaborate with their assigned leaders. All messaging and communications are done through messenger agents. In our case, one of the rulers fails suddenly, and as a result, assigned leaders will not be able to collaborate with their rulers anymore. In the first method, each leader sends messages periodically to its assigned ruler. Every leader waits a defined time for receiving the reply message. If this does not happen, it will broadcast a message to the leaders in range with special content showing the loss of the assigned ruler. The receiver leader checks the availability of its assigned ruler and in case of ruler availability sends the information of the ruler to the requester. By receiving the first ruler feedback message, the leader perceives the sender as the nearest ruler and replaces the ruler’s information with the information of its current ruler. The main portion of monitor and analyze in this can be accomplished by the sender messenger, as it can be observed in Figure 8.

Although this may work fine only by some pinging and simple messaging activities, it is not always the case. For instance, we consider the case illustrated in Figure 9. In this scenario, there is a specific area in which agents do not work properly, and the ruler1 has entered that area. In this case, following the above adaptation process, a new ruler would be assigned to the leader. However, after a while, the previous ruler may leave the area. In this situation, the ruler has lost all of its connections. Moreover, it is possible for the new ruler to enter the affected area and another ruler (possibly the previous ruler) gets assigned to the leaders. This loop may iterate many times, imposing a lot of overhead, while the source of the problem is not detected. However, using a semantic mechanism like AUTONOML would facilitate collaborative monitoring in order to find the main source of incidents and to analyze the incidents and suggest appropriate solutions.

Using interoperation mechanisms like AutonoML enables agents to monitor the changes and discover the main source of the problem. For instance, in the case shown in Figure 10, messenger1 observes that after a while messenger3 that was not responding becomes available, and messenger2 is out of access. Also, the leader of the team that has lost one of its workers observes that two other workers become unavailable too. Utilizing AutonoML-based collaborations, messenger1, the affected leader, and other entities share their event perceptions and become able to perform more accurate analysis. For instance, the affected leader analyzes the location of lost workers and also the reported lost messenger and ruler and realizes that they are approximately in the same area. Therefore, the leader analyzes the fact and concludes that there may be a problem in that area. A similar process of analyzing may be done by messenger1. Each agent considers the opinion and the results of analyzing neighbors and propagates its own opinion to them. After concluding, the leader of each organization plans for the agents and propagates the plan to them.

In order to utilize the AutonoML for collaborations in this case, first, the functional organizations should be identified. For this purpose, three organizations are defined in the instance documents. Each of these organizations has some goals, underlying objectives, and consequently needs some roles in order to satisfy them. Of course, each organization has some members who take part in roles through contracts. Each contract should be specified by the needed role and members playing it. A summarized definition for an organization in NASA ANTS can be seen in the following code:
When an event occurs, a self-adaptive organization would be created with the employment of the monitoring agents. Of course, this organization evolves during the adaptation process by employing more agents in different roles like analyze, plan, and execution. Even more detailed roles may be defined as descendants of main MAPE-K roles. In addition to completing the definition of self-adaptive organization, AutonoML facilitates the interoperation between members of the self-adaptive organization in order to attain a self-* property. For instance, in the case of the scenarios explained previously, the messenger is the first agent who discovers the ruler's failure through implicit monitoring during sending messages from a leader to ruler1. This agent creates an AutonoML instance document and explains its monitoring details. This document begins with the description of the entity as below:

```
<org:organization>
  <org:organizationID>01</org:organizationID>
  <org:goal>
    <org:goalDesc>finding asteroids</org:goalDesc>
  </org:goal>
  <org:objective>
    <org:objDesc>traverse to defined locations</org:objDesc>
  </org:objective>
  <org:objective>
    <org:objDesc>Find asteroid</org:objDesc>
  </org:objective>
  ...
  <org:role>worker</org:role>
  <org:role>leader</org:role>
  ...
  <org:membrt>r1</org:membrt>
  <org:membrt>r11</org:membrt>
  <org:contract>
    <org:role>worker</org:role>
    <org:member>r1</org:member>
  </org:contract>
  <org:contract>
    <org:role>leader</org:role>
    <org:member>r11</org:member>
  </org:contract>
</org:organization>
```

