A Defeasible Logic-based Framework for Contextualizing Deployed Applications

Noor Sami Al-Anbaki[1], Nadim Obeid[2], Khair Eddin Sabri[3]
King Abdullah II School for Information Technology
University of Jordan, Amman, Jordan

***Abstract*—In human to human communication, context increases the ability to convey ideas. However, in human to application and application to application communication, this property is difficult to attain. Context-awareness becomes an emergent need to achieve the goal of delivering more user-centric personalized services, especially in ubiquitous environments. However, there is no agreed-upon generic framework that can be reused by deployed applications to support context-awareness. In this paper, a defeasible logic-based framework for context-awareness is proposed that can enhance the functionality of any deployed application. The nonmonotonic nature of defeasible logic has the capability of attaining justifiable decisions in dynamic environments. Classical defeasible logic is extended by meta-rules to increase its expressiveness power, facilitate its representation of complex multi-context systems, and permit distributed reasoning. The framework is able to produce justified decisions depending on both the basic functionality of the system that is itself promoted by contextual knowledge and any cross-cutting concerns that might be added by different authorities or due to further improvements to the system. Active concerns that are triggered at certain contexts are encapsulated in separate defeasible theories. A proof theory is defined along with a study of its formal properties. The framework is applied to a motivating scenario to approve its feasibility and the conclusions are analyzed using argumentation as an approach of reasoning.**

***Keywords*—Context-awareness; nonmonotonicity; defeasible logic; distributed reasoning; argumentation**

I. Introduction

It is fair to say that the ubiquitous computing paradigm revolutionized our understanding of computing and what it can deliver. It merges computer devices and sensors in an integrated environment, to provide better communication and enhanced accessibility to information sources. The final objective is to provide users with services available whenever, however, and wherever needed [1]. Applications should be intelligent enough to handle the mobility of users and resources themselves as well as the ever-changing context in a seamless manner with minimum human intervention. In other words, applications should be context-aware.

The term Context-Aware Computing was first introduced in 1994 [2], this study focused on the communication aspects related to broadcasting information from a server to its clients. Context was considered to be the information related to the location of users and other objects in the system and how this information changes over time, in addition to the communication overload. In [3], context awareness role in mobile computing was discussed, the study considered context to be the identity of the user, nearby users, location, time and season. Other studies that discussed what context could be can be found in [4] [5] [6].

In 2001, Dey [7] introduced the most well-known definition of context: "Any information that can be used to characterize 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". This definition was a milestone in the growth of the notion of context as it is generic, operational and exceeded the boundary of (time, location and user's identity) where context was always defined accordingly. On the other hand, Context-Awareness is considered to be the ability of the system to sense (gather information) about its surrounding physical and operational environment at any given time, perceive and adapt behavior accordingly [8].

A context-aware system should support mechanisms for collecting contextual information, representation, reasoning and application [9]. Contextual information is domain-dependent, it can be any piece of information that describes the entity involved in the interaction, it could be time, location, task, identity, etc. or a group of them. The acquisition of this information is beyond the scope of this work, it is achieved using different technologies. The emphasis of this work is on the two most important phases in any framework that supports context awareness: representation and reasoning.

In this paper, a generic framework is present that can guide the contextualizing process of deployed applications. The framework provides a powerful mechanism to represent multi-context distributed systems and permits distributed reasoning. An extension to defeasible logic theory was proposed by adding the notion of meta-rules that are able to reason over theories; this enhancement would open the door of new usage of DL in the representation and reasoning of complex systems.

The significance of the study lies in its conceptual analysis of context by considering it to be both, information that can characterize entities and information that has the ability to characterize a whole new behavior of the system.

Another advancement of the framework is that it permits distributed reasoning which is a challenging area in AI, as there is no central authority to control the context flow in the overall system, but rather each component in the system is allowed to add its own view of manipulating contextual knowledge. This is achieved using a separation of concerns principle and can highly increase users' and administrators' satisfaction.

The work is of both theoretical and empirical significance to the research in context awareness and contextual reasoning. The theoretical importance lies in the proposed extension to the defeasible theory that permits the representation of complex multi-context systems and facilitates distributed reasoning, while empirical significance lies in the ability to employ the framework to contextualize any kind of application. It allows the developers of context-aware applications to easily represent and manage different behaviors of the application in different contexts.

This paper is organized as follows: Section 2 highlights some issues in contextual reasoning. Section 3 presents related work. Section 4 presents the defeasible logic. An illustrative scenario is presented in Section 5. Section 6 discusses our interpretation of context and context-awareness. Section 7 presents the proposed framework of context-awareness. Section 8 defines the formal proofs of the framework. An implementation of the illustrative case study in the proposed framework is presented in Section 9 along with its analysis. A brief discussion is presented in Section 10 and finally, Section 11 covers the conclusions and future work.

II. Some Issues in Contextual Reasoning

There are many alternatives in the literature that deal with knowledge representation and reasoning issues [10] [11] [12] [13] [15] [14] [16] [17] [18]. However, when this knowledge is characterized as contextual knowledge (i.e. "as information that can be used to characterize the situation of an entity"), there are extra properties that need special treatment.

- First of all, context is domain-dependent (e.g. the identity of a user plays a subtle role in an access control system, but it is not important in a supermarket billing system). This is considered an appealing property that helps to develop personalized services.

- Second, context is a conflict-sensitive concept, i.e. multiple sources of contextual information might lead to infer conflicting decisions. This happens due to multiple sources of contextual information which lead to ambiguity. The study in [19] highlighted other problems related to contextual information in that they might be unknown, imprecise, and erroneous.

- Third, when reasoning is employed, context becomes nested. In complex systems, the context of an entity is not merely restricted to basic contextual attributes that are collected directly from sensors (e.g. the temperature of a room) but rather, it refers to complex contextual attributes that are inferred from basic contextual attributes. For example, if the temperature of the room is between (72 F and 76 F), the room warmth is comfortable), in this way, a room with a temperature degree (74 F) is characterized by two contextual attributes, its temperature is (74 F) and its warmth is comfortable. This different level of abstraction gives context an operational power, such that a basic contextual attribute may lead to a whole new behavior and direct the characterization of many other aspects in the system e.g. a room's temperature may affect not only the degree of relief in the room but rather may play a role in deciding the placement of certain assets in the room e.g. a server, or turning on the air conditioning which is, in turn, affects the energy consumption, and so on.

These characteristics lead to challenges that cannot be avoided especially in complex systems that operate in ubiquitous environments where the system contains multiple entities and the process integration spans organizations where interactive entities in the system may belong to different authorities and each works under different regulations. The system should be able to reason and reach justifiable decisions regardless of these complications.

