

Article

A Context-Aware Middleware for Context Modeling and Reasoning: A Case-Study in Smart Cultural Spaces

Konstantinos Michalakis ^{1,*}, Yannis Christodoulou ¹, George Caridakis ¹, Yorghos Voutos ² 
and Phivos Mylonas ² 

¹ School of Social Science, Cultural Technology and Communication, University of the Aegean, University Hill, 81132 Mytilene, Greece; yannischris@aegean.gr (Y.C.); gcari@aegean.gr (G.C.)

² Humanistic and Social Informatics Lab, Department of Informatics, Ionian University, 49132 Corfu, Greece; c16vout@ionio.gr (Y.V.); fmylonas@ionio.gr (P.M.)

* Correspondence: kmichalak@aegean.gr; Tel.: +30-698-250-1914

Abstract: The proliferation of smart things and the subsequent emergence of the Internet of Things has motivated the deployment of intelligent spaces that provide automated services to users. Context-awareness refers to the ability of the system to be aware of the virtual and physical environment, allowing more efficient personalization. Context modeling and reasoning are two important aspects of context-aware computing, since they enable the representation of contextual data and inference of high-level, meaningful information. Context-awareness middleware systems integrate context modeling and reasoning, providing abstraction and supporting heterogeneous context streams. In this work, such a context-awareness middleware system is presented, which integrates a proposed context model based on the adaptation and combination of the most prominent context categorization schemata. A hybrid reasoning procedure, which combines multiple techniques, is also proposed and integrated. The proposed system was evaluated in a real-case-scenario cultural space, which supports preventive conservation. The evaluation showed that the proposed system efficiently addressed both conceptual aspects, through means of representation and reasoning, and implementation aspects, through means of performance.

Keywords: middleware; context-awareness; context modelling; context reasoning; Internet of Things; smart cultural spaces



Citation: Michalakis, K.; Christodoulou, Y.; Caridakis, G.; Voutos, Y.; Mylonas, P. A Context-Aware Middleware for Context Modeling and Reasoning: A Case-Study in Smart Cultural Spaces. *Appl. Sci.* **2021**, *11*, 5770. <https://doi.org/10.3390/app11135770>

Received: 30 May 2021
Accepted: 12 June 2021
Published: 22 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

1.1. Background

Nowadays, we are surrounded by smart things, interacting with them in our daily activities. In this ubiquitous environment, a variety of things communicate with each other and adapt their behavior in order to provide suitable services and reach common goals [1]. *Ubiquitous computing* involves the employment of usually low-cost devices that add intelligence to an environment and offer personalized services to the users [2]. The emergence of the *Internet of Things* (IoT) paradigm expands the ubiquitous characteristics to a wider area, exploiting the existing internet backbone to provide seamless connectivity anywhere and by anything, transcending a single environment to cover all ecosystems and applications [3].

Context is defined as “any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves” [4]. *Context-awareness* (CA) refers to the ability of the system to be continually aware of its situation in the physical, virtual and user environment, and has been established as a major tool of ubiquitous computing [5]. Reviewing the field, Perera et al. [6] propose the following *context life cycle*, which has since then been widely adopted: context (a) acquisition, (b) modeling, (c) reasoning and (d) dissemination. A CA application may enhance the interaction between humans and machines by adding perception of the environment, which

eventually leads to intelligence. Machine-to-machine interaction can also be improved upon fusion, analysis and reasoning on the context. Indeed, many ubiquitous applications incorporate CA computing procedures [6,7].

Context modeling has always been at the center of CA computing. Early approaches included key-value and markup schemes (e.g., XML), which are computationally efficient and easy to manage but support limited reasoning and fail to represent more complex relations among contextual entities. In order to achieve efficient representation of relations, graphical models were introduced, such as the context modeling language (CML) [8]. Object-based models are also exploited to integrate inheritance and encapsulation, influenced by the success of object-based programming. In past years, logic formalism, in the form of formal ontologies, was used as the preferred method for developing context models [9]. The emerging concept of the semantic web of things [10] combines human, machine, physical and abstract things and environments toward a convergence of IoT with the semantic web. Hybrid approaches were introduced as early as 2004, when [11] combined formal ontologies with CML. Reviewing the field, Perera et al. [6] claim that incorporating multiple modeling techniques is the best way to mitigate each other's weaknesses.

Many *reasoning* techniques for making inferences about context are proposed in the literature [12], including probabilistic logic, rule-based, fuzzy logic, supervised and unsupervised learning and ontology-based semantic reasoning. The selection of the appropriate reasoning technique is heavily influenced by the adopted modeling method. Since formal ontologies have taken the world by the storm, it would be expected that ontology-based reasoning also follows the same popularity. Additionally, rule-based reasoning is easy to implement and has a broad applicability to all domains, making it the most commonly used technique [6].

The various reasoning techniques commonly suffer from certain drawbacks, mainly that each one is not versatile enough to individually address all the requirements of an IoT ecosystem [13]. Rules are error-prone due to manual labor and have complex maintenance. Supervised learning needs trained data, while unsupervised learning is not easily validated. Ontologies fail at uncertain values and require computational power. Probabilistic logic works only with numerical values, while fuzzy logic is prone to error and offers no validation [14]. On the other hand, each technique shows good results in tackling specific problems, e.g., supervised learning for activity recognition [15]. Thus, hybrid solutions have emerged to formulate complementary techniques in order to address each other's weaknesses [16].

The IoT paradigm envisions a wide range of deployed sensors, generating large volumes of data, which CA applications need to collect and model. The abundance of sensors and smart devices also introduced issues concerning the heterogeneity of sensor protocols, which traditional applications cannot tackle efficiently. Thus, *middleware* systems were proposed and integrated in IoT ecosystems in order to address diverse data sources and large data streams [6]. In the IoT context, a middleware system executes all stages of the context life cycle (acquiring, modeling, reasoning and dissemination). It abstracts communication details of smart devices and allows the connectivity of heterogeneous devices, while providing interoperability and diversity for applications and services [17].

Cultural heritage (CH) constitutes a worldwide resource, which defines artistically and economically many cities and regions [18,19]. Nowadays, cultural institutions seek new ways to attract their visitors, e.g., by providing digital guides, installed screens and other services, offering a new type of interactive and personalized user experience [20,21]. Furthermore, the integration of ubiquitous computing in cultural spaces paves the way for smart CH institutions, where visitors, equipped with their mobile devices, and artifacts, equipped with appropriate devices, interact with each other. New, personalized services are available, while the delivered content and recommendations are tailored to user characteristics and preferences, and the environment is adapted accordingly [22,23].

The latest advances in technologies that support the IoT paradigm, such as the emergence of low-cost sensors, have motivated academic research toward IoT solutions for

cultural spaces [24]. A typical research project includes the localization of visitors and the delivery of content based on user proximity with points of interest [25], often integrating a recommendation system [26]. Such solutions universally exploit CA techniques, ranging from location-based processing to social context and environmental parameters [22]. Context in a cultural space is multidimensional and may include the locations and profiles of users and artifacts, environmental data, behavioral patterns of users or other external data [27]. Finally, CA can also enhance conservation procedures, such as automatic monitoring of a cultural space for detecting critical conditions [28].

1.2. Contribution

Prior to modeling, the context needs to be categorized. A survey of categorization schemata proposed in the literature can be found in [12]. While each schema addresses context from a different aspect, they do not capture context from all aspects, i.e., conceptual (context representation) and implementation (context sources and processing). On this premise, this work proposes an adjustment and combination of three existing schemata that capture context from both aspects. A resulting combined schema, such as the one proposed in this work, is necessary in order to provide the basis for a concrete context model that is both able to represent the context as accurately as possible, and also incorporates requirements and directives for its realization into an IoT middleware system.

Furthermore, based on the proposed categorization schema, we propose a context model, which may be applied to define context in many scenarios. The proposed model includes five core classes that represent the basic features of a context observation, namely *thing*, *location*, *time*, *activity* and *reason*. Typically, in similar works, the reason feature is represented only in an implicit fashion (e.g., as a service), while the proposed approach enhances this feature, by introducing an exclusive core class in order to explicitly represent it. Additionally, a hybrid reasoning technique is proposed, which applies specific reasoning techniques based on their suitability to address specific sub-problems. Furthermore, the proposed reasoning procedure is divided into low-level processing, which is executed by the middleware, and high-level processing, which is executed by the front-end application.

The proposed context modeling and reasoning techniques are employed into a middleware, which is deployed in a cultural space in order to be assessed in terms of functionality and performance. The case study explores the benefits of context modeling and hybrid reasoning in a scenario of preventive conservation. The experimental results indicate that smart cultural spaces can benefit from the proposed CA middleware.

1.3. Paper Overview

The paper proceeds as follows: Section 2 presents an overview of related work with regard to the context categorization schemata, context models, context reasoning systems and CA middleware. Section 3 describes the proposed context model in detail, including the graphs of all core classes and their relations to subclasses. Section 4 presents the hybrid reasoning architecture and the underlying layers. The case study of preventive conservation is described in Section 5, along with the evaluation and results. Conclusions and closing remarks are discussed in Section 6.

2. Related Work

2.1. Context Categorization Schemata

The context can be identified and categorized in many different ways. One of the first efforts to categorize context uses three common questions to determine context: where you are, who you are with and what is nearby [29]. Abowd et al. [5] categorize context into primary and secondary, which is still one of the most used schemata. In their seminal work on ubiquitous computing, Abowd and Mynatt [30] indicate five questions as the minimum set that corresponds to every contextual observation, the 5 Ws, namely, who, what, when, where, and why. Benerecetti et al. [31] categorize context into physical and cultural, while Hofer et al. [32] categorize it into physical and logical. Henricksen [33]

distinguishes context according to source and mutability in (i) sensed context, i.e., data directly collected from sensors, (ii) static context, i.e., immutable data, (iii) profiled context, i.e., data that change infrequently and (iv) derived context, i.e., computed information. Direct and indirect context types are used by [34]. Mei and Easterbrook [35] propose a distinction between objective and cognitive context. Rizou et al. [36] categorize context as either observable or non-observable. Temporal categorization (active and past) is proposed by [37]. Physical and virtual (derived) context is proposed in [38]. Finally, contextual data may be event-driven or time-driven [12], depending on the triggering condition. Perera et al. [6] highlight that each schema has its strengths and weaknesses, thus an ideal middleware solution should employ a combination of different schemata. The authors also argue that most categorization schemata fall exclusively under the conceptual or operational type, each of which alone is not sufficient to represent context from all perspectives. On this premise, the current work goes beyond the state of the art by proposing a combination of existing categorization schemata that satisfies both the conceptual and implementation perspective.

2.2. Context Models

Context modeling has received much attention from researchers, maintaining an essential role in the context life cycle. Many surveys of context-modeling methods can be found in the literature [6,9,12,39]. The current overview does not focus on the different modeling methods, for which the reader may refer to the aforementioned citations, but rather focuses on the details of context representation.