When an event occurs, a self-adaptive organization would be created with the employment of the monitoring agents. Of course, this organization evolves during the adaptation process by employing more agents in different roles like analyze, plan, and execution. Even more detailed roles may be defined as descendants of main MAPE-K roles. In addition to completing the definition of self-adaptive organization, AutonoML facilitates the interoperation between members of the self-adaptive organization in order to attain a self-* property. For instance, in the case of the scenarios explained previously, the messenger is the first agent who discovers the ruler's failure through implicit monitoring during sending messages from a leader to ruler1. This agent creates an AutonoML instance document and explains its monitoring details. This document begins with the description of the entity as below:

```
<dismapek:entity>
  <dismapek:entityID>1</dismapek:entityID>
  <dismapek:entityName>messenger1</dismapek:entityName>
  <dismapek:organizationID>01</dismapek:organizationID>
  <dismapek:selfadaptiveOrganization>01</dismapek:selfadaptiveOrganization>
</dismapek:entity>
```

After describing the entity specifications, the details of the monitored event would be described. The following code illustrates such monitoring details provided by messenger1 for the event of a failed ruler. As it can be observed, one of the main elements is dedicated to the
details of the perceived event including description, location of the event, and entities that get affected by this event. In addition, a confidence level is provided showing that the agent is how much sure about its perception. Similar information would be provided by different entities participating in monitoring activities.

```
<monitor>
  <monitorDateTime> 2017/02/12 9:30 </monitorDateTime>
  <data> r1 not responds </data>
  <perceivedEvent>
    <eventDesc> r1 failed </eventDesc>
    <eventLocation> 9.80,71.56 </eventLocation>
    <affectedEntities> l1 </affectedEntities>
    <affectedOrganization> G1 </affectedOrganization>
    <confidenceLevel> 80 </confidenceLevel>
  </perceivedEvent>
</monitor>
```

After this step, the messenger sends the monitor instance document to the nearest ruler. The ruler analyzes the event and suggests the solution. As it can be observed in the following code, the solution contains steps to replace the ruler. After this step, the the results of analyze should be sent to the planning entity.

```
<analyze>
  <recognizedEvent>
    <eventDesc>
      There is an area in which agents cannot operate well
    </eventDesc>
    <impact> some agents are disabled </impact>
    <affectedEntities> m2,r1,w1 </affectedEntities>
    <affectedOrganization> 01,02 </affectedOrganization>
    <confidenceLevel> 80 </confidenceLevel>
  </recognizedEvent>
  <solution>
    <solutionDesc>
      Prohibit workers to enter the area
    </solutionDesc>
    <solutionProcess>
      <solutionStep>
        Send message to all agents in the area specifying the location
      </solutionStep>
      <solutionStep>
        update routing parameters for each agent
      </solutionStep>
    </solutionProcess>
  </solution>
</analyze>
```
In this case, the planning entity is identical to the analyzer entity. Therefore, the ruler who analyzed the event and proposed the solution performs the planning task and creates planning instance documents.

```xml
<plan>
    <organizationContract>
        <organizationID> SO1 </organizationID>
        <task> Prohibit entrance to the affected area </task>
    </organizationContract>
    <entityContract>
        <entityID> m1 </entityID>
        <task> Send message to all agents </task>
    </entityContract>
    <entityContract>
        <entityID> m2, m3 </entityID>
        <task> route message </task>
    </entityContract>
    <entityContract>
        <entityID> l1, w1, w2,... </entityID>
        <task> update routing parameters </task>
    </entityContract>
</plan>
```

In order to evaluate the applicability of the proposed mechanism, some criteria are needed to measure the conformance of the proposed approach to the requirements of successful self-adaptive systems. For this purpose, four well-known factors are considered: stability, accuracy, short settling time, and cost.\(^{44-46}\) The first two measures are more qualitative and harder to be measured by direct metrics. However, the two latter ones are more quantitative and may be measured by inspecting the time and number of exchanged messages. In the remainder of this section, each of these factors would be investigated separately for both solutions.

**Stability:** In order to measure stability, two metrics can be employed: the duration in which the system works regularly without failure or changes after completing an adaptation process; and the number of failures that occurred after completing adaptation. In our case, in the first solution, each agent monitors the events individually and manages a separate self-adaptive process. Therefore, after each adaptation, the next event occurs in a short period. In fact, after each local adaptation in Situation 1, the agents would fail due to entering the area affected by the storm. However, in Solution 2, the agents are capable of detecting the main event through interoperation and therefore would probably be stable until the next failure. Therefore, the average number of changes after adaptation in these two scenarios can be measured by formulas in Equations (6) and (7). As it can be observed, the number of changes until time \( t \) in Situation 1 is affected by the average number of failures (NF) and the average number of failures caused by the entrance to the storm (EF). However, due to the identification of the main event in Situation 2, the number of changes in Situation 2 is only affected by the number of regular failures.

\[
\text{ChN}2(t) = \frac{\sum \text{NF}(t)}{n} + \frac{\sum \text{EF}(\text{UnsT})}{n}, \tag{6}
\]

\[
\text{ChN}2(t) = \frac{\sum \text{NF}(t)}{n}. \tag{7}
\]

Considering the average failure rate as 1 in every 10 s and the entrance rate to the storm area 5 times after each failure, the chart of changes rate would be like Figure 11. As it can be seen, the failure rate of AutonoML approach is always lower in comparison with the basic one and moreover experiences fewer fluctuations.