To handle these issues, a solid representation mechanism should be employed that can deal with ambiguity and a concrete conflict resolution mechanism that enables inferring justifiable non-conflicted decisions. McCarthy [20] was one of the first scientists that point out the issue of contextual reasoning. He suggested that the combination of nonmonotonic reasoning and context formalism would constitute an adequate solution to overcome the problems associated with including contextual information in the decision-making process. Nonmontonicity provides mechanisms that allow the system to reason and reach justifiable decisions by retracting conclusions that turned out to be incorrect and derive new, better-justified conclusions instead [21]. This makes it very suitable to tackle the reasoning process in dynamic situations with incomplete/changing information.

Defeasible logic (DL) [22] is a well-known skeptical nonmonotonic logic that can be used in dynamic environments due to its characteristics: it is expressive, natural, not ambiguous and programmable. It has attracted many researchers to incorporate it in different application domains such as modeling of contracts [23], legal reasoning [24], modeling social agents [25], modeling social commitments [15] [17] [18], etc. The most significant feature of DL is that it preserves the consistency of the system regardless of conflicts because it does not produce contradictory conclusions. When a conflict occurs, conflicting rules do not arouse. It supports the use of priorities to resolve these conflicts to allow the system inferring with incomplete/partial information.

III. Related Studies

There are many attempts in the literature to formalize context in order to be able to reason based on its attributes along with its accompanied obstacles that might lead to conflicts in the decision-making process.

As the issue of context sensing and integration in highly connected to the technical infrastructure of the system, most of the researches that aimed to define generic frameworks for context awareness, pointed out the architecture aspects of the framework, e.g. the authors in [26] proposed a context management framework that enables the collaboration of multiple domains by exchanging contextual information. Their framework highlighted the architectural issues; it is based on a peer-to-peer architecture. The framework imposes a hierarchic ordering of context sources and multiple reasoning tools. This facilitates adaptability as new context sources and reasoning techniques can be added. The most important parts of the framework are the uniform interface where all of the context-provides are attached to a reasoner where all the reasoning methods can be employed.

Other studies presented techniques to deal with contextual information, e.g. [27] defined a Context Toolkit that provided an infrastructure for prototyping context-aware applications. However, it didn't provide a mechanism to reason about contexts. There is no formal tool to write reasoning rules for contexts or to infer higher-level contexts decisions.

Formal representation of context can be found in [28], where an architecture and programming framework for triggering application adaptation to changes in context was proposed. It employed basic (if-then rules) to formalize the behavior of an application in different contexts. In [29], first-order logic was used to describe contextual information and reasoning was done using Boolean operators and existential and universal quantifiers.

Recent trends in context-awareness pointed out the significance of generic frameworks in manipulating context flow in smart environments.

A formal representation of context can be found in [30] where the authors used ontologies to model information gathered from IoT devices in a smart home environment and used Description logic [31] to deduce activities depending on the gathered contextual attributes from the devices.

Another study [32] proposed a context-aware framework for multi-agent environment. Agents in their framework extract contextual information from ontologies; in fact, an agent can extract its rules and facts from one or more ontologies. Each agent performs reasoning based on the collected information and communicates with other agent(s) using bridge rules; the concluded decision is used to adapt the system behavior. The framework is used to generate preference sets for users, which is a set of active rules for each user.

Defeasible Logic DL [16] [22] had approved to be one of the famous logic tools that are successful to characterize contextual reasoning; it has a nonmonotonic relation between the premises and their consequences which is an effective way of formalizing the dynamic nature of ubiquities computing. Several studies succeeded to build models that could reason in the shade of contextual information based on DL [19] [33] [34]. However, these studies handle context in an environment of operating agents, they consider context to be whatever local knowledge the agent has. This view is correct and it serves the goal of showing how collaborating agents can cooperate to achieve a specific goal regardless of the challenges caused by the imperfect nature of context.

These approaches can be viewed as enhanced versions of previous approaches that aim at solving the partial knowledge issues of autonomous agents by collaboration. This is achieved using bridge rules [34] and mapping rules [33]. None of these studies investigated the effect of context on the decision made by each agent/entity and how contextual information can affect the overall behavior of the system.

The proposed framework discusses how to enhance deployed applications using context. Rather than considering it to be raw agent's knowledge received from contributed sensing devices, a conceptual view of context is adopted, it considered it as a concern/goal that needs to be achieved, it is different than the models in the literature as it defines the boundary between what local knowledge the system is already designed to manipulate (i.e. what is the input information that system rules make decisions accordingly) and what is contextual knowledge that could be used to enhance the system operation. We argue that the integration of contextual information in the reasoning process of a system that is driven by many concerns can not only be achieved by adding additional attributes/predicates that describe contextual information and additional rules that manipulate them. The projection of contextual knowledge on the system affects both the nature of its base functionality (base concern) and the way it handles cross-cutting objectives, concerns or exceptions. This simulates how humans think. Humans' decisions are never static; they are always changing based on upcoming knowledge, i.e. current context. For example, a student might choose an academic major based on his/her interest (a basic aspect), in addition to the GPA, budget, family opinion, the need for the labor market at that time (a contextual aspect).

The framework is implemented using Defeasible Logic DL, it benefits from both the expressiveness power of logic in representing knowledge and the nonmonotonic feature of the defeasible theory that facilitates a smooth reasoning process in a dynamic environment.

Based on this representation of context, the framework can be viewed as a platform that can be used to augment ubiquitous applications with context awareness by employing a conceptual view of context that is able to infer high-level decisions. The framework allows easy integration of different modes of operations triggered by different contexts and at the same time preserves the consistency of the decisions made by the system.

IV. Defeasible Logic

Defeasible logic (DL) was proposed by Nute in 2001 [22], unlike monotonic reasoning, it has a nonmonotonic relation between the premises and their logical consequences which made it suitable for reasoning in dynamic environments. In order to illustrate the nonmonotonic reasoning power, assume the situation of the following example that resembles the monotonic kind of reasoning.

Example 1: *Bob is often invited to social events by his friends. He usually attends these events; however, he has the following two preferences about going to a party.*

> *P_1: If the inviting person is one of his closest friends, he would go.*
>
> *P_2: He prefers not to go if Adam is invited.*
>
> *Bob was invited by his best friend, Julie, and she told him that Adam is invited as well.*

In a monotonic kind of reasoning, the two rules are applied and both of their consequences are valid (*go* and *don't go*) which leads to inconsistency, it is the system developer's responsibility to design rules that avoid such conflicts. Monotonic reasoning needs a lot of administrative effort and it neither scales well nor can be used in an environment with multiple administrative authorities. On the other hand, a nonmonotonic reasoning approach is founded on the ability to infer tentative conclusions that can be retracted based on new evidence [14].