Capeus [40] uses a key-value technique to map context data. Actors, devices and abstract devices are entities that expose events, which have a positive or negative distinction. Henriksen and Indulska [11] propose a model that includes the entities *person*, *device*, *activity* and *place* as core classes, while their method of choice is a graphical language called CML. Each entity has relations to other entities with a certainty parameter attributed. CONON [41] is one of the first attempts to represent context, using formal ontologies. *Computational entity*, *person*, *location* and *activity* are subclasses of the main class of *context entity*. Upper-level as well as domain-specific ontologies are utilized to specialize from generic to actual ecosystems. ContextUML [42] models context, using atomic and complex context classes to represent low-level and high-level data. Context acquisition is not modeled but abstract context resources are utilized. *Object*, *service* and *mechanism* are core classes of the ContextUML model.

CARE [43] categorizes context providers as users, network operators and service providers. Each entity falls under one of these categories and utilizes a profile manager to declare policies and derive high-level context data. SeCoMan [44] is a CA framework suitable to represent privacy and security. As such, the model is centered around the location and privileges related to persons. It follows a two-layer architecture, with the first layer capturing a representation of generic ecosystems tagged with privacy conditions and the second layer capturing the specifics of the applied ecosystem (a supermarket). Mcheick [45] proposes a model where context is the main class, while location, user profile, environmental parameter and other contextual data are subclasses. Furthermore, a meta-model is defined, which relates context to rules and contextual elements.

ECOPPA [46] is specialized in physical activity applications, thus activity assumes an important role in the model. *Computational entity*, *person*, *location* and *environmental entity* are associated with *activity*. The distinction between the upper level and domain level, found in other models, is also present in ECOPPA. Persuasion service is an interesting addition, which represents the recommendation service of the CA application. CAMEOnto [47] proposes a meta-ontology, which includes six main classes: *user*, *device*, *activity*, *time*, *location* and *service*. Each class is further analyzed into subclasses, and relations between the main classes are specified. An interesting design choice is that time is represented as a class rather than as a class property. Zhong-Jun et al. [48] propose a meta-context model, which efficiently models uncertainty. *User*, *actuator*, *environment* and *service* are all subclasses of

thing, which is the core class of the model. Furthermore, context is categorized as internal, external or boundary.

MSSN-Onto [49] proposes an ontology that models multimedia sensory data, providing data interoperability in identifying events applied in various domains. The ontology captures different aspects of context, such as the observation aspect that includes *sensor*, *system*, *procedure* and *observation* classes. The modeling of sensor and data-specific contexts includes *scalar sensor*, *media sensor*, *multimedia data* and *media segment* classes. HSSN ontology [50] extends the semantic sensor network in order to apply data, sensor and platform diversity. *Mobile sensor* and *static sensor* are added as child concepts to *sensor*.

Most of the approaches presented above utilize only a subset of the features device, user, activity, location, and service, as their core classes. In addition, only a few of them divide their models into generic and domain-specific layers. Finally, to our knowledge, no published context model includes a class that represents the *reason* feature of a contextual observation (as opposed to the proposed approach).

2.3. Context Reasoning Techniques

Extensive surveys of reasoning techniques can be found in the literature [6,12,39]. Wongpatikaseree et al. [51] propose an activity recognition reasoner for smart home applications. Similar rule-based reasoning, using formal ontologies for activities and smart home scenarios, is presented by [52,53]. Semantic reasoning in realistic IoT applications is explored by [54]. Probabilistic techniques are exploited in [55] for the recognition of daily activities. The Dempster Schafer theory of evidence is exploited in [56] and the Hidden Markov model is utilized in [57], again, for activity recognition.

Typical rule-based approaches include multiple if-then rules, which are often supported by formal ontologies [58] or key-value or other modeling methods [59]. Machine Learning (ML) is also integrated in reasoning systems with the most popular techniques being neural networks [60] and Bayesian networks [15]. Unsupervised learning techniques, such as clustering and k-nearest neighbor [61], are also available. Hybrid solutions have emerged in the past years, fusing complementary techniques to address each other's weaknesses. Case-based and rule-based hybrid reasoning is explored in order to improve CA applications in [62]. Ontologies, rules and Bayesian networks are exploited in [16] for context reasoning under extreme conditions (underwater). The middleware system proposed in the current work integrates a hybrid reasoning schema that applies specific reasoning techniques based on their suitability to address specific sub-problems.

2.4. CA Middleware Systems

Many middleware systems are proposed in many IoT installations, typically for sensors abstraction but also for covering the modeling and reasoning functionality of the context life cycle. Next, a non-exhaustive list of middleware solutions for various IoT applications is presented. Razzaque et al. [7] present a comprehensive survey of middleware, including pioneer solutions, such as Gaia [63], Carisma [64], CoBrA [65] and Hydra [66]. Among the later approaches, C4IOT [67] focuses on fusing context data from various streams in order to produce more comprehensive and meaningful information. Zhang et al. [68] present a CA middleware that is specialized in mobile applications by recognizing user behaviour accurately. Machado et al. [69] propose a hybrid architecture that exploits data correlation for enriched context awareness. Their proposed system, called HACCED, integrates rules, ontologies and supervised learning toward a hybrid processing layer. Kim and Yoon [70] describe an ambient intelligence middleware that supports multi-modal context data. Their novel awareness cognition framework aims to address the scalability issues of processing multiple context sources. Belcastro et al. [71] present a middleware for mobile environments that focuses on scalability. Kali-smart [72], an autonomic CA platform for mobile pervasive environments, exploits semantic services to provide semantic multimodal event detection, centralized semantic decision making and optimization of the response time of the situation.

Middleware systems applied in cultural spaces are very scarce. Chianese et al. [73] present a platform hosting an integrated middleware system named gateway server, which performs the acquisition of information from mobile devices for cultural institutions. The authors extend their work by adding a context manager, which stores data from the environment and the user in [25]. Dossis et al. [74] propose IRME, a framework for smart museums that transfers middleware functionality to the Cloud. SCRABS [22] adapts a generic middleware called Perla, integrating it into their CA assistant for cultural environments. The multi-protocol middleware for location-aware indoor smart museums implemented in [75] integrates transparent accessibility to heterogeneous IoT protocols, a functionality typically executed by middleware. Finally, the middleware integrated into the framework described in [76] manages installed sensors in a recommender system for cultural paths.

The middleware solutions presented above and summarized in Table 1 do not fully integrate every stage of the context lifecycle. They typically address heterogeneity issues and provide abstraction of the sensor layer, while providing limited context modeling of location and user profiles, and also limited reasoning. In contrast, the proposed middleware integrates a complete context model and a hybrid reasoning technique, addressing all stages of the context life cycle and supporting a fully operational CA computing procedure in CH environments.

Table 1. Overview of middleware for CH environments.

Name	Modeling	Reasoning	Heterogeneity	Scalability
SmartTweet [73]	Key-value	Rules	No	No
Location [25]	Key-value	Binary Tree	No	No
IRME [74]	Graphical	Rules	Yes	Limited
SCRABS [22]	Ontology/key-value	Rules	Yes	Yes
MUST [75]	Key-value	Rules/ML	Yes	Yes
Paths [76]	Graphical	Rules	Yes	No

3. Context Modeling

3.1. Adjustment of Categorization Schemata

Reviewing the existing categorization schemata (see Section 2.1), and based on the premise that the ideal context model should be based on a combination of schemata in order to capture context from two perspectives, the following three categorization schemata were selected: 5 Ws [30], operational schema of [33] and categories of event and time-driven contexts [12]. Each of them captures different aspects of context, while their synthesis can be exploited to develop a better context model, which not only represents the conceptual correlation of the data to a situation, but also the circumstances of data generation and their variability. Such a model will be able to capture the context of an area (conceptual aspect) but also describe which data need to be updated on what condition (implementation aspect). Next, an extension and adaptation of each one of the original schemata included in the proposed context model are presented.

On the first schema [30], the who corresponds to the user but in the proposed model, it is extended to any smart device of an IoT ecosystem as an autonomous entity that exploits context awareness. The what is closely related to activity, but may also include a service (as a potential activity). The when encompasses two types of temporal contexts: the timestamp, which is related to an observation, and the period during which an observation is captured (e.g., during bedtime). The former is not considered an autonomous contextual entity but a parameter of a sensed observation or derived information. The where is conceptually related to locations of varying size, but traditional localization cannot represent a translocation activity, which should be addressed by complex location entities, such as “along a bus ride”. Lastly, context data corresponding to the why question are either derived information, meaning that the reasoning process assumes why a certain activity is executed, or explicit user input.

On the second schema [33], a merging of static and profiled data is proposed since from an implementation point of view, an IoT platform does not distinguish between them. The necessity lies in the knowledge of whether the required contextual data are sensed (thus a series of actions, such as collection, fusion and processing, are performed) or not (thus only data retrieval is performed). On the other hand, sensed data are split between sensory data and user input, a distinction that interests a CA system during a context retrieval procedure and affects its reliability and responsiveness. In conclusion, the adapted schema includes the following types of data: user input, sensed, profiled, and derived.

The third schema [12] represents context that is created based on an event or on specific timeframes. A third type of context is added in the proposed model, context that is created on demand. Let us illustrate the differences between those three types with a temperature sensor as an example. The sensor may send its observation every five minutes (time-driven), every time it falls under a certain value (event-driven), or every time it is asked to do so. Some sensors may be able to function under all three conditions, but most will only be programmed to function under one of them. A fourth type of context is proposed that is archived and is related to past observations. A CA application needs to know whether a required contextual parameter can be pulled on demand, retrieved at specific time ticks, captured only on specific scenarios or retrieved from a repository. From an implementation point of view, the availability of contextual data is validated with the use of this schema of time-driven, event-driven, on demand and archived data.

3.2. Context Model

At the core of the proposed context model lies the thing—the abstract building block of the IoT. It is associated with the *who* feature. Things can be users, devices or other objects, and they are defined by a list of properties that are usually profiled data. There are two main observations that need to be captured by the context model: the state of each thing and the interactions among them. In essence, both types of observations can be combined into one, introducing the axiom that “Each observation includes one or more entities in a temporal and spatial extent”. The *where* and *when* features are thus integrated into the scheme. Furthermore, many observations include a target entity, the *what* feature, which may be one or more entities that are related to the observation in a passive way (e.g., an activity undertaken by an individual). The combined set of thing(s), location, time, and target(s) defines an observation, which is captured per activity. Thus, the proposed context model is responsible for representing all those sets of observations. Finally, the *why* feature is derived data from the inference procedure, or explicit user input. Aguilar et al. [47] present their CA meta-ontology with a hierarchical structure that includes a general layer and a domain-specific layer. They describe six contextual classes, exploiting the 5 Ws schema to represent any possible contextual information. Although conceptually their work succeeds in modeling context efficiently, it is missing the implementation aspect that allows the identification of data sources and flow and the representation of the inference process and output. Influenced by this meta-model, the following core contextual classes are used as the basis of the proposed model:

- Thing: User or device.
- Location: Any entity that describes a location of varying range, such as point, area, room, building, district, etc.
- Time: Any period of time that has a meaningful representation, e.g., lunchtime or commute time. Specific timestamps are not objects of the class.
- Activity: Any potential activity or status of a thing.
- Asserted context: The set of contextual parameters that infer the identification or triggering of an activity.