**Time to adapt:** In order to calculate the time to adapt, the simulator time ticks spent from event occurrence to completion of adaptation should be considered. In the first situation, several local adaptations take place before monitoring and adaptation to the main event. However, many of the local adaptations occur simultaneously. Finally, the calculated time tick for this situation is 30. In the latter case, the time needed for local adaptation is omitted, but adapting to the main event is a little more time-consuming due to the communications. The calculated time, in this case, is 16. These findings implicate major time enhancement.

**Cost:** The main cost of adaptation in our scenario is related to message passing. Therefore, the number of messages passed for adaptation should be measured in each situation. Although, in the first case, many messages should be exchanged for the purpose of local adaptation, in the latter case, the exchanged AutonoML messages for adaptation imposes a significant cost. Although the number of messages is still lower in comparison with the first situation, the sizes of messages are usually much bigger than in the first case. However, in this case, due to the high distance between entities and the high technology of communication infrastructures, the impact of messages number is much higher than the size of messages. The evaluation results are summarized in Table 3.
The results indicate that the proposed mechanism can facilitate the implementation of a holistic feedback loop which in turn improves the performance of adaptation. The only constraint is the size of AutonoML messages that limits the applicability of such an approach for less developed devices like ones in the Internet of Things (IoT) environments. Overcoming this constraint needs further research and technologies that are not in the scope of this paper.

5 | CONCLUSION

In this paper, a semantic approach is introduced to facilitate interoperation in a distributed self-adaptive system. The intent of such interoperations is to obtain some self-* properties. The core component of such a mechanism is AutonoML language, which provides a common vocabulary for autonomic computing concepts. This semantic language consists of some main schema documents including the main AutonoML schema and specifications of organizational elements. This language is complemented by processes and procedures describing the way of its utilization for enhancing interoperability in self-adaptive and autonomic systems. After proposing the mechanism, it was evaluated in two parts: the first one, evaluating the proposed language and semantic structures, and the second part, evaluating the proposed adaptation mechanism. The first part is accomplished by utilising three ontological metrics named relation richness, attribute richness, and inheritance richness. The results that are extracted and compared with an existing autonomic ontology show the greater relationship and attribute richness and more horizontal structure from the inheritance perspective for the AutonoML. The horizontal structure is well-matched with the nature and purpose of the proposed language in providing interoperability in distributed autonomic environments. The second part is evaluated by considering some scenarios from a well-known NASA-ANTS project. The scenario represents a case in which the adaptation may not be successfully done by local self-adaptive solutions. However, the mechanism proposed in this paper can satisfy this requirement and improve the stability and time needed for adaptation. The results show that the stability of local self-adaptive solutions is much less than the holistic approach. This is due to the deficiency in finding the main source of changes and consequently doing adaptations that cannot make the system stable, as the undiscovered source of changes imposes new events on the system. However, the proposed mechanism facilitates the interoperation of entities in order to attain holistic self-adaptive solutions that are more stable. Moreover, although the time of each local adaptation is less than the holistic adaptation, the total time needed for discovering and overcoming the main event is more than a holistic approach due to the many local adaptations needed in this process. The number of messages passed between agents obeys the same rule. However, the size of messages is much more in the case of passing AutonoML messages. Although this is not a big issue in most cases due to the robustness of the storage and communication technologies, it is an important issue in some applications like IoT. Although the approach proposed in this paper is implemented in XML format, it is
independent of the representation language and therefore can be implemented by utilizing JSON and other formats. However, this decision may become crucial for cases in which the size of messages is a big concern. Therefore, there is a need for solutions that extend the proposed approach from implementation perspective, considering the base formats. Moreover, the proposed approach here does not cover the communication protocols between different agent exchanging documents based on the metadata. However, such a protocol is an important piece of puzzle to enable the approach to become applicable in practical solutions and as a result can be considered as a future work.