Formally, DL can be seen as an extension to first-order predicate calculus FOPC [35], with the addition of the defeasible implication (⇒) that is used to infer the tentative conclusions, and the ambiguity-blocking priority relation (>) that is used to preserve the consistency of the system and infer justifiable conclusions in both static and dynamic domains.

Basically, a defeasible theory D (also called a knowledge base in DL) is a triple *(F, R, >)*, it consists of three main components:

1. **Facts (F)**: is a finite set of literals that represent indisputable statements.
2. **Rules (R)**: is a finite set of three types of rules ($R = R_s \cup R_d \cup R_f$) each rule comes in the form,

$$R: Ant(R) \bullet\!\!\rightarrow Conseq(R)$$

Where, (*R*) is a unique label, *(Ant(R))* is an antecedent, ($\bullet\!\!\rightarrow$) is a set of one-direction arrows that identify three types of implications ($\rightarrow$ to denote strict rules, $\Rightarrow$ to denote defeasible rules and $\rightsquigarrow$ to denoted defeaters) and a *(Conseq(R))* is the head/consequence which is the conclusion of the rule. R[B] means the set of rules in R with consequence B.

A) *Strict rules* R_s: is a set of rules that cannot be defeated, e.g. " if a country is on the equator, it would be very hot during summer",

$R_1: equatorial(X) \wedge during\_summer(X) \rightarrow very\_hot(X),$

B) *Defeasible rules* R_d: is a set of rules that can be defeated by contrary evidence, e.g. " if a country is on the coast, it is usually very hot during summer",

$R_2: coastal(X) \wedge during\_summer(X) \Rightarrow very\_hot(X),$

Rule R_2 indicates that during summer, coastal counties weather is very hot unless there is other evidence suggesting a contrary result, such as (R_3) which states that "if it rains in summer, the weather would not be very hot"

$R_3: raining\_at(X) \wedge during\_summer(X) \Rightarrow \neg very\_hot(X)$

C) *Defeaters* R_f: rules presented by ($\rightsquigarrow$), they do not use to conclude but rather to prevent deriving conclusions of some defeasible rules by producing evidence to the contrary e.g.

$R_2': coastal(X) \wedge during\_spring(X) \rightsquigarrow \neg very\_hot(X),$

3. **The superiority relation** >: is a binary relation over the set of defeasible rules R_d i.e. ($> \subseteq R_d \times R_d$). It is defined externally and statically to resolve conflicts. For example, given that defeasible rules R_2 and R_3 are both approved, no conclusive decision can be made about whether the weather is very hot or not. But, if the superiority relationship ($R_3 > R_2$) is introduced, then R_3 overrides R_2 and it can be concluded that the weather is not very hot while it is raining even during the summer season. The superiority relation > is acyclic, that is, the transitive closure of > is irreflexive.

The interaction of these three components permits the conclusion of justifiable decisions. This is referred to in Fig. 1.

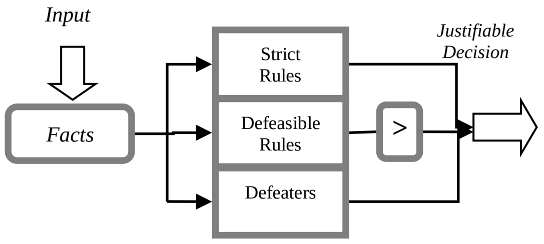

Fig. 1. Classical Defeasible Logic.

V. Illustrative Scenario

In this section, a motivating scenario from a ubiquitous environment is presented in order to illustrate the challenges of the domain and the capabilities of the proposed framework in reasoning in such environments.

Assume a situation where a smart application for lecturers' and employees' phone calls management inside a university campus was installed on all lecturers' and employees' mobile phones. This application manages the calls during the time when the lecturer is giving an online course.

The system was originally designed after an anti-disturbance base concern; it filters out calls based on the *identity* and *location* of the caller. It contains three rules that reason about two contextual attributes: (1) the identity of the caller that is identified by either: a) the caller being in the urgent list e.g. the dean's secretary b) the identity is unknown i.e. the number cannot be mapped to any of the names in the phone database, and (2) the location where the call is issued, it can be either a local or international call. The system makes its decision according to the following three rules:

- If the call is issued by a person on the urgent list, the phone rings.
- If the call is an international call, the phone rings.
- If the caller was unknown, the phone wouldn't ring.

To resolve the conflicts that might occur due to characterizing the caller as (being in the urgent list, international, and unknown); the user has to set priorities to decide which argument to support if more than one attribute hold. Sami set that if the call is international, the phone would ring even if the caller is unknown.

To further enhance the capabilities of the system using context awareness, the users of the application opted to personalize the service by formulating their own preferences. The application was attached to three different context providers that their knowledge can be used to better personalize the functionality of the system: a location detection service of the lecturer, schedule and the number of students engaged with the professor in the online session. A system user can set his/her own preferences based on the three available contextual attributes (location, schedule, and status that could be either busy or not busy depending on the number of students that are active during the online session). Preferences are activated upon turning on a flag of interest on the user's mobile phone. For example, Professor Sami has the following rules:

- If he is located inside Samsung-Lab, the phone wouldn't ring.
- If there is a scheduled lecture, the phone wouldn't ring.
- If he is engaged with less than five students in an online session, he is not busy and the phone could ring. This rule overrules the first two rules.

Suppose the situation when the dean asked the secretary Linda to call Professor Sami. Linda's number is in the urgent list; according to the anti-disturbance system rules, the phone should ring. However, Sami is inside Samsung Lab and is in an active session with five students, the phone should not ring.

From Linda's point of view, the phone should ring. She is sure that her number is already listed in the urgent list; however, she is not aware of Sami's preferences. The system is not able to decide which argument to support, the anti-disturbance concern argument or the users' preferences argument. Thus, an inter-concern conflict resolution mechanism is used to regulate the decision-making process.

As the end goal is to deliver personalized context-aware service, the designer sets that the decision inferred by users' preferences overrules the base system decision. In this arrangement, Sami won't be informed about the call.

One of the stakeholders, namely, the dean, was not satisfied with the services provided by this system, as his secretary uses the schedule of all professors and the administrative staff to determine the time of urgent meetings and she calls them based on this knowledge. However, according to the above settings, even though the user has no scheduled lecture at the time of the call but is inside the lab giving advice to some students on an online session, he/she was not informed of urgent meetings.

To resolve this issue, the system should address stakeholders' concerns such as urgent invitations. The system is connected to a meeting database that is controlled by several stakeholders, it saves the time, location, invitees of meetings, some of them are saved in prior e.g. a workshop and some of them are set up in an ad-hoc manner e.g. urgent meeting to discuss exams results. This concern manages the system as follows:

- If a person is invited, he/she should be informed.
- However, if the invitee has a scheduled lecture, he/she should not be informed.