Those five core classes are connected with relationships as illustrated in Figure 1. Each observation tuple is represented as a single instance of this relationship graph for a single observation. This instance describes an activity (durative or instant) or status change that occurs at a specific time and includes things that are located somewhere, while the event

is identified or triggered based on asserted context data. Each class is analyzed in the following sections.

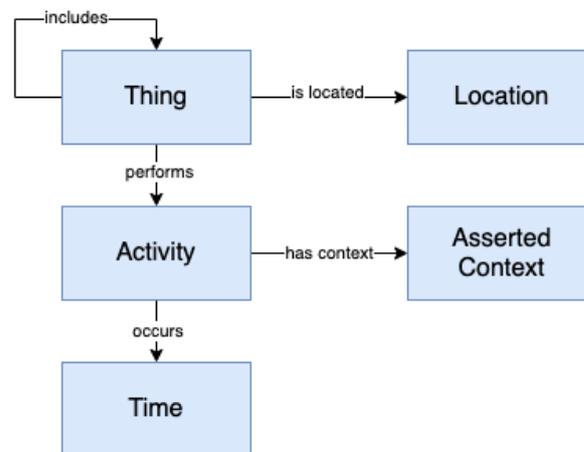


Figure 1. Relations among the five core classes of the model.

3.2.1. Thing Class

A *thing* has two subclasses: *user* and *object*. Each subclass has static properties, such as *id*, *name* and *birthdate*, for the *user* subclass. Furthermore, each thing has multiple *profiles* that define the preferences, requirements or restrictions for a specific situation. Each profile has specific time and location constraints and may be reusable by multiple thing objects. For example, two users may share a profile that sets specific environmental preferences for the office room during winter months.

A thing has multiple *roles* which also have time and location constraints, e.g., a user is a museum administrator during his/her work hours. *Relationships* correlate things, e.g., defining friendships, husbandry for users and ownership, and segmentation for objects. Relationships are not typically constrained but active all the time until an event disables them. Each thing has a location and is related to multiple activities. Both of these connections are not static but derived from the inference procedure which, as described in Section 4, is constantly executed to keep such information updated. Figure 2 illustrates the thing class.

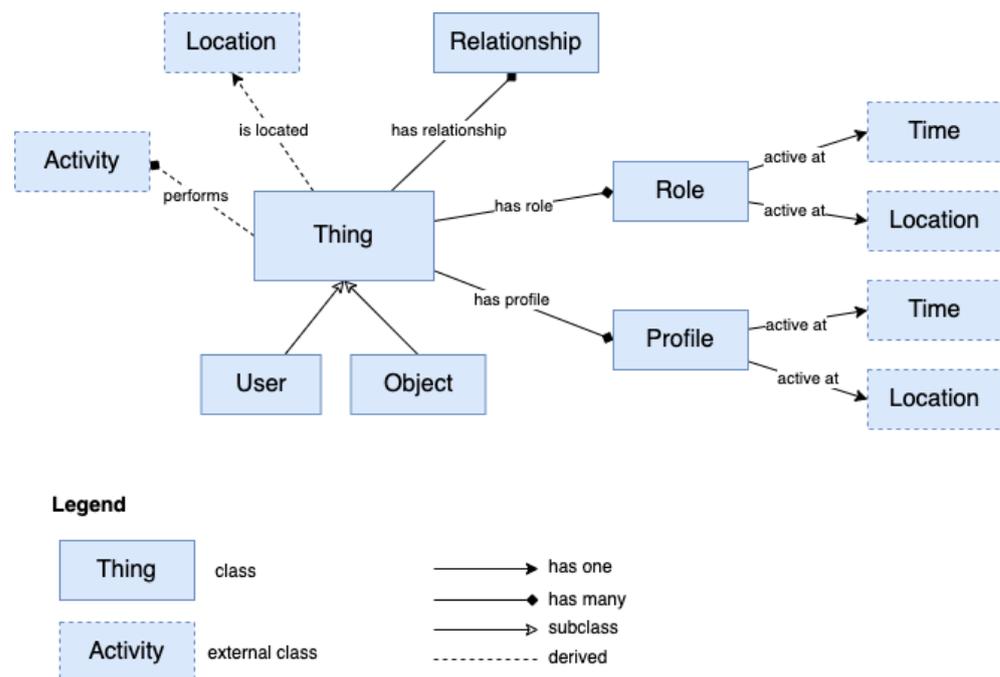


Figure 2. Thing class.

3.2.2. Location Class

A *location* can be either *indoors* or *outdoors*. Although other inheritance approaches are valid for the definition of the location object, such as building, floor, corridor, and room scheme presented in SeCoMan ontology [44], this work argues that conceptually, there is no need to integrate such differences in the location class. A location may belong to other locations, thus modeling the inheritance functionality. Apart from the self-referential relation of *belonging*, a location may also be *adjacent* to other locations, modeling spatial proximity. A thing can be located to a location which may be determined with high accuracy (e.g., a room, or an exact geolocated point), or with lower accuracy (e.g., a building, or a district), depending on the availability of sensory data and necessity. Multiple things may be located in the same locations, adjacent locations or locations of the same hierarchy (e.g., building), allowing inferences on proximity and interaction.

A location has a *position*, which may be defined by absolute measurements such as exact geo-positioning and effective radius for points or geo-constrained spaces for outdoors areas, or by relative positioning such as the indoors rooms, corners, areas. The different subclasses of position allow the modeling of both indoors locations whose absolute position is not applicable (e.g., floors with same geolocation) and outdoors locations that can be easily defined by absolute positioning systems (e.g., GPS). A location may have many *profiles* with either a *general* scope to model various properties and requirements, or an *authorization* scope to model accessibility and privileges.

Finally, a location may have an *environment*, which models the environmental parameters such as temperature, humidity, etc. It is a derived relation since it is computed by the fusion of sensory measurements that are related to that location. Thus, the calculation of the temperature of a room is based on the data collected from relevant sensors that are located in the room, either fixed or mobile from the users also present inside. Figure 3 illustrates the location class.

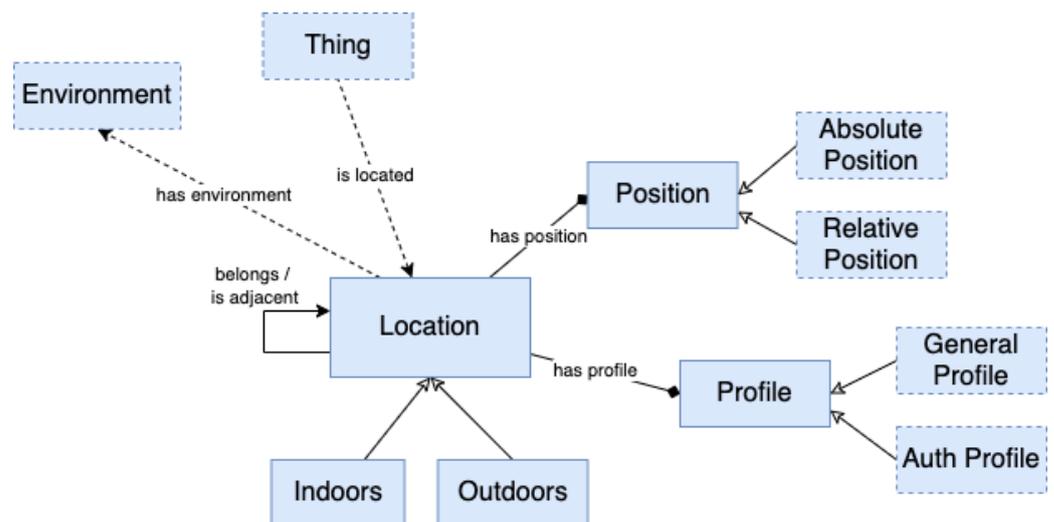


Figure 3. Location class.

3.2.3. Time Class

The *time* object models the temporal properties of activities. Time in the proposed CA model, though, is not limited to activities (events) that occur and need to be timestamped. It is also used to model scheduling, which is also related to activities and indicates tentative events. Thus, an activity is either connected to time with a derived relation that is captured upon the activity onset, or with the profiled relation *is scheduled*. The time class is illustrated in Figure 4.

Furthermore, a time object may be *instant*, an exact timestamp without duration useful to capture instantaneous events such as a door ringing, or *interval*, a timeframe with duration, usually with a conceptual meaning, e.g., Monday work hours. This hierarchy is influenced by the OWL time ontology (available online: <https://www.w3.org/TR/owl-time/> (accessed on 10 June 2021)). A time object may also have multiple profiles, which may indicate requirements or restrictions. For example, the time object “S3-MD23 Exams” may have a profile that requires a flight mode for all participants’ mobile phones, independent of the thing, activity and location involved.

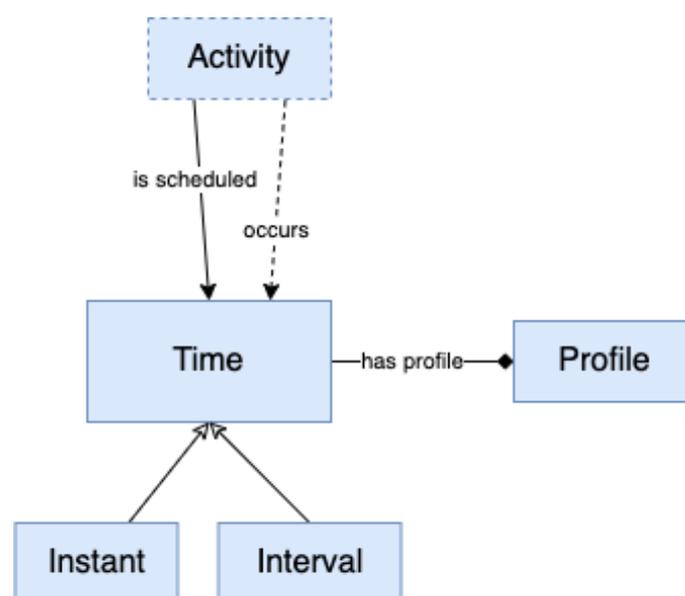


Figure 4. Time class.

3.2.4. Activity Class

An *activity* may be either *instant*, e.g., the air-conditioning was turned on, or *durative*, e.g., the TV is showing a movie. Some models presented in the literature distinguish between action and event, implying an active (caused by users) or passive nature. Since the proposed model represents both user and device in the same class, all activities can be assumed to be triggered by a device or user, thus a thing object. Furthermore, conceptually, it is more efficient to model the triggering of an activity by capturing all of the related context.

An activity may be part of or occur after another activity, properties that are modeled with appropriate self-referential relations. As already mentioned in the time class, an activity engages one or more thing(s). The activity class is not directly related to a location, but rather indirectly through the engaged things and their locations. Thus, an activity occurring in different locations (e.g., a phone call) can be accurately modeled.

This class illustrated in Figure 5, aims to represent a wide range of cases, from typical human activities, to events triggered by human decisions or device actuation. Furthermore, it models not only observations of occurring activities, but also tentative activities. In order to model this distinction, an activity is related to the time class in different ways. It may occur in a specific timeframe, or is connected to time through the *is scheduled* relation. An activity has similar profiling classes with location: general profile and authorization profile to model privileges of things in respect to the activity, e.g., users permitted to turn on/off a device.