ACKNOWLEDGMENT
This work was supported, in part, by SFI grants 13/RC/2094 and 16/SP/3804. Open access funding provided by IReL. [Correction added on 18 May 2022, after first online publication: IReL funding statement has been added.]

DATA AVAILABILITY STATEMENT
The data, codes, and simulations that support the findings of this study are available on request from the authors. However, some pieces of the data are modeled and presented publicly in the paper.

ORCID
Ali Farahani https://orcid.org/0000-0002-0741-640X

REFERENCES
1. Sharma DP, Singh BK, Gure AT, Choudhury T. Autonomic computing: models, applications, and brokerage. Autonomic computing in cloud resource management in industry 4.0: Springer; 2021:59-90.
2. Dehraj P, Sharma A. A review on architecture and models for autonomic software systems. J Supercomputing. 2021;77(1):388-417.
3. Northrop L, Feiler P, Gabriel RP, et al. Ultra-large-scale systems: the software challenge of the future, Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst; 2006.
4. Weyns D, Schmerl B, Grassi V, et al. On patterns for decentralized control in self-adaptive systems. Software Engineering for Self-adaptive Systems II: Springer; 2013:76-107.
5. Weyns D, Malek S, Andersson J. On decentralized self-adaptation: lessons from the trenches and challenges for the future. In: Proceedings of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems; 2010:84-93.
6. Famaey J, Latré S, Strassner J, De Turck F. An ontology-driven semantic bus for autonomic communication elements. In: IEEE International Workshop on Modelling Autonomic Communications Environments Springer; 2010:37-50.
7. Lasierra N, Alesanco A, O'Sullivan D, Garcia J. An autonomic ontology-based approach to manage information in home-based scenarios: from theory to practice. Data Knowledge Eng. 2013;87:185-205.
8. Kephart JO, Chess DM. The vision of autonomic computing. Computer. 2003;36(1):41-50.
9. di Marco Serugendo G, Gleizes M-P, Karageorgos A. Self-organization in multi-agent systems. Knowl Eng Rev. 2005;20(2):165-189.
10. Heylighen F. The science of self-organization and adaptivity. Encyclopedia Life Support Syst. 2001;5(3):253-280.
11. Prokopenko M. Design versus self-organization. Advances in applied self-organizing systems: Springer; 2013:3-21.
12. Di Marco Serugendo G, Gleizes M-P, Karageorgos A. Self-organisation and emergence in MAS: an overview. Informatica (03505596). 2006;30(1):45-54.
13. De Wolf T, Holvoet T. Emergence versus self-organisation: different concepts but promising when combined. In: International Workshop on Engineering Self-Organising Applications Springer; 2004:1-15.
14. Farahani A, Cabri G, Nazemi E. Self-* properties in collective adaptive systems. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct; 2016:1309-1314.
15. Farahani A, Nazemi E, Cabri G, Capodieci N. Enabling autonomic computing support for the jade agent platform. Scalable Comput Pract Exper. 2017; 18(1):91-103.
16. Tesauro G, Chess DM, Walsh WE, et al. A multi-agent systems approach to autonomic computing. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1 Citeseer; 2004:464-471.
17. Farahani A, Heasevoets R, Helleboogh A. The MACODO organization model for context-driven dynamic agent organizations. ACM Trans Autonom Auton Syst (TAAS). 2010;5(4):1-29.
18. Weyns D, Heasevoets R, Helleboogh A, Holvoet T, Joosen W. MACODO: middleware architecture for context-driven dynamic agent organizations. ACM Trans Autonom Auton Syst (TAAS). 2009;5(1):1-25.
19. Rafi KY, Farahani A, Nazemi E. An organizational model for autonomic intelligent distributed systems. Global J Technol. 2012;1:624-630.
20. Rosa NS, Campos GMM, Cavalcanti DJM. Lightweight formalisation of adaptive middleware. J Syst Architect. 2019;97:54-64.
21. Farahani A, Nazemi E. OMCM-CAS: organisational model and coordination mechanism for self-adaptation and self-organisation in collective adaptive systems. Int J Adv Intell Paradigms. 2020;16(1):43-64.
22. Alti A, Lakehal A, Laborie S, Roose P. Autonomic semantic-based context-aware platform for mobile applications in pervasive environments. Future Int. 2016;8(4):48.
23. Melekhova O, Malenfant J, Mescheriakov R, Chueshev A. A decentralised solution for coordinating decisions in large-scale autonomic systems. In: Matec Web of Conferences, Vol. 161 EDP Sciences; 2018:3024.