The inter-concern conflict resolution mechanism should be carefully designed to represent the directions of the stakeholders as they represent a higher administrative authority. Such that the decision made according to the urgent invitations concern would be supported.

This simple scenario clarifies the challenges of using contextual knowledge in the decision-making process. In addition to the challenges of distributed reasoning in systems that encompass multiple authorities, each has its own preferences/regulations and its own interpretation of internal contextual knowledge. Each authority aims at making the decision referring to its own rules. This motivates the need to employ a distributed reasoning mechanism that can handle the production of justified and solid decisions in multi-context ubiquitous environments.

VI. Context and Context Awareness

Due to the enormous improvement of how computers, diffused sensors, and other devices collect situational/ contextual information, a lot of researchers tried to define context in several manners. Basically, context is identified by its attributes i.e. contextual information/variables that: (1) describe the user in an interaction with an application, the application/process, the environment and the interaction itself, (2) can be used to deliver more user-centric personalized services. The range of this information is quite vast and it depends on the domain itself, it could be time, location, number of users, the identity of the user, user's emotional states, the focus of attention, etc. [8] [9] [36] [37].

In order to build a framework that is able to enhance the operation of any application using contextual knowledge, it is very important to define the system's manipulated knowledge and enhancing contextual knowledge. Thus, throughout this work Dey's definition of context is extended to best suit this purpose: "For a deployed application, context is any information used to characterize the situation of an entity and can be sensed, collected and represented. This information is not part of the group of information that already describes that entity in the deployed application. 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".

In this work, the set of contextual information that represents domain knowledge C in an environment would be classified according to its presence in the system, as shown in Fig. 2:

1) Information that is collected from the environment in the digital form or can be presented digitally, collected context, ($C^o \subseteq C$) e.g. identity of the user, light, sound, location, size, etc.

2) Information that the system is designed to manipulate ($C^u \subseteq C^o$) e.g. in an access control model, the identity of the user and his role is used to determine what object(s) he/she can access.

3) Contextual information that can be added to enhance the functionality of the system ($C^h \subseteq C^o$) e.g. in an access control model in a dynamic environment, the time and location of the user requesting access is of major importance.

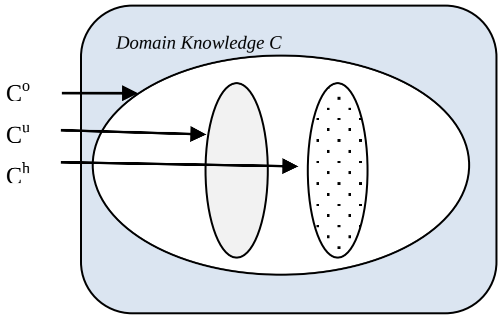

Fig. 2. Spectrum of Context.

A context of an entity is the net of all contextual attributes that describe that entity, the attributes might be physically collected by sensors, e.g. a GPS service can return the location of a person, LocatedIn(sami, home) to state that Sami is located at home (different schemes can be used to describe the location of an entity, e.g. XY coordinates) or logically constructed.

The proposed framework is built on defeasible logic; first-order literals are used to represent contextual attributes. Each literal represents a property that an entity holds or can be characterized by, it contains a name and a value e.g. location("university", near), role("mary", manger), connected_users(5) and so on. A literal is an atomic formula or its negation if α is an atomic formula; $\neg\alpha$ is its complement [38]. It should be defined in advance what is the meaning of the name and what is the range of values the literal may hold.

In the proposed framework, enhancing contextual attributes are referenced in four different ways:

1) According to the way they are gathered:

- Basic contextual attributes, to refer to the attributes that are collected directly from sensors, e.g. Humidity(basement, 40%), Temperature(basement, 65 F).
- Complex contextual attributes, to refer to the attributes that can be assembled as a result of logical operations on basic or other complex attributes, e.g. Humidity(basement, 40%) $\wedge$ Temperature(basement, 65 F) $\Rightarrow$ Comfortable(basement).
- Due to the multiple sources of contextual information, the decidability of complex attributes needs extra care, e.g. Noise(basement, 120 dB) $\Rightarrow$ ~Comfortable(basement).

In this case, any reasoning mechanism should uses priorities to resolve this issue.

2) According to the way they are manipulated:

- Internal contextual attributes, to refer to the attributes that are manipulated locally by the entity/administrative domain.
- External contextual attributes, to refer to the attributes that are manipulated outside the entity/administrative domain.

A context of a system that contains multiple collaborative sub-systems or components is a snapshot of the system's situation at a given point/interval of time. This encapsulates the external contextual attributes, internal contextual attributes of each sub-system and component in addition to the relationship between those subsystems and components.

A context-aware system is a system that is able to make a solid justified decision for every upcoming context. A contextualized deployed application is a system that is able to adapt its behavior in the shade of collected enhancing contextual attributes. In the proposed framework, the defeasible logic machinery would be employed for knowledge representation and reasoning and a concern-based model for context integration.

VII. Defeasible Logic Framework for Context Awareness

A deployed application, a base system, can be seen as a domain of knowledge that is governed by static rules that manipulate its local knowledge, i.e. (C^u); it is designed to serve a certain purpose. Augmenting such a system with context awareness can considerably improve its functionality by making it adaptable to the processing environment in order to provide a better experience to the user, better utilization of resources, etc. Contextual knowledge that is integrated into the system (C^h) can be embedded either implicitly or explicitly. In other words, it can be used to contextualize the base system's rules, if it is added by the same administrative authority and it serves the same purpose/concern of the base system, or it can be used to characterize other entities concerns that could be compatible or crosscutting to the base system's concern, they are triggered at some exceptional situations, normally this knowledge is perceived by other participating authorities or components in the system and it indicates their concerns from their own viewpoint. A context-aware system should be carefully designed to permit distributed reasoning and at the same time make justifiable decisions in spite of the fact that the distributed entities might have conflicting concerns.

The proposed context-aware framework based on defeasible logic is a theory L<G, β, D, λ>, that consists of the following components.

A. Triggers G

Triggers is a finite set of positive and negative ground literals that represents external basic contextual attributes acquired from the application domain. Triggers are imported from the system's global knowledge that is not necessarily known by the participating entities/users. They have a certain property is that they are issued by/collected from multiple participating sub-domains or different authorities in the application domain. It should be noted that not all contextual attributes can be used as triggers, a trigger's impact extends far beyond changing a single rule, yet, it can add/remove/change different rules and regulations in different components of the system e.g. an emergency situation that leads to a break glass procedure.