Finally, the measurement of a sensor device is considered an activity and can be accurately captured by the activity class. It includes a thing, a timestamp and a value. Similarly, a service is considered an activity (a tentative action to support a need). Overall, the activity class is wide enough to include all types of systems or user behaviors.

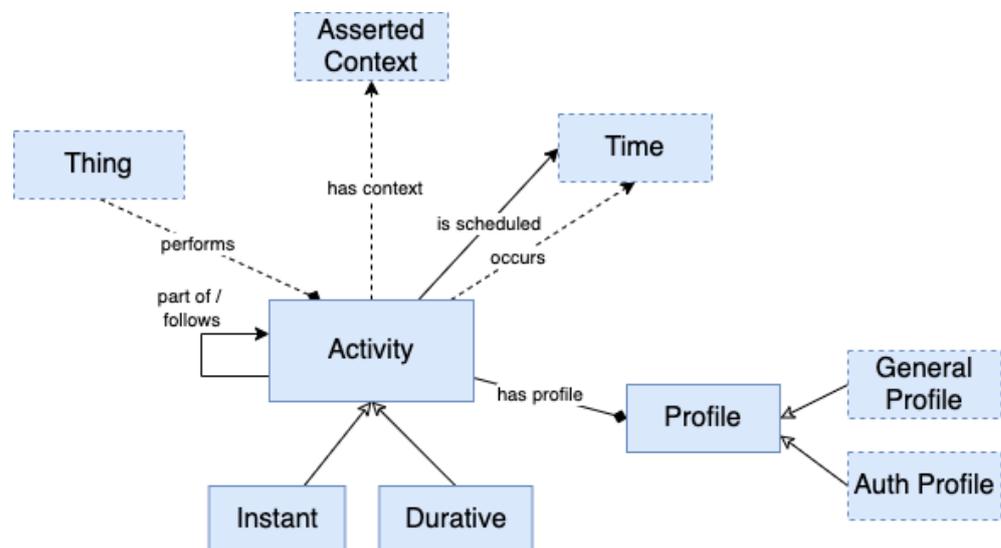


Figure 5. Activity class.

3.2.5. Asserted Context Class

The *asserted context* class models the *why* question and is related to the circumstances of an observed action. Thus, an asserted context object is created for each observation and captures things, locations, activities and times correlated to that observation. For example, let us assume an air-conditioning device that was automatically turned on. The action was triggered because of specific contextual data, such as a person (thing) entered a room (location) whose profile indicated a higher preferred room temperature (rule) than the measured temperature (activity) during a specific time period (time).

The asserted context class allows the modeling of inference results, which can be exploited during subsequent inference procedures. Not all actions need an asserted context

object, such as those that happen on a scheduled basis (e.g., sensor measurements), but most of the observations require it to capture the reason behind their emergence. This class is illustrated in Figure 6.

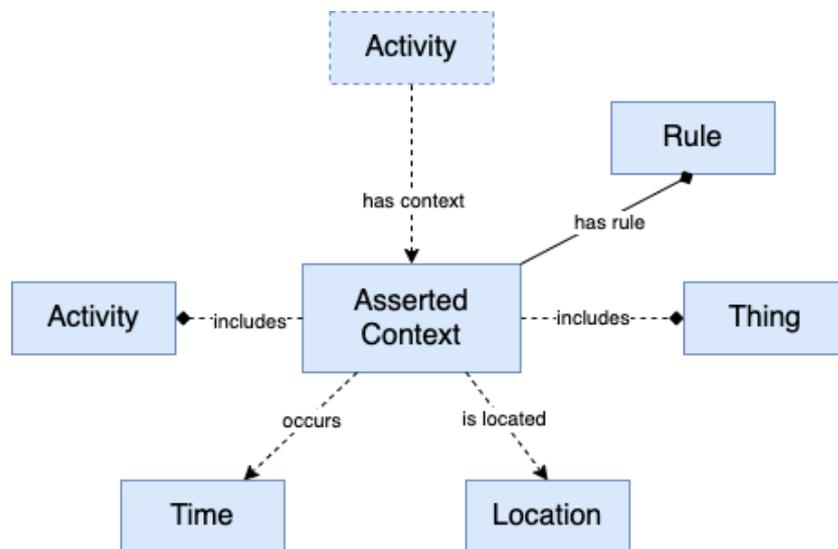


Figure 6. Asserted context class.

3.2.6. Class Properties

The classes described in Figures 2–6 have properties that define intrinsic features (called data properties in ontologies or fields in relational databases) and properties that relate them with other classes (called object properties in ontologies or foreign keys in relational databases). The number and range of such properties are influenced by the application and level of automation provided by the CA system. An indicative presentation of the location class and its properties is shown in Tables 2 and 3.

Table 2. Properties of outdoor location class.

name	Short text to name the location
description	Long text to describe the location
shape	Point, circle, polygon
latitude, longitude	An array of points that define the area
radius (optional)	Used in circle shapes
address (optional)	Possible address of the location

Table 3. Properties of indoor location class.

name	Short text to name the location
description	Long text to describe the location
type	building, floor, room, corridor, etc
order	Numerical order of floors, rooms etc
adjacent_ids [Array]	An array of location IDs that are adjacent to the location
part_of (optional)	ID of location that the current location is part of
address (optional)	Possible address of the location

4. Context Reasoning

4.1. Reasoning Challenges

Although the process of modeling and storing contextual data has its own merits, unprocessed data collected from the sensory layer of the IoT infrastructure have conceptually low importance and usefulness. Thus, the integration of context reasoning, the next step according to the context life cycle, is fundamental in fulfilling the purpose of CA systems,

i.e., the dissemination of high level contextual information to interested applications and devices for efficient service delivery.

The main challenges that a reasoning procedure for IoT ecosystems should address are (a) complex representation, (b) uncertainty or lack of contextual data and (c) computational performance [77]. In other words, the inference engine should be able to model and perform the logic despite its complexity to tackle situations where contextual data are missing or noisy and to respond in real time. Each one of the above challenges will be further analyzed with respect to the methods that best address them.

Complex representation requires that the appropriate method can be exploited in every real-world situation. Supervised and unsupervised methods excel at solving known problems (such as activity recognition) but are not suitable to support a generic reasoner that addresses a variety of situations. Still, ML models can be used to solve the specific problem that they were trained for and share their output with the main reasoner. Probabilistic and fuzzy logic are niche techniques that supplement the reasoner with their ability to tackle uncertainty and probabilities instead of facts. Thus, only rules and ontologies support complex representation and can be used as the basis for a reasoner that addresses a wide range of situations, as required by the IoT paradigm.

Contextual data in an IoT ecosystem can be considered big data, thus inheriting their traits and characteristics. Similarly, they inherit uncertainty due to noise, incompleteness, and inconsistency [78]. The most common techniques to mitigate the effects of uncertainty on context reasoning are fuzzy logic and probabilistic logic [9]. Fuzziness as a technique to address uncertainty is introduced by [79], while [80] combine fuzzy logic with rules to implement a decision support system.

Finally, computational performance is related to the ability of the reasoner to produce an updated, high-level context when required by the IoT applications and devices. Ontologies may show decreased performance as the ecosystem scales up and the volume of data increases [9], although Instance Store [81] was introduced to partially solve the issue. Rule-based logic, being less resource intensive, shows better performance and behavior with scalable ecosystems [6].

4.2. Proposed Hybrid Reasoning Process

The review of reasoning techniques strongly suggests that hybrid systems are required for IoT middleware and have indeed been introduced on many occasions. This argument is further extended here, claiming that a fully deployed reasoning procedure of IoT ecosystems requires the integration of most reasoning methods, each one addressing those situations that better fit their capabilities. The architecture of the proposed hybrid reasoning process is illustrated in Figure 7.

The reasoning procedure is split into middleware layer processing and application layer processing. The middleware, which is also responsible for collecting and modeling the context, performs low-level fusion of contextual data into high-level contextual information that is usable for interested applications. The application is responsible for performing its own reasoning exploiting high-level context in order to initiate appropriate actions. Figure 7 also contains part of the environment, including notations for IoT sensors and actions performed in order to illustrate the input and output of CA processing.

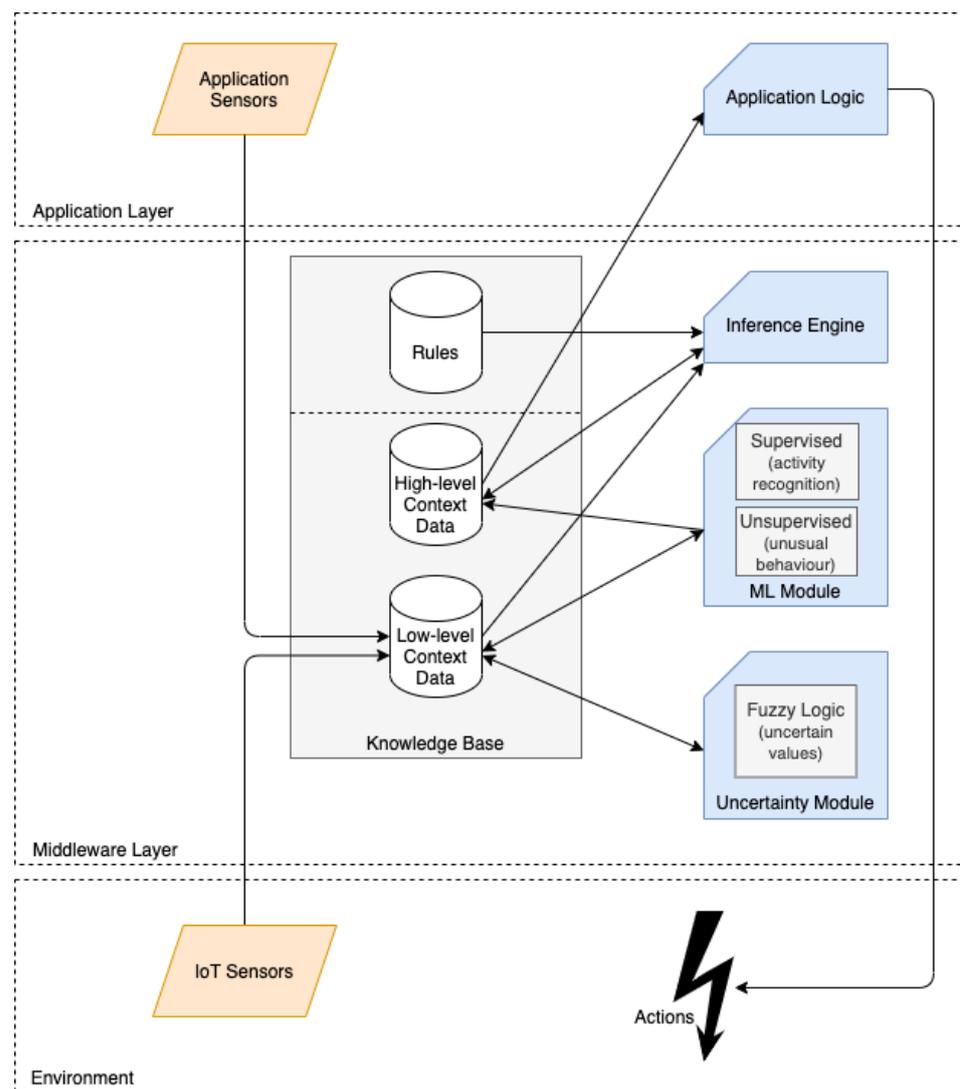


Figure 7. Architecture of context reasoning.