24. Capodieci N, Hart E, Cabri G. Artificial Immunology for collective adaptive systems design and implementation. ACM Trans Autonom Auton Syst (TAAS). 2016;11(2):1-25.
25. Zambonelli F, Omicini A, Anzengruber B, et al. Developing pervasive multi-agent systems with nature-inspired coordination. Pervasive Mobile Comput. 2015;17:236-252.
26. Stojanovic L, Schneider J, Maedche A, et al. The role of ontologies in autonomic computing systems. IBM Syst J. 2004;43(3):598-616.
27. Breitman K, Perazolo M. Using formal ontology representation and alignment strategies to enhance resource allocation in multi vendor autonomic environments. In: Fourth IEEE International Workshop on Engineering of Autonomic and Autonomous Systems (EASE'07) IEEE; 2007:117-126.
28. Hadzic M, Chang E, Wongthongtham P, Dillon T. Ontology-based Multi-agent Systems: Springer; 2009.
29. Anand N, van Duin R, Tavasszy L. Ontology-based multi-agent system for urban freight transportation. Int J Urban Sci. 2014;18(2):133-153.
30. Obikó M, Manik V. Ontologies for multi-agent systems in manufacturing domain. In: Proceedings. 13th International Workshop on Database and Expert Systems Applications IEEE; 2002:597-602.
31. Guo Q, Zhang M. Semantic information integration and question answering based on pervasive agent ontology. Expert Syst Appl. 2009;36(6):10068-10077.
32. Chocron P, Schorlemmer M. Attuning ontology alignments to semantically heterogeneous multi-agent interactions. In: Proceedings of the Twenty-Second European Conference on Artificial Intelligence; 2016:871-879.
33. Famaey J, Latré S, Strasser J, De Turck F. A hierarchical approach to autonomic network management. In: 2010 IEEE/IFIP Network Operations and Management Symposium Workshops IEEE; 2010:225-232.
34. Bagheri M, Sirjani M, Khamespanah E, et al. Coordinated actor model of self-adaptive track-based traffic control systems. J Syst Softw. 2018;143:116-139.
35. Yaghoobirafi K, Nazemi E. A self-adaptive middleware for attaining semantic self-interoperation property. In: 2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS* W) IEEE; 2016:293-298.
36. Weißbach M, Chrszon P, Springer T, Schill A. Decentralized coordinated execution of adaptations in distributed self-adaptive software systems. In: 2017 IEEE 11th International Conference on Self-Adaptive and Self-Organizing Systems (SASO) IEEE; 2017:111-120.
37. Lalande P, Morand D, Chollet S. Autonomic mediation middleware for smart manufacturing. IEEE Int Comput. 2017;21(1):32-39.
38. Baresi L, Guine S, Saeedi P. Achieving self-adaptation through dynamic group management. Assurances for Self-adaptive Systems: Springer; 2013:214-239.
39. Farinelli A, Rogers A, Jennings NR. Agent-based decentralised coordination for sensor networks using the max-sum algorithm. Autonomous Agents Multi-Agent Syst. 2014;28(3):337-380.
40. Terán J, Aguilar J, Cerrada M. Integration in industrial automation based on multi-agent systems using cultural algorithms for optimizing the coordination mechanisms. Comput Indust. 2017;91:11-23.
41. Tartir S, Arpinar IB, Moore M, P. Sheth A, Aleman-Meza B. OntoQA: metric-based ontology quality analysis; 2005.
42. Truszkowski W, Hinchey M, Rash J, Rouff C. NASA’s swarm missions: the challenge of building autonomous software. IT Professional. 2004;6(5):47-52.
43. Truszkowski W, Hallock H, Rouff C, et al. Autonomous and Autonomic Systems: With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems: Springer Science & Business Media; 2009.
44. Farahani A, Nazemi E, Cabri G, Rafizadeh A. An evaluation method for self-adaptive systems. In: 2016 IEEE International Conference on Systems, Man, and CYBERNETICS (SMC) IEEE; 2016:2814-002820.
45. Villegas NM, Müller HA, Tamura G, Duchien L, Casallas R. A framework for evaluating quality-driven self-adaptive software systems. In: Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems; 2011:80-89.
46. Wolf TD, Holvoet T. Evaluation and comparison of decentralised autonomic computing solutions. Department of Computer Science, KU Leuven; 2006.

How to cite this article: Yaghoobirafi K, Farahani A. An approach for semantic interoperability in autonomic distributed intelligent systems. J Softw Eval Proc. 2022;34(10):e2436. doi:10.1002/smr.2436