Formally, each trigger is an atomic formula. A valid framework can have no two complementary triggers i.e. an atomic formula and its negation. Triggers activate concerns using meta-rules. Meta-rules are rules that consequences are rules; they have been used in the literature as a powerful machine that facilitates reasoning about rules for contextualizing the provability of goals [34]. In the proposed framework their use would be extended; meta-rules are rules that consequences are defeasible theories.

Each trigger activates one concern using a defeasible meta-rule such that

G $=\{g_1,g_2,\ldots,g_n\}$ where, $n\geq 0$ is the number crosscutting concerns in the system

$M_i : g_i \Rightarrow D_i$

When a meta-rule contains an empty bode i.e. no antecedent, it denotes the activation of the base system,

$M_0 :\Rightarrow \beta$

For the illustrative scenario, the trigger that activates the preferences concern is the flag on the user's mobile phone.

B. *The Base System β*

The base system is the actual deployed application that is governed by rules that reflect obligations; these rules are put at the design phase to achieve a certain purpose or goal. In this framework β is represented using defeasible theory, it contains rules that reason about local attributes of the system (C^u) in order to serve a certain goal. When the need arises to integrate a new contextual knowledge in the decision-making process, the designer has two options, (1) If the newly added contextual knowledge is a simple attribute that does not crosscut the base concern of the system and is issued by the same administrative authority, it can be added implicitly to the base system, either as a new rule or as a predicate in an existing rule. (2) However, if the newly added contextual knowledge serves a concern that crosscuts the base system or is issued by a different administrative authority, it will be encapsulated as a distinct concern that is formalized.

Formally, the base system is a defeasible theory denoted as $\beta(F^{\beta}, R^{\beta}, >^{\beta})$, The formal definition of β flows naturally from the definition of classical defeasible theory, however, the components of the base system theory would be superscripted with the base system name β.

C. *Distributed Contextual Concerns Theories D*

Based on the separation of concerns principle, when the collected contextual knowledge refers to a cross-cutting concern or is issued by a different administrative authority, it will be encapsulated in a distinct theory(s). This would considerably enhance the development, maintenance, and security of the overall system and can enable reaching justifiable decisions even if only partial knowledge is available.

A concern refers to the context of participating entities/authorities regarding the service provided by the base system; it reflects their interpretation of the service based on their own manipulating of internal contextual knowledge that they can access. For example, suppose an energy-saving software to control an air conditioning system in a building, its base/main concern is to manage energy consumption; it turns ACs off for uninhabited areas when the energy level exceeds a certain threshold. At the same time, the system is affected by an asset safety concern, the IT department controls the air conditioning system operation regarding the safety of certain assets in the building e.g. servers. On the other hand, the operation of the system is further influenced by the maintenance department rules that turn off ACs in case of any problems related to the hardware parts of the AC, etc.

Concerns are used to alter the behavior of the base system by applying their conclusion. It should be noted that concerns do not only affect the base system, but rather affect other concerns of the system; for example, the user's preferences in the illustrative scenario.

Concerns are represented as a set of distributed defeasible theories D. Each theory has a unique name. System components are referred to as,

Sys-c = $\{\beta\} \cup D$, where $D = \{D_1,\ldots, D_n\}$, n is the number of concerns in L

The formal definition of each theory in D flows naturally from the definition of classical defeasible theory, however, the components of each theory would be superscripted with the concern name, e.g. concern theory D_i is a tuple $(F^{D_i}, R^{D_i}, >^{D_i})$. Each concern is activated by one trigger using a meta-rule.

It should be mentioned that throughout the work of this paper, the decision inferred by *β* is called *a base conclusion*, while the decision inferred by any concern theory is superscripted with the name of the concern, e.g. $Pass^{D1}(X)$, means that according to concern D_1, the conclusion Pass(X) is inferred.

D. *Inter-Concerns Conflict Resolution λ*

Basically, concerns conclusions overrule the base system conclusion. In other words, when a query is issued for a service provided by a system that includes multiple concerns, if any of these concerns concluded a decision that contradicts the conclusion concluded by the base system, the concerns conclusion would be preferred; this is exactly where the effect of context in changing the behavior of the system, is captured.

However, in certain contexts several concerns can be activated; this might lead to conflicts in the decision-making process. This case happens when the conclusion inferred from one concern i.e. defeasible theory contradicts the conclusion inferred from another concern(s). In this case, the system would use λ, a conflict resolution mechanism that follows a prioritized ordering scheme to resolve inter-concern conflicts.

$\lambda = \{(D_i, D_j) \in \text{Sys-c}^2 \mid (D_i \sqsupset D_j)\ D_i,D_j \in D$ and

$(D_k \sqsupset \beta)\ \forall k\ D_k \in D\}$

λ is a total ordering relation that is defined over system components, it uses the operator $\sqsupset$ to denote priority, such that $D_i \sqsupset D_j$ states that the conclusion of D_i is preferred over the conclusion of D_j, and so on. However, it has another property; the definition also implies that the conclusion of any concern is preferred over the base conclusion.

It is important to notice that a total ordering relation is used instead of a partial ordering relation to prioritized concerns. Whenever a new concern is added, λ should be re-evaluated;

and the relation between the newly added concern and all other system components should be set in a proper way. It is the designer's responsibility to decide how to prioritize concerns based on the criticality level in the decision-making process.

VIII. Formal Proofs

The provability of the framework would be discussed according to the concern-level local distributed theory and the system level theory.

A. Concern Level Proof

Each concern is represented as a defeasible theory D, the probability of a defeasible logic is based on the concept of a derivation (or proof) from the theory [22]. A derivation is a finite sequence Pn=(P(1), . . ., P(n)) of tagged literals satisfying the following four conditions (i.e. the inference rules for each of the four kinds of conclusion).

Let P(1..i) denote the initial part of the sequence P_n of length i where $i \leq n$. Then a conclusion, proved subsequently [16], could be either:

(1) Definitely provable in D.
$+\Delta$: If $P(i+1) = +\Delta B$ then
(1) $B \in F$ or
(2) $(\exists R1 \in Rs[B])(\forall A \in Ant(R1): +\Delta A \in P(1..i))$.
(2) Not definitely provable in D.
$-\Delta$: If $P(i+1) = -\Delta B$ then
$(\forall R1 \in Rs[B]) (\exists A \in Ant(R1): -\Delta A \in P(1..i))$.