4.2.1. Middleware Layer

Reasoning starts as soon as the first raw contextual data are collected, which includes three steps: pre-processing, sensor fusion and context inference [39]. Pre-processing and sensor fusion can be enhanced with fuzzy logic, which transforms numerical data into meaningful literal categories, thus addressing uncertainty. The *uncertainty module* works in parallel with the context acquisition procedure increasing the quality and expressiveness of sensed data. The output is still considered low-level context.

The *ML module* includes all the ML models, which are built to solve specific CA problems. Unsupervised learning techniques can facilitate the preprocessing step by performing the detection of outliers or unusual measurements, while supervised learning techniques excel at identifying missing values [82]. Pretrained ML models can also tackle specific problems, such as activity recognition. The ML Module receives low-level data and either enhances them or delivers high-level information, working independently of the other modules.

The *inference engine* tackles the core domain logic, i.e., the rules and relations which are valid in the domain. They process low-level and high-level data and produce high-level data which are meaningful inside the IoT ecosystem. Ontologies and/or rules can be used to perform the inference, depending on the requirements and limitations of the domain. The selection of the technique does not affect the architecture or the function of the other modules but determines the performance, expressiveness, scalability and validity of the

inference engine. The choice is highly affected by the domain, while hybrid solutions are applicable as well.

4.2.2. Application Layer

The application layer is responsible for performing reasoning that results in tentative actions since the actuation is closely related to the application. The application logic refers to all the rules that define events, actions and services supported by the application. It retrieves high-level contextual information and decides upon equivalent actions. The actuation is excluded from the middleware layer, thus if specific actions are needed in the domain scope, an appropriate universal application should be built to tackle them. In this way, it is easier to handle the permissions and privileges of actions and allow the distribution of a reasoning load.

Collision detection is necessary when conflicting rules are applicable. Similarly, collision may happen when entities with antagonistic profiles exist in an area. Collision issues are addressed application-wise but are executed server-side to involve all participating entities. Thus, the implementation and execution of application logic is executed on the server, while the actuation is performed by the client. For example, the calculation of a user's phone ringing volume is executed on the server in order to decide whether it collides with the restrictions of the area and other users.

5. Case Study: Smart Conservation of Cultural Heritage

In order to be assessed, the proposed modeling and reasoning methods were applied to a smart conservation case study for CH spaces. An IoT infrastructure composed of sensors, beacons, middleware and other smart devices, was installed in a CH Institution. The sensors, installed on Arduino mega boards (available online: <https://store.arduino.cc/arduino-mega-2560-rev3> (accessed on 10 June 2021)), measured temperature, relative humidity and light intensity. The middleware collected data from the sensory layer and the user devices, using Bluetooth Low Energy and Wi-Fi protocols. The acquired data were appropriately modeled and stored in a NoSQL database (MongoDB). The reasoner was executed to identify actions that were needed for the conservation of the displayed CH artifacts. In the next subsections, the modeling and reasoning procedures of the case study are described in detail.

5.1. Modeling a Smart Cultural Heritage Space

Based on the context model described in Section 3.2 a number of required subclasses were identified for the realization of a smart conservation ecosystem. The results of the instantiation of the model for the case study are shown in Table 4. The subclasses involved are as follows:

- Thing (user): Includes the three types of users, i.e., the curators and the conservators who are recipients of proactive notifications about conservation issues and the visitors of the cultural space.
- Thing (device): Includes (a) the IoT infrastructure, i.e., sensors for environmental monitoring, beacons for location and proximity measurements and smart devices for environmental regulation and (b) devices of visitors and staff that are recipients of notifications, but also allow user localization.
- Location: Includes the building layout (rooms and corridors) and the points of interest (POIs), which are tagged locations of artifacts.
- Time: Includes the different timeframes that affect the environment, i.e., closer hours (minimal effect), visit hours (normal effect) and rush hour (maximum effect).
- Activity: Includes IoT activity (sensor measurement and manipulation of environmental devices), visitor activity (their movement and behaviour in the cultural space) and notification activity (for visitors and staff).

Table 4. Modeling the smart conservation.

Main Class	Subclasses
User	Visitor, curator, conservator
Device	Sensor, beacon, visitor device, environmental device, staff device
Location	Room, corridor, POI
Time	Visit hours, rush hour, closed hours
Activity	Sensor measurement, environmental device manipulation, visitor activity, visitor device notification, staff device notification

Although the main classes are designed for generic CA ecosystems, they are easily adapted to the case study of the preventive conservation of CH. Figure 8 shows indicative examples with objects of the main classes that comprise part of the context model instantiation used in the present scenario. Specifically, a room with an artifact and a beacon installed on it are depicted with indicative object properties. Both the room and the artifact have a profile that stores the environmental preferences and visitor restrictions (such as the number of maximum visitors, or their minimum allowed proximity). The artifact profile is active on Monday during visiting hours (other possible profiles are active on other days). The artifact is paired with a beacon, which measures proximity with other BLE devices. Finally, the position is absolute for the POI (in terms of coordinates) and relative for the room (in terms of building layout).

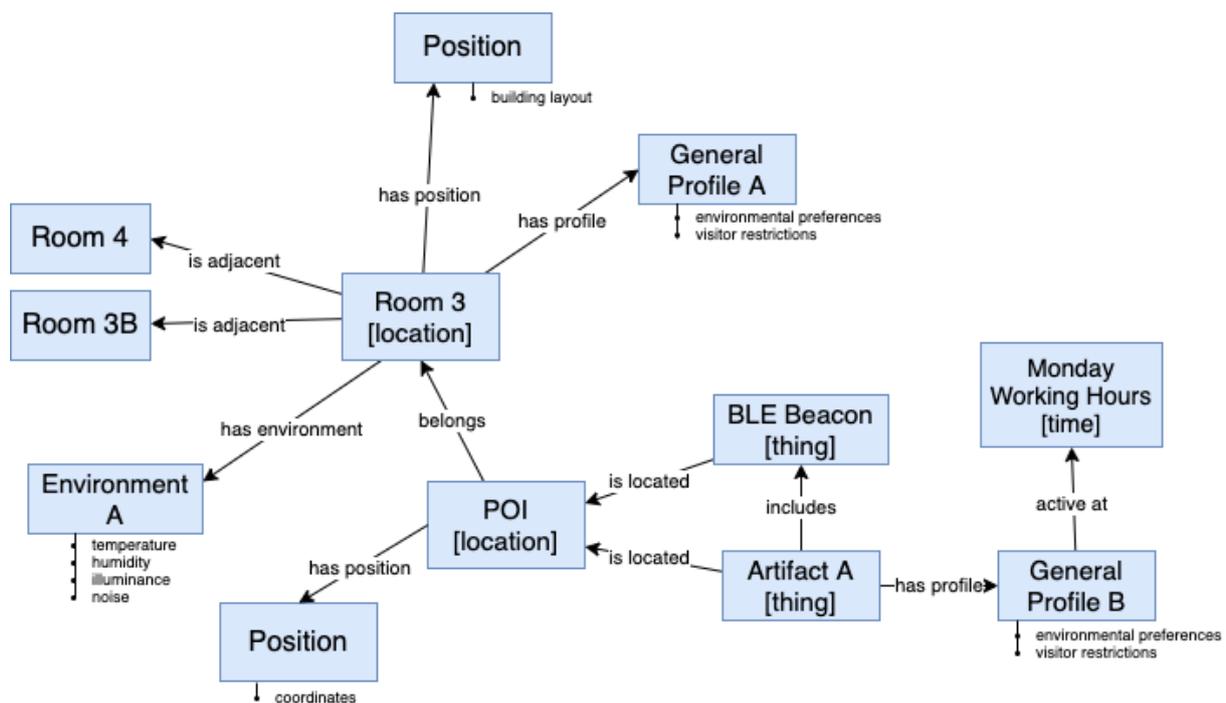


Figure 8. Examples of main class objects.

The implemented subclasses and produced observation tuples model all the contextual situations that are applicable to the conservation process. Examples of such observation tuples are shown in Table 5, which illustrates two cases, each one of them composed of two tuples. The first case represents the measurement of abnormal temperature value that triggers the manipulation of an air conditioning device. The asserted context for the second tuple stores the triggered rule and the relevant contextual data, which in this case, is the previous observation. The second case represents an irregular visitor behavior, which is captured by the proximity of the visitor’s device to a beacon installed on an artifact (POI location). This observation triggers a staff notification activity, which is targeted to the device of the closest available staff member. All relevant asserted contexts, such as the

observation of visitor behavior, the triggered rule, the available staff (from their profiles and working hours) and their proximity are stored in the tuple in order to record the context involved in the reasoning.

Table 5. Examples of observation tuples.

ID	Activity	Thing	Location	Time	Asserted Context
2F1	Temperature measurement	Sensor: 456FA	Room 6	Visit hour Monday	-
2F2	Turn on A/C	A/C: 6B	Room 6	(exact timestamp)	Observation: 2F1 Rule: 17AF
			...		
2FC	Visitor Proximity	Visitor Device: 0122	Room 4	Rush hour Monday	-
2FD	Staff Notification	Staff Device: 0023	Room 3	(exact timestamp)	Observation: 2FC Profiled data on staff Sensed data on staff location Rule: 15ED

5.2. Reasoning in a Smart Cultural Heritage Space

A hybrid reasoning system was implemented based on the generic reasoning architecture presented in Figure 7. The reasoning system was executed on a machine running macOS Catalina, equipped with a six-core i7 processor and 16GB RAM. The reasoning scripts were coded in Python 3.7, including libraries, such as *scapy* for packet manipulation, *numpy* for computing, *sklearn* for machine learning, *pandas* for data manipulation and *pymongo* for database connectivity. The inference process is composed of three autonomous modules.

5.2.1. Uncertainty Module

When contextual data collected from uncertain sources (e.g., sensors) are required, the uncertainty module is responsible to pre-process such data by fusing different sources and tagging them with a confidence value that is related to the sensor quality, oldness of measurements and other parameters. Furthermore, if requested, the uncertainty module provides fuzzy sets instead of crisp sets to represent the required property. Defuzzification is performed by the *rules module*.

5.2.2. Machine Learning Module

In order to identify abnormal observations of environmental parameters, ML algorithms were exploited in an autonomous module that is executed in parallel. Specifically, outliers detection [83], an unsupervised learning method, was implemented, which detects irregular observations of temperature, relative humidity and illuminance. The *sklearn* (available online: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html> (accessed on 10 June 2021)) isolation forest method was used with the following parameters: *estimators* = 100, *max_features* = 2.0, *contamination* = 0.05 and *max_samples* = 256. This technique of identifying abnormal observations was used in addition to traditional *if-then* logic rules, enhancing the procedure with automated detection that does not require manual definition of the rules but exploits the ability of the model to identify patterns and isolate irregular numbers. The resulting output is stored in the knowledge base (KB) and used by the rules module when needed.

5.2.3. Rules Module

The rules module performs the core of reasoning processing. It requests sensory data from the uncertainty module, retrieves profiled data from the KB and performs reasoning in real-time. The module is continuously executed and transforms low-level contextual data into high-level contextual information. It is not application specific, thus it can be exploited by any application that is active inside the smart cultural space. The rules follow *if-then* logic, but also take into consideration confidence and fuzzy logic, dealing with probabilities rather than certainties, tagging all high-level contextual information

with a probability score. Specific rules include the triggering of an action when sensory measurements surpass predetermined values, such as the rule shown in Figure 9.

5.2.4. Application Module

The actuation process is not performed by the middleware since the application is responsible for performing such actions based on the reasoning output. Thus, an application module is also implemented, which takes high-level information and acts upon it. The necessary actions for the preventive conservation of CH are performed in this module, which is executed independently. For example, the application module checks whether the temperature of a room is appropriate, whether visitors are keeping a safe distance from artifacts or whether a room is crowded, thus any more visitors should be avoided. The inference engine of the application module follows *if-then* logic, also taking into consideration the probability score tagged to contextual information. Thus, a low probability event may not trigger the equivalent action depending on other parameters, such as temporal restrictions.

The implemented application module addresses the necessity for the automated preventive conservation for cultural spaces. Other applications that could be built on top of the existing middleware may perform personalized visits, recommendation services, management and optimization of exhibitions, etc. The open architecture of the proposed infrastructure allows the interconnection with any number of applications that require contextual information.

5.3. Evaluation and Results

The proposed CA middleware was deployed for a period of 8 weeks. Initially contextual data were collected by a relatively small number of installed sensors, but tests were also performed on scalability and performance with the artificial data of thousands of sensors. The behavior of the proposed middleware was evaluated with respect to the following criteria: (a) representation, (b) reasoning veracity and (c) computational performance and scalability.

5.3.1. Representation

The proposed modeling method (see Section 5.1) implemented in the proposed middleware efficiently represents the context of a cultural space, capturing sensed and profiled data and modeling them appropriately. The representation of context exploits the NoSQL characteristic of non-strict data structures, which preserves semantics and allows interoperability.

MongoDB, the NoSQL DB of choice, proved to be very efficient at sustaining a large variety of data collections. Although the application built on top of the middleware, addressing preventive conservation, had limited interaction with user devices, it required a diverse range of sources, including user and staff devices, sensors, environmental devices and smart lamp infrastructure. Overall, the observation tuples proved to be an efficient tool to capture events, which are the building blocks of context-awareness.

5.3.2. Reasoning Veracity

BSON (a JSON variant) collections were utilized to model rules. For instance, Figure 9 shows an example of a rule that checks whether the temperature of a sensor has surpassed a certain threshold. Such rules were defined for various sensors, artifacts and rooms, including in their criteria for sensed data acquired from user devices. Artificial emergencies were initiated to test the system's response. Additionally, conflicting rules were defined to test collision detection and resolution. Finally, error-prone sensors were mimicked in order to test the uncertainty module. Each testing case was successfully addressed by the middleware.

```

    _id: ObjectId("5ec7a30edc36d6d33c963dc2")
    sensor: "123FF21"
    text: "temperature>=25"
    rule: Array
      0: Object
        type: 1
        op: Object
          param: "temperature"
          threshold: 25
          comparison: "gte"
    action: 2

```

Figure 9. Rule in BSON format stored in the DB.

Figure 10 shows an instance of the real-time monitoring script, which checks all rooms, and their equivalent rules for required actions. In this case, two actions were performed based on sensory measurements that required supervisor communication. In each case, the user's preferred type of notification (SMS and email) was selected. The script (written in Python) was continuously executed and performed actions on a real-time basis.

```

Checking rules for Room: 1
=====
Total actions: 0
=====
Checking rules for Room: 2
Action required for sensor 123FF25: SMS Manager
  Parameter: temperature, value: 24.5642, rule: temperature<=25
  Sending SMS to 00306982501914
Action required for sensor 123FF25: Send email to supervisor
  Parameter: humidity, value: 50.69856, rule: humidity >= 50
  Email sent to kmichalakis@gmail.com
=====
Total actions: 2
=====

```

Figure 10. Screenshot of real time monitoring script.

5.3.3. Scalability and Performance

The experiment was initiated with a small-sized installation of approximately 50 deployed things (including sensors, beacons, mobile devices). During the first weeks, the CA middleware did not show any performance bottlenecks, which is expected due to the low volume of contextual data and rules to be checked. In the following weeks, the number of deployed things was gradually increased with artificially created sensors, beacons and mobile devices, which were emulated with scripts. Similarly, the number of rules was gradually increased, although by a lower factor, emulating the extended logic due to larger volumes of contextual data.

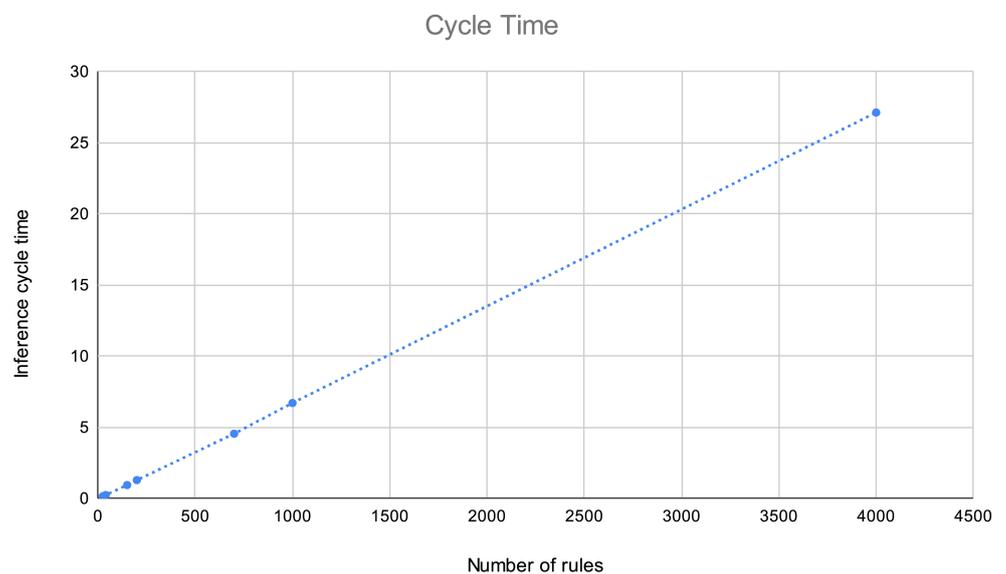
For the evaluation of the performance of each week, the following indices were calculated: (a) *inference cycle time*, which is the average time that a complete check of all rules was performed and (b) *response time*, which is the average response time for an emergency. Table 6 summarizes the findings for each index during the 8 weeks.

Table 6. Performance results.

Week	# Things	# Rules	Inference Cycle Time (s)	Response Time (s)
1	50	25	0.15	0.10
2	50	25	0.15	0.09
3	100	40	0.24	0.13
4	500	150	0.94	0.51
5	1000	200	1.29	0.66
6	5000	700	4.55	2.18
7	10,000	1000	6.71	3.54
8	50,000	4000	27.12	15.12

The inference cycle time, which is indicative of how quickly the middleware can perform a complete scan of CA situations that require attention, follows a linear correlation to the number of rules. Figure 11 illustrates this correlation. Similarly, the inference cycle time follows a linear correlation to the number of things, which is shown in Figure 12. The response time follows a linear correlation to both the number of rules and things. Both indices indicate that the middleware scales linearly with the number of added rules and things, which means that it scales well with higher volumes of context data.

By integrating hybrid reasoning, the proposed middleware manages to both capture and address a wide range of CA situations and show efficient scalability with increased sensor nodes. The combination of computationally heavy machine-learning methods with lightweight rules offers a solution that exhibits flexibility in representation and efficiency in performance. Overall, the findings justify the need for a scalable middleware that can address the requirements of a CH environment in terms of context representation and reasoning without performance loss.

**Figure 11.** Correlation between inference cycle time and number of rules.

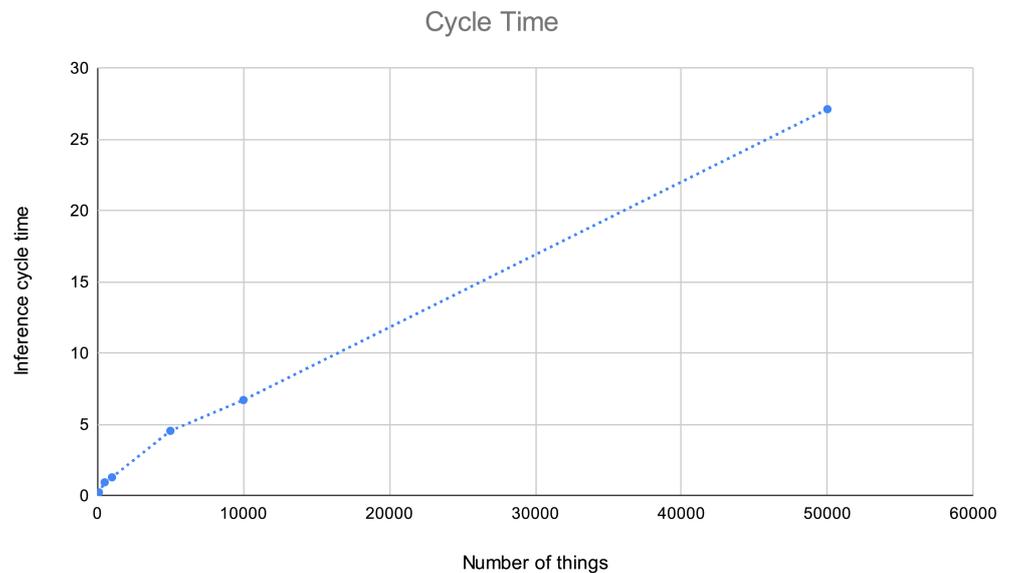


Figure 12. Correlation between inference cycle time and number of things.

6. Conclusions

In this article, we presented a CA middleware system that integrates context modeling and reasoning techniques. First, an adaptation and combination of the most prominent context categorization schemata was proposed, which formed the basis of the context model. The proposed model includes five core classes, namely, thing, location, time, activity and asserted context. A hybrid reasoning schema, which combines multiple techniques, was also proposed. Each individual reasoning technique was applied to solve problems that it is more suitable for, providing an efficient reasoning procedure that can output all types of inferences.

In order to be assessed with respect to representation, reasoning veracity and performance, the proposed middleware system was employed in a cultural space, with the purpose of facilitating preventive conservation of artifacts. The evaluation results showed that each criterion is satisfied, that the system can accurately represent all required context and that the integrated reasoning procedure can efficiently address various types of contextual emergencies. Additionally, the performance tests showed a linear correlation between response time and volume of things, which proves the scalability of the proposed middleware.