(3) Defeasibly provable in D.
$+\delta$: If $P(i + 1)= +\delta B$ then
(1) $+\Delta B \in P(1..i)$ or
(2) (2.1) $(\exists R1 \in R[B])(\forall A \in Ant(R1): +\delta A \in P(1..i)$
and (2.2) $(-\Delta\neg B \in P(1..i))$
and (2.3) $(\forall R1 \in R[\neg B])$ either
(2.3.1) $((\exists A \in Ant(R1): -\delta A \in P(1..i))$ or
(2.3.2) $((\exists R2 \in R[B])$ such that $(R2>R1)$ and
$(\forall A \in Ant(R2): +\delta A \in P(1..i))$

B is defeasibly provable from D, if either: (1) B is definitely provable or (2) use the defeasible part of D which requires: (2.1) finding a strict or defeasible rule with consequent B which can be applied, **and** (2.2) showing that ¬ B is not definitely provable **and** (2.3) counterattacking each rule that attacks the conclusion B by **either** (2.3.1) proving that the attacking rule is not defeasible proved **or** (2.3.2) finding a stronger rule that defeasible prove B.

Not defeasibly provable in D [38].

$-\delta$: If $P(i + 1)= -\delta B$ then
(1) $-\Delta B \in P(1..i)$ or
(2) (2.1) $(\forall R_1 \in R[B])$ ($\exists A \in Ant(R1): -\delta A \in P(1..i)$
or (2.2) $(+\Delta\neg B \in P(1..i))$ or
(2.3) $(\exists R2 \in R[\neg B])$ such that
(2.3.1) $((\forall A \in Ant(R2): +\delta A \in P(1..i))$ and
(2.3.2) $((\forall R3 \in R[B])$ either $(R2>R3)$ or
$(\exists A \in Ant(R3) : -\delta A \in P(1..i))$ ■

B. System-Level Proof

For a conclusion to be inferred from the framework it should be either strictly or defeasibly approved by the base system (when no concerns are addressed) or by a higher priority concern. Two types of tagged literals are introduced to approve/not approve a conclusion:

- $+\theta$ B, globally approved in system L, which means that there is a reasoning chain that strictly or defeasibly approves B in concern Di that is not defeated by any applicable reasoning chain of a higher priority concern Dj, where both Di and Dj ∈ Sys-c.
- $-\theta$ B, not globally approved in system L, which means that every reasoning chain that strictly or defeasibly approves B in concern Di is defeated by an applicable reasoning chain of a higher priority concern Dj, where both D_i and D_j ∈ Sys-c.

The tagged literals can be formally defined by the following proof conditions:

- Globally defeasibly provable in L:

$+\theta$: If $P(i + 1)= +\theta B$ then
(1) $((+\Delta B^{\beta})$ or $(+\delta B^{\beta}))$ and $(D = \{\}))$ or

(2) $(\exists M_i \in M[+\delta D_i]$ $(\forall g_i \in Ant(M_i): +\Delta g_i \in P(1..i))$ and

$(\exists M_j \in M[+\delta D_j]$ $(\forall g_j \in Ant(M_j): +\Delta g_j \in P(1..i))$ and

$((+\Delta B^{Di})$ or $(+\delta B^{Di}))$ and either

(2.1) $(\forall D_j \in D)$ $((+\Delta\neg B^{Dj})$ or $(+\delta\neg B^{Dj}))$ and $(D_i \sqsupset D_j)$ or (2.2) $(\forall D_j \in D)$ $((-\Delta\neg B^{Dj})$ or $(-\delta\neg B^{Dj}))$ and $(D_j \sqsupset D_i)$

- Globally not defeasibly provable in L

$-\theta$: If $P(i + 1)= -\theta B$ then
(1) $((+\Delta\neg B^{\beta})$ or $(+\delta\neg B^{\beta}))$ and $(D = \{\}))$ or

(2) $(\exists M_i \in M[+\delta D_i]$ $(\forall g_i \in Ant(M_i): +\Delta g_i \in P(1..i))$ and

$(\exists M_j \in M[+\delta D_j]$ $(\forall g_j \in Ant(M_j): +\Delta g_j \in P(1..i))$ and

$((+\Delta\neg B^{Di})$ or $(+\delta\neg B^{Di}))$ and either

(2.1) $(\forall D_j \in D)$ $((+\Delta B^{Dj})$ or $(+\delta B^{Dj}))$ and $(D_i \sqsupset D_j)$ or

(2.2) $(\forall D_j \in D)$ $((-\Delta B^{Dj})$ or $(-\delta B^{Dj}))$ and $(D_j \sqsupset D_i)$ ■

IX. Case Study and Analysis

In this section, a formalization of the illustrative scenario would be presented and analyzed.

A. Case Study

The illustrative scenario's base system is represented in the proposed framework as follows:

$\beta = (F^{\beta}, R^{\beta}, >^{\beta})$,

F^{β} = {calling(X,Y), unknown(X), international(X)}

R_1^{β} :calling(X,Y) $\wedge$ in-urgent(X) $\Rightarrow$ ring(Y)

R_2^{β} :calling(X,Y) $\wedge$ international(X)$\Rightarrow$ ring(Y)

R_3^{β} :calling(X,Y) $\wedge$ unknown $\Rightarrow$ ¬ring(Y)

$>^{\beta} = \{(R_1^{\beta} > R_3^{\beta}), (R_2^{\beta} > R_3^{\beta}) \}$

However, Concern D_1 encodes the lecturer's preferences regarding call management. The lecturer makes his decision based on three contextual attributes, his location, schedule and his status that could be either busy or not based on the number of students that are active during the online session. Professor Sami's preferences are formalized by a contextual concern theory D_1 which is activated using meta-rule M_1 due to a flag on the user's phone.

M_1: flag(on) $\Rightarrow D_1$

F^{D1} = {calling(X,Y), samsung-lab(Y), lecture-time(Y), busy(Y), nStudents(Y)}

R_1^{D1} :calling(X,Y) $\wedge$ samsung-lab(Y) $\Rightarrow$ ¬ring(Y)

R_2^{D1} :calling(X,Y) $\wedge$ lecture-time(Y) $\Rightarrow$ ¬ring(Y)

R_3^{D1} : nStudents(Y) <5$\Rightarrow$ ¬busy(Y)

R_4^{D1} : ¬busy(Y) $\Rightarrow$ ring(Y)

$>^{D1} = \{(R_4^{D1} > R_2^{D1}), (R_4^{D1} > R_1^{D1}) \}$

Stakeholders concern for urgent meetings is formalized by the following context theory D_2:

$D_2 = (F^{D2}, R^{D2}, >^{D2})$,

F^{D2} = {calling(X,Y), lecture-time(Y), invited(Y)}

R_1^{D2} :calling(X,Y) $\wedge$ lecture-time(Y) $\Rightarrow$ ¬ring(Y)

R_2^{D2} :calling(X,Y) $\wedge$ invited(Y) $\Rightarrow$ ring(Y)

$>^{D2} = \{(R_1^{D2} > R_2^{D2})\}$

When an urgent meeting is set up, the stakeholder activates an immediate indication. The meta-rule that activates this concern is M_2,

M_2: urgent-meeting $\Rightarrow D_2$

As this concern is added by a higher authority, the dean, the designer decided to set λ as,

$\lambda = \{(D_2 \sqsupset D_1), (D_2 \sqsupset \beta), (D_1 \sqsupset \beta)\}$

B. Analysis

In this framework, Argumentation is used to analyze the conclusions of the contextual distributed theories. Argumentation is a mechanism used for tracing the reasoning process over a knowledge base that contains possibly partial and/or conflicting knowledge [39] [40]. It can be used to obtain useful conclusions from the defeasible logic theory.