In the future, the proposed middleware and the underlying modeling and reasoning techniques will be employed in more activities inside cultural spaces, i.e., recommendation systems, personalized visits, management and optimization of exhibitions. Thus, the efficiency of building multiple CA applications on top of an existing middleware will be evaluated, analyzing the system performance and ability to address multiple context stakeholders and manage their possible conflicting requirements. Furthermore, since the current work does not address connectivity and network performance, such open research questions will be tackled in future work. More specifically, future evaluation of the proposed middleware will perform the testing of network performance metrics, such as latency and overload, including a comparison between the proposed KB installation (mongoDB cloud) and other solutions. In addition, advanced connectivity open research questions will be researched, including the automatic discovery, registration and configuration of sensors and smart objects inside a cultural ecosystem.

Author Contributions: Conceptualization, K.M., Y.C. and G.C.; methodology, K.M. and Y.C.; software, K.M., G.C. and Y.V.; validation, K.M., Y.V. and P.M.; formal analysis, Y.C., G.C. and P.M.; investigation, G.C., Y.V. and P.M.; resources, Y.C.; data curation, Y.V. and P.M.; writing—original draft preparation, K.M., Y.C. and Y.V.; writing—review and editing, G.C. and P.M.; visualization, K.M.; supervision, G.C.; project administration, P.M.; funding acquisition, P.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was co-financed by the European Union and Greek national funds through the Competitiveness, Entrepreneurship and Innovation Operational Programme, under the Call “Research-Create-Innovate”; project title: “Development of technologies and methods for cultural inventory data interoperability—ANTIKLEIA”; project code: T1EDK-01728; MIS code: 5030954.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Rahman, M.A.; Asyhari, A.T. The emergence of Internet of Things (IoT): Connecting anything, anywhere. *Computers* **2019**, *8*, 40. [\[CrossRef\]](#)
2. Sundmaeker, H.; Guillemin, P.; Friess, P.; Woelfflé, S. Vision and challenges for realising the Internet of Things. *Clust. Eur. Res. Proj. Internet Things Eur. Commission* **2010**, *3*, 34–36.
3. Atzori, L.; Iera, A.; Morabito, G. The internet of things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [\[CrossRef\]](#)
4. Dey, A.K.; Abowd, G.D.; Salber, D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Hum. Comput. Interact.* **2001**, *16*, 97–166. [\[CrossRef\]](#)
5. Abowd, G.D.; Dey, A.K.; Brown, P.J.; Davies, N.; Smith, M.; Steggle, P. Towards a better understanding of context and context-awareness. In Proceedings of the International Symposium on Handheld and Ubiquitous Computing, Karlsruhe, Germany, 27–29 September 1999; pp. 304–307.
6. Perera, C.; Zaslavsky, A.; Christen, P.; Georgakopoulos, D. Context aware computing for the internet of things: A survey. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 414–454. [\[CrossRef\]](#)
7. Razzaque, M.A.; Milojevic-Jevric, M.; Palade, A.; Clarke, S. Middleware for internet of things: A survey. *IEEE Internet Things J.* **2015**, *3*, 70–95. [\[CrossRef\]](#)
8. Henriksen, K.; Indulska, J.; Rakotonirainy, A. Modeling context information in pervasive computing systems. In Proceedings of the International Conference on Pervasive Computing, Zurich, Switzerland, 26–28 August 2002; pp. 167–180.
9. Bettini, C.; Brdiczka, O.; Henriksen, K.; Indulska, J.; Nicklas, D.; Ranganathan, A.; Riboni, D. A survey of context modelling and reasoning techniques. *Pervasive Mob. Comput.* **2010**, *6*, 161–180. [\[CrossRef\]](#)
10. Jara, A.J.; Olivieri, A.C.; Bocchi, Y.; Jung, M.; Kastner, W.; Skarmeta, A.F. Semantic web of things: An analysis of the application semantics for the iot moving towards the iot convergence. *Int. J. Web Grid Serv.* **2014**, *10*, 244–272. [\[CrossRef\]](#)
11. Henriksen, K.; Indulska, J. Modelling and using imperfect context information. In Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, Orlando, FL, USA, 14–17 March 2004; pp. 33–37.
12. Pradeep, P.; Krishnamoorthy, S. The MOM of context-aware systems: A survey. *Comput. Commun.* **2019**, *137*, 44–69. [\[CrossRef\]](#)
13. Li, X.; Eckert, M.; Martinez, J.F.; Rubio, G. Context aware middleware architectures: Survey and challenges. *Sensors* **2015**, *15*, 20570–20607. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Machado, R.S.; Almeida, R.B.; Pernas, A.M.; Yamin, A.C. State of the art in hybrid strategies for context reasoning: A systematic literature review. *Inf. Softw. Technol.* **2019**, *111*, 122–130. [\[CrossRef\]](#)
15. Park, H.S.; Oh, K.; Cho, S.B. Bayesian network-based high-level context recognition for mobile context sharing in cyber-physical system. *Int. J. Distrib. Sens. Netw.* **2011**, *7*, 650387. [\[CrossRef\]](#)
16. Li, X.; Martínez, J.F.; Rubio, G. Towards a hybrid approach to context reasoning for underwater robots. *Appl. Sci.* **2017**, *7*, 183. [\[CrossRef\]](#)
17. Paridel, K.; Bainomugisha, E.; Vanrompay, Y.; Berbers, Y.; De Meuter, W. Middleware for the internet of things, design goals and challenges. *Electron. Commun. EASST* **2010**, *28*. [\[CrossRef\]](#)
18. Chianese, A.; Piccialli, F.; Valente, I. Smart environments and cultural heritage: A novel approach to create intelligent cultural spaces. *J. Locat. Based Serv.* **2015**, *9*, 209–234. [\[CrossRef\]](#)
19. Petronela, T. The importance of the intangible cultural heritage in the economy. *Procedia Econ. Financ.* **2016**, *39*, 731–736. [\[CrossRef\]](#)

20. Ardissono, L.; Kuflik, T.; Petrelli, D. Personalization in cultural heritage: The road travelled and the one ahead. *User Model. User Adapt. Interact.* **2012**, *22*, 73–99. [[CrossRef](#)]
21. Fidas, C.A.; Avouris, N.M. Personalization of mobile applications in cultural heritage environments. In Proceedings of the 2015 6th International Conference on Information, Intelligence, Systems and Applications (IISA), Corfu, Greece, 6–8 July 2015; pp. 1–6.
22. Amato, F.; Moscato, V.; Picariello, A.; Colace, F.; Santo, M.D.; Schreiber, F.A.; Tanca, L. Big data meets digital cultural heritage: Design and implementation of scrabs, a smart context-aware browsing assistant for cultural environments. *J. Comput. Cult. Herit.* **2017**, *10*, 1–23. [[CrossRef](#)]
23. Konstantakis, M.; Aliprantis, J.; Michalakakis, K.; Caridakis, G. Recommending user experiences based on extracted cultural personas for mobile applications-REPEAT methodology. In Proceedings of the MobileCH@ Mobile HCI, Barcelona, Spain, 3–6 September 2018.
24. Shah, N.F.M.N.; Ghazali, M. A systematic review on digital technology for enhancing user experience in museums. In Proceedings of the International Conference on User Science and Engineering, Puchong, Malaysia, 28–30 August 2018; pp. 35–46.
25. Piccialli, F.; Chianese, A. A location-based IoT platform supporting the cultural heritage domain. *Concurr. Comput. Pract. Exp.* **2017**, *29*, e4091. [[CrossRef](#)]
26. Pavlidis, G. Recommender systems, cultural heritage applications, and the way forward. *J. Cult. Herit.* **2019**, *35*, 183–196. [[CrossRef](#)]
27. Not, E.; Petrelli, D. Blending customisation, context-awareness and adaptivity for personalised tangible interaction in cultural heritage. *Int. J. Hum. Comput. Stud.* **2018**, *114*, 3–19. [[CrossRef](#)]
28. Michalakakis, K.; Moraitou, E.; Aliprantis, J.; Caridakis, G. Semantic Representation and Internet of Things in Cultural Heritage Preventive Conservation. In Proceedings of the International Conference on Cultural Informatics, Communication & Media Studies, Mytilene, Greece, 13–15 June 2019; Volume 1.
29. Schilit, B.; Adams, N.; Want, R. Context-aware computing applications. In Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, USA, 8–9 December 1994; pp. 85–90.
30. Abowd, G.D.; Mynatt, E.D. Charting past, present, and future research in ubiquitous computing. *ACM Trans. Comput. Hum. Interact.* **2000**, *7*, 29–58. [[CrossRef](#)]
31. Benerecetti, M.; Bouquet, P.; Bonifacio, M. Distributed context-aware systems. *Hum. Comput. Interact.* **2001**, *16*, 213–228. [[CrossRef](#)]
32. Hofer, T.; Schwinger, W.; Pichler, M.; Leonhartsberger, G.; Altmann, J.; Retschitzegger, W. Context-awareness on mobile devices—the hydrogen approach. In Proceedings of the 36th annual Hawaii International Conference on System Sciences, Big Island, HI, USA, 6–9 January 2003; p. 10.
33. Henricksen, K. A Framework for Context-Aware Pervasive Computing Applications. Ph.D. Thesis, School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, Australia, 2003.
34. Gu, T.; Pung, H.K.; Zhang, D.Q. A service-oriented middleware for building context-aware services. *J. Netw. Comput. Appl.* **2005**, *28*, 1–18. [[CrossRef](#)]
35. Mei, L.; Easterbrook, S. Capturing and modeling human cognition for context-aware software. In Proceedings of the International Conference for Research on Computational Models and Computation-Based Theories of Human Behavior, Manchester, UK, 24–26 July 2009.
36. Rizou, S.; Häussermann, K.; Dürr, F.; Cipriani, N.; Rothermel, K. A system for distributed context reasoning. In Proceedings of the 2010 Sixth International Conference on Autonomic and Autonomous Systems, Cancun, Mexico, 7–13 March 2010; pp. 84–89.
37. Krishnamoorthy, S.; Bhargava, P.; Mah, M.; Agrawala, A. Representing and managing the context of a situation. *Comput. J.* **2012**, *55*, 1005–1019. [[CrossRef](#)]
38. Zhang, D.; Huang, H.; Lai, C.F.; Liang, X.; Zou, Q.; Guo, M. Survey on context-awareness in ubiquitous media. *Multimed. Tools Appl.* **2013**, *67*, 179–211. [[CrossRef](#)]
39. Sezer, O.B.; Dogdu, E.; Ozbayoglu, A.M. Context-aware computing, learning, and big data in internet of things: A survey. *IEEE Internet Things J.* **2017**, *5*, 1–27. [[CrossRef](#)]
40. Samulowitz, M.; Michahelles, F.; Linnhoff-Popien, C. Capeus: An architecture for context-aware selection and execution of services. In Proceedings of the IFIP International Conference on Distributed Applications and Interoperable Systems, Kraków, Poland, 17–19 September 2001; pp. 23–39.
41. Wang, X.H.; Zhang, D.Q.; Gu, T.; Pung, H.K. Ontology based context modeling and reasoning using OWL. In Proceedings of the IEEE Annual Conference on Pervasive Computing and Communications Workshops, Orlando, FL, USA, 14–17 March 2004; pp. 18–22.
42. Sheng, Q.Z.; Benatallah, B. ContextUML: A UML-based modeling language for model-driven development of context-aware web services. In Proceedings of the International Conference on Mobile Business (ICMB'05), Sydney, Australia, 11–13 July 2005; pp. 206–212.
43. Agostini, A.; Bettini, C.; Riboni, D. Hybrid reasoning in the CARE middleware for context awareness. *Int. J. Web Eng. Technol.* **2009**, *5*, 3–23. [[CrossRef](#)]
44. Celdrán, A.H.; Clemente, F.J.G.; Pérez, M.G.; Pérez, G.M. SeCoMan: A semantic-aware policy framework for developing privacy-preserving and context-aware smart applications. *IEEE Syst. J.* **2014**, *10*, 1111–1124. [[CrossRef](#)]
45. Mcheick, H. Modeling context aware features for pervasive computing. *Procedia Comput. Sci.* **2014**, *37*, 135–142. [[CrossRef](#)]

46. Hoda, M.; Montaghani, V.; Al Osman, H.; El Saddik, A. ECOPPA: Extensible Context Ontology for Persuasive Physical-Activity Applications. In Proceedings of the International Conference on Information Technology & Systems, Libertad City, Ecuador, 10–12 January 2018; pp. 309–318.
47. Aguilar, J.; Jerez, M.; Rodríguez, T. CAMEnto: Context awareness meta ontology modeling. *Appl. Comput. Inform.* **2018**, *14*, 202–213. [\[CrossRef\]](#)
48. Lu, Z.-J.; Li, G.-Y.; Pan, Y. A method of meta-context ontology modeling and uncertainty reasoning in swot. In Proceedings of the 2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Chengdu, China, 13–15 October 2016; pp. 128–135.
49. Angsuchotmetee, C.; Chbeir, R.; Cardinale, Y. MSSN-Onto: An ontology-based approach for flexible event processing in Multimedia Sensor Networks. *Future Gener. Comput. Syst.* **2020**, *108*, 1140–1158. [\[CrossRef\]](#)
50. Mansour, E.; Chbeir, R.; Arnould, P. HSSN: An ontology for hybrid semantic sensor networks. In Proceedings of the 23rd International Database Applications & Engineering Symposium, Athens Greece, 10–12 June 2019; pp. 1–10.
51. Wongpatikaseree, K.; Ikeda, M.; Buranarach, M.; Supnithi, T.; Lim, A.O.; Tan, Y. Activity recognition using context-aware infrastructure ontology in smart home domain. In Proceedings of the 2012 Seventh International Conference on Knowledge, Information and Creativity Support Systems, Melbourne, Australia, 8–10 November 2012; pp. 50–57.
52. Hoque, M.R.; Kabir, M.H.; Thapa, K.; Yang, S.H. Ontology-based context modeling to facilitate reasoning in a context-aware system: A case study for the smart home. *Int. J. Smart Home* **2015**, *9*, 151–156. [\[CrossRef\]](#)
53. Meditskos, G.; Kompatsiaris, I. iKnow: Ontology-driven situational awareness for the recognition of activities of daily living. *Pervasive Mob. Comput.* **2017**, *40*, 17–41. [\[CrossRef\]](#)
54. Maarala, A.I.; Su, X.; Riekk, J. Semantic reasoning for context-aware Internet of Things applications. *IEEE Internet Things J.* **2016**, *4*, 461–473. [\[CrossRef\]](#)
55. Riboni, D.; Sztyler, T.; Civitarese, G.; Stuckenschmidt, H. Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, Heidelberg, Germany, 12–16 September 2016; pp. 1–12.
56. Zhang, D.; Cao, J.; Zhou, J.; Guo, M. Extended Dempster-Shafer theory in context reasoning for ubiquitous computing environments. In Proceedings of the 2009 International Conference on Computational Science and Engineering, Vancouver, BC, Canada, 29–31 August 2009; Volume 2, pp. 205–212.
57. Sanchez, D.; Tentori, M.; Favela, J. Hidden Markov models for activity recognition in ambient intelligence environments. In Proceedings of the Eighth Mexican International Conference on Current Trends in Computer Science (ENC 2007), Morelia, Mexico, 24–28 September 2007; pp. 33–40.
58. Skillen, K.L.; Chen, L.; Nugent, C.D.; Donnelly, M.P.; Burns, W.; Solheim, I. Ontological user modelling and semantic rule-based reasoning for personalisation of Help-On-Demand services in pervasive environments. *Future Gener. Comput. Syst.* **2014**, *34*, 97–109. [\[CrossRef\]](#)
59. Nalepa, G.J.; Bobek, S. Rule-based solution for context-aware reasoning on mobile devices. *Comput. Sci. Inf. Syst.* **2014**, *11*, 171–193. [\[CrossRef\]](#)
60. Bahramian, Z.; Ali Abbaspour, R.; Claramunt, C. A cold start context-aware recommender system for tour planning using artificial neural network and case based reasoning. *Mob. Inf. Syst.* **2017**, *2017*, 9364903. [\[CrossRef\]](#)
61. Lin, T.N.; Lin, P.C. Performance comparison of indoor positioning techniques based on location fingerprinting in wireless networks. In Proceedings of the 2005 International Conference on Wireless Networks, Communications and Mobile Computing, Maui, HI, USA, 13–16 June 2005; Volume 2, pp. 1569–1574.
62. Strobbe, M.; Van Laere, O.; Dhoedt, B.; De Turck, F.; Demeester, P. Hybrid reasoning technique for improving context-aware applications. *Knowl. Inf. Syst.* **2012**, *31*, 581–616. [\[CrossRef\]](#)
63. Román, M.; Hess, C.; Cerqueira, R.; Ranganathan, A.; Campbell, R.H.; Nahrstedt, K. A middleware infrastructure for active spaces. *IEEE Pervasive Comput.* **2002**, *1*, 74–83. [\[CrossRef\]](#)
64. Capra, L. Context-aware reflective middleware system for mobile applications/Licia Capra, Wolfgang Emmerich, Cecilia Mascolo. *IEEE Trans. Softw. Eng.* **2003**, *29*, 929–945. [\[CrossRef\]](#)
65. Chen, H. An Intelligent Broker Architecture for Pervasive Context-Aware Systems. Ph.D. Thesis, University of Maryland, College Park, MD, USA, 2004.
66. Eisenhauer, M.; Rosengren, P.; Antolin, P. A development platform for integrating wireless devices and sensors into ambient intelligence systems. In Proceedings of the 2009 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, Rome, Italy, 22–26 June 2009; pp. 1–3.
67. Perera, C.; Zaslavsky, A.; Christen, P.; Georgakopoulos, D. Ca4iot: Context awareness for internet of things. In Proceedings of the 2012 IEEE International Conference on Green Computing and Communications, Besancon, France, 20–23 November 2012; pp. 775–782.
68. Zhang, H.; Huang, T.; Liu, Y.; Zhu, S.; Gui, G.; Chi, Y. Senz: A Context Awareness Middleware System Used in Mobile Devices. In Proceedings of the 2017 IEEE 85th Vehicular Technology Conference (VTC Spring), Sydney, Australia, 4–7 June 2017; pp. 1–7.
69. Machado, R.; Rosa, F.; Almeida, R.; Primo, T.; Pilla, M.; Pernas, A.; Yamin, A. A hybrid architecture to enrich context awareness through data correlation. In Proceedings of the 33rd Annual ACM Symposium on Applied Computing, Pau, France, 9–13 April 2018; pp. 1451–1453.

70. Kim, S.; Yoon, Y.I. Ambient intelligence middleware architecture based on awareness-cognition framework. *J. Ambient. Intell. Humaniz. Comput.* **2018**, *9*, 1131–1139. [[CrossRef](#)]
71. Belcastro, L.; Marozzo, F.; Trunfio, P. A scalable middleware for context-aware mobile applications. *Int. J. Hoc Ubiquitous Comput.* **2019**, *31*, 112–122. [[CrossRef](#)]
72. Altı, A.; Lakehal, A.; Laborie, S.; Roose, P. Autonomic semantic-based context-aware platform for mobile applications in pervasive environments. *Future Internet* **2016**, *8*, 48. [[CrossRef](#)]
73. Chianese, A.; Marulli, F.; Moscato, V.; Piccialli, F. SmARTweet: A location-based smart application for exhibits and museums. In Proceedings of the 2013 International Conference on Signal-Image Technology & Internet-Based Systems, Kyoto, Japan, 2–5 December 2013; pp. 408–415.
74. Dossis, M.; Kazanidis, I.; Valsamidis, S.; Kokkonis, G.; Kontogiannis, S. Proposed open source framework for interactive IoT smart museums. In Proceedings of the 22nd Pan-Hellenic Conference on Informatics, Athens, Greece, 29 November–1 December 2018; pp. 294–299.
75. Alletto, S.; Cucchiara, R.; Del Fiore, G.; Mainetti, L.; Mighali, V.; Patrono, L.; Serra, G. An indoor location-aware system for an IoT-based smart museum. *IEEE Internet Things J.* **2015**, *3*, 244–253. [[CrossRef](#)]
76. Bartolini, I.; Moscato, V.; Pensa, R.G.; Penta, A.; Picariello, A.; Sansone, C.; Sapino, M.L. Recommending multimedia visiting paths in cultural heritage applications. *Multimed. Tools Appl.* **2016**, *75*, 3813–3842. [[CrossRef](#)]
77. Henriksen, K.; Livingstone, S.; Indulska, J. Towards a hybrid approach to context modelling, reasoning and interoperation. In Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management, in Conjunction with Ubicomp, Nottingham, UK, 7 September 2004; Volume 2004.
78. Hariri, R.H.; Fredericks, E.M.; Bowers, K.M. Uncertainty in big data analytics: Survey, opportunities, and challenges. *J. Big Data* **2019**, *6*, 1–16. [[CrossRef](#)]
79. Haghghi, P.D.; Krishnaswamy, S.; Zaslavsky, A.; Gaber, M.M. Reasoning about context in uncertain pervasive computing environments. In Proceedings of the European Conference on Smart Sensing and Context, Zurich, Switzerland, 29–31 October 2008; pp. 112–125.
80. Soufi, M.D.; Samad-Soltani, T.; Vahdati, S.S.; Rezaei-Hachesu, P. Decision support system for triage management: A hybrid approach using rule-based reasoning and fuzzy logic. *Int. J. Med. Inform.* **2018**, *114*, 35–44. [[CrossRef](#)]
81. Horrocks, I.; Li, L.; Turi, D.; Bechhofer, S. The instance store: DL reasoning with large numbers of individuals. In Proceedings of the 2004 Description Logic Workshop (DL 2004), Whistler, BC, Canada, 6–8 June 2004; pp. 31–40.
82. Abidin, N.Z.; Ismail, A.R.; Emran, N.A. Performance analysis of machine learning algorithms for missing value imputation. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 442–447. [[CrossRef](#)]
83. Ayadi, A.; Ghorbel, O.; Obeid, A.M.; Abid, M. Outlier detection approaches for wireless sensor networks: A survey. *Comput. Netw.* **2017**, *129*, 319–333. [[CrossRef](#)]