Definition: Suppose D (F, R, >), is a defeasible logic theory and B is a ground literal. Arg is said to be an argument that supports the conclusion B from D, denoted by <Arg, B>, if Arg is a minimal set of defeasible rules (Arg $\subseteq$ Rd), such that:

1) B can be defeasibly derivate from (Arg $\cup$ F $\cup R_s$),

2) No pair of contradictory literals can be defeasibly derived from (Arg $\cup$ F $\cup R_s$), and

3) Arg contains no rule that contains an antecedent that is complementary to an antecedent of another rule in Arg.

With respect to analyzing the behavior of the theory in the case study using argumentation, suppose the situation when the dean asked the secretary Linda to call Professor Sami. Professor Sami has the preferences flag set on. Linda's number is in the urgent list; Sami is inside Samsung Lab and is in an active session with five students.

There are two arguments that support conflicting conclusions:

< $Arg_{1,\beta}$, $ring^{\beta}$> = ({calling(linda,sami) $\wedge$ in-urgent(linda)}, ring(sami))

< $Arg_{1,D1}$, ¬$ring^{D1}$> = ({calling(linda,sami) $\wedge$ samsung-lab(sami)}, ¬ring(sami))

Although argument $A_{1,\beta}$ supports the conclusion ringβ, argument $A_{1,D1}$ counterarguments $A_{1,\beta}$ i.e. it attacks its conclusion and vice versa. The global conflict resolution mechanism λ is used to support the conclusion of the higher priority theory. In this case $\lambda = \{(D1 \sqsupset \beta)\}$ and (¬$ring^{D1}$) is globally approved.

Now, suppose the preferences flag is on and the urgent meeting indication is active, meta-rule M_1 will activate concern D_1 and meta-rule M_2 will activate concern D_2. Linda called Professor Sami. Sami is invited and he is in the Samsung-lab but he has no lecture at this time, he is giving advice to 5 students. The argumentation process would go as follows:

<$Arg_{1,\beta}$, $ring^{\beta}$> = ({calling(linda,sami) $\wedge$ in-urgent(linda)}, ring(sami))

<$Arg_{1,D1}$,¬$ring^{D1}$> = ({calling(linda,sami) $\wedge$ samsung-lab(sami)}, ¬ring(sami))

<$Arg_{1,D2}$, $ring^{D2}$> = ({calling(linda,sami) $\wedge$ invited(sami)}, ring(sami))

According to D_1, (ring) cannot be inferred as it is defeasibly approved that Sami is not busy which blocks the conclusion of (ring), add to that (¬ring) is defeasibly approved by R_1^{D1}. However, according to λ, $Arg_{1,D1}$ is attacked by a higher priority argument $Arg_{1,D2}$ that supports the conclusion ($ring^{D2}$). As long as the external contextual attribute (urgent-meeting) is active, ($ring^{D2}$) is globally approved.

X. Discussion

The proposed framework differs from all the approaches in literature in that it captures the effect of the different conceptual aspects of context using defeasible logic. It provides a powerful mechanism to manipulate multi-context distributed systems. It complies with the main characteristics of ubiquitous and distributed systems in providing transparency, reliability, and scalability. At the same time, it enables the evaluation of distributed decisions and produces globally justifiable conclusions. This is achieved using triggers and the relation between active concerns, unlike the bridge rules and mapping rules that were used in literature.

The consistency of the system is attained by the consistency of defeasible logic itself as for any statement; there is a proof/reasoning chain that can determine whether or not that statement holds and inconsistencies can be detected using the proof theory.

XI. Conclusions and Future Work

In this paper, we proposed a novel framework for context-awareness that can contextualize any deployed application. The framework is based on a conceptual analysis of context; it captures the behavior of contextual knowledge as it penetrates into deployed applications. It fairly simulates how the human being perceives context either as plain attributes or as concerns that need to be considered in order to make better decisions, change behavior and personalize services.

The framework is efficiently mapped to defeasible theory. It is generic, flexible and scalable. It allows the system to make justifiable decisions regardless of the number of available contextual attributes, concerns, or the number of administrative authorities that control the decision-making process. Its main strength lies in its distributed approach of reasoning and its ability to represent concerns in defeasible theories.

The analysis showed that the framework is able to capture both the contextual aspects and the concerns of different authorities in the system. The consistency in the system is attained by two levels of conflict resolution mechanisms, concern level, and system level.

The proposed extension of the defeasible theory using meta-rules improved the expressiveness power of the logic through enabling nonmonotonic reasoning over sets of defeasible theories rather than defeasible rules.

We have investigated the capabilities of the system in reasoning in environments with multiple entities that have cross-cutting concerns. Future work may exploit the flexibility of the proposed framework and its augmented power of expressing complex systems in providing personalized services i.e. entities/users that share the same concern but each one of them preserves its own right of manipulating contextual knowledge in its own way. For example, according to the scenario, more than one user has the same concern (e.g. preferences) but each of them has its own setting of preferences.

Further work would also consider investigating the capabilities of this framework by implementing it on real-world ubiquitous systems where context plays an important role.

Another aspect to be considered in contextual reasoning is the effect of context on the manipulation of the prioritizing scheme of both the classical defeasible logic and the proposed framework. We believe the management of this issue can present a magnificent step in the field of context-awareness.

References

[1] Park, I. S., Kim, W. T., & Park, Y. J. (2004, February). A ubiquitous streaming framework for multimedia broadcasting services with QoS based mobility support. In International Conference on Information Networking (pp. 65-74). Springer Berlin Heidelberg.

[2] Schilit, B. N., & Theimer, M. M. (1994). Disseminating active map information to mobile hosts. IEEE network, 8(5), 22-32.

[3] Brown, P. J., Bovey, J. D., & Chen, X. (1997). Context-aware applications: from the laboratory to the marketplace. IEEE personal communications, 4(5), 58-64.

[4] Capurso, N., Mei, B., Song, T., Cheng, X., & Yu, J. (2018). A survey on key fields of context awareness for mobile devices. Journal of Network and Computer Applications, 118, 44-60.

[5] Schmidt, A., Beigl, M., & Gellersen, H. W. (1999). There is more to context than location. Computers & Graphics, 23(6), 893-901.

[6] Gruber, T. R., Brigham, C. D., Keen, D. S., Novick, G., & Phipps, B. S. (2018). U.S. Patent No. 9,858,925. Washington, DC: U.S. Patent and Trademark Office.

[7] Dey, A. K. (2001). Understanding and using context. Personal and ubiquitous computing, 5(1), 4-7.

[8] Fischer, G. (2012). Context-aware systems: the 'right' information, at the 'right' time, in the 'right' place, in the 'right' way, to the 'right' person. In Proceedings of the International Working Conference on Advanced Visual Interfaces (pp. 287-294). ACM.

[9] Ryan, N., Pascoe, J., & Morse, D. (1999). Enhanced reality fieldwork: the context aware archaeological assistant. Bar International Series, 750, 269-274.

[10] Pollock, J. L. (1996). OSCAR: A general-purpose defeasible reasoner. Journal of applied non-classical logics, 6(1), 89-113.

[11] Moubaiddin, A., & Obeid, N. (2009). Partial information basis for agent-based collaborative dialogue. Applied Intelligence, 30(2), 142-167.

[12] Obeid, N., & Moubaiddin, A. (2010). Towards a formal model of knowledge sharing in complex systems. In Smart Information and Knowledge Management (pp. 53-82). Springer, Berlin, Heidelberg.

[13] Obeid, N., & Rao, R. B. (2010). On integrating event definition and event detection. Knowledge and information systems, 22(2), 129-158.

[14] Obeid, N. (2012). Three-Values Logic and Non-Monotonic Reasoning. COMPUTING AND INFORMATICS, 15(6), 509-530.

[15] Moubaiddin, A., & Obeid, N. (2013). On formalizing social commitments in dialogue and argumentation models using temporal defeasible logic. Knowledge and information systems, 37(2), 417-452.

[16] Sabri, K. E., & Obeid, N. (2016). A temporal defeasible logic for handling access control policies. Applied Intelligence, 44(1), 30-42.

[17] Moubaiddin, A., Salah, I., & Obeid, N. (2018). A temporal modal defeasible logic for formalizing social commitments in dialogue and argumentation models. Applied Intelligence, 48(3), 608-627.

[18] Mobaiddin, A., & Obeid, N. (2018, June). On Commitments Creation, Compliance and Violation. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 465-476). Springer, Cham.

[19] Bikakis, A. and Antoniou, G. (2010). Rule-based contextual reasoning in ambient intelligence. In International Workshop on Rules and Rule Markup Languages for the Semantic Web (pp. 74-88). Springer Berlin Heidelberg.

[20] McCarthy, J. (1987). Generality in Artificial Intelligence. Communications of the ACM, 30(12), 1030-1035.

[21] Antoniou, G., & Williams, M. A. (1997). Nonmonotonic reasoning. Mit Press.

[22] Nute, D. (2001, October). Defeasible logic. In International Conference on Applications of Prolog (pp. 151-169). Springer Berlin Heidelberg.

[23] Governatori G (2005) Representing business contracts in RuleML. International Journal of Cooperative Information Systems 14(2-3):181–216.

[24] Governatori, G., Olivieri, F., Scannapieco, S., & Cristani, M. (2012). Revision of defeasible logic preferences. arXiv preprint arXiv:1206.5833.

[25] Governatori G, Rotolo A, Padmanabhan V (2006) The cost of social agents. In: Proceedings of the AAMAS 2006, pp 513–520.

[26] Van Kranenburg, H., Bargh, M. S., Iacob, S., & Peddemors, A. (2006). A context management framework for supporting context-aware distributed applications. IEEE Communications Magazine, 44(8), 67-74.

[27] Dey AK et al. (2001) A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Cont Aw Comput-HCI J 16:97–116.

[28] Schilit WN (1995). A context-aware system architecture for mobile distributed computing. Dissertation, Columbia University.

[29] Ranganathan, A., & Campbell, R. H. (2003). An infrastructure for context-awareness based on first order logic. Personal and Ubiquitous Computing, 7(6), 353-364.

[30] Alirezaie, M., Renoux, J., Köckemann, U., Kristoffersson, A., Karlsson, L., Blomqvist, E., ... & Loutfi, A. (2017). An ontology-based context-aware system for smart homes: E-care@ home. Sensors, 17(7), 1586.

[31] Obeid, M., Obeid, Z., Moubaiddin, A., & Obeid, N. (2019, July). Using Description Logic and Abox Abduction to Capture Medical Diagnosis. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 376-388). Springer, Cham.

[32] Uddin, I., Rakib, A., Haque, H. M. U., & Vinh, P. C. (2018). Modeling and reasoning about preference-based context-aware agents over heterogeneous knowledge sources. Mobile Networks and Applications, 23(1), 13-26.

[33] Antoniou, G., Bikakis, A., Karamolegou, A., & Papachristodoulou, N. (2006). A Context-Aware Meeting Alert Using Semantic Web and Rule Technology-Preliminary Report. Semantic Web Technology For Ubiquitous & Mobile Applications (SWUMA'06), 23.

[34] Dastani, M., Governatori, G., Rotolo, A., Song, I., & Van Der Torre, L. (2007, November). Contextual agent deliberation in defeasible logic. In Pacific Rim International Conference on Multi-Agents (pp. 98-109). Springer, Berlin, Heidelberg.

[35] Harel, D. (1979). First-order dynamic logic (Vol. 68). Berlin: Springer.

[36] Al-Zyoud, M., Salah, I., & Obeid, N. (2012, November). Towards a model of context awareness using web services. In International Conference on Computational Collective Intelligence (pp. 121-131). Springer, Berlin, Heidelberg.

[37] Musumba, G. W. and Nyongesa, H. O. (2013). Context awareness in mobile computing: A review. International Journal of Machine Learning and Applications, 2(1), 5-pages.

[38] Antoniou, G., Maher, M. J., & Billington, D. (2000). Defeasible logic versus logic programming without negation as failure. The Journal of Logic Programming, 42(1), 47-57.

[39] Moubaiddin, A., & Obeid, N. (2008). Dialogue and argumentation in multi-agent diagnosis. In New Challenges in Applied Intelligence Technologies (pp. 13-22). Springer, Berlin, Heidelberg.

[40] García, A. J., & Simari, G. R. (2014). Defeasible logic programming: Delp-servers, contextual queries, and explanations for answers. Argument & Computation, 5(1), 63-88.