Computer Science and Information Systems 11(1):171–193 DOI: 10.2298/CSIS130209002N

Rule-Based Solution for Context-Aware Reasoning on Mobile Devices

Grzegorz J. Nalepa[1] and Szymon Bobek[1]

AGH University of Science and Technology
Al. Mickiewicza 30, 30-059, Krakow, Poland
gjn@agh.edu.pl, szymon.bobek@agh.edu.pl

Abstract. With the rapid evolution of mobile devices, the concept of context aware applications has gained a remarkable popularity in recent years. Smartphones and tablets are equipped with a variety of sensors including accelerometers, gyroscopes, pressure gauges, light and GPS sensors. Additionally, the devices become computationally powerful which allows real-time processing of data gathered by their sensors. Universal network access via WiFi hot-spots and GSM network makes mobile devices perfect platforms for ubiquitous computing. Most of existing frameworks for context-aware systems, are usually dedicated to static, centralized, client-server architectures. However, mobile platforms require from the context modeling language and inference engine to be simple and lightweight. The model should also be powerful enough to allow not only solving simple context identification tasks but more complex reasoning. The original contribution of the paper is a proposal of a new rule-based context reasoning platform tailored to the needs of such intelligent distributed mobile computing devices. It contains a proposal of a learning middleware supporting context acquisition. The platform design is based on a critical review and evaluation of existing solutions given in this paper. A preliminary evaluation of the platform is given along with use cases including a social system supporting crime detection and investigation.[1]

Keywords: context-aware, mobile devices, knowledge management.

1. Introduction

The notion of context has been important in application modeling for many years. The research community aimed at giving constructive and precise definition of this concept, which proved to be a non-trivial task. A general observation is that "context is about evolving, structured, and shared information spaces, and that such spaces are designed to serve a particular purpose" [15]. Research in the area of pervasive computing and ambient intelligence aims to make use of context information to allow devices or applications behave in a context-aware, thus "intelligent" way. Dey [19] defines context as *"any information that can be used to characterize the situation of an entity"*.

Creation of context-aware applications can be considered on several levels. The main challenges are how to model, represent and classify context, and then how to reason about it. Raw information captured by device sensors is usually useless without further processing and interpretation. Therefore, the issue of context representation needs to be solved.

[1] The paper is supported by the AGH UST Grant 11.11.120.859.

In order to do that, a practical model or knowledge representation has to be selected. Considering the fact, that the classification and reasoning tasks often require soft real time responsiveness of the application, this selection might be a non trivial one. Using the specified model context identification and classification can be provided. Context classifier are most commonly build, or trained using machine learning techniques. The output of this layer includes concepts describing context. Using it, the context-based reasoning is performed. Reasoning task is much more computationally demanding. Thus, considering the expected system responsiveness logic-based solutions are coupled or replaced with simple pattern-matching approaches. A specific combination and setup of the classification and reasoning layer, is commonly referred to as a context-aware architecture. A practical implementation of such an architecture, usable from the developer point of view is usually called a framework, or middleware. Some frameworks can be targeted at an off line context processing, while others are designed to work in soft real-time. The growing popularity of mobile platforms, e.g. smartphones or tablets, stimulates the development of such frameworks. On the other hand mobile platforms still impose serious resource constraints.

The primary objective of this paper is to provide an overview of context-aware approaches on mobile platforms. Moreover, selected frameworks are also described. The original contribution of the paper is a proposal of a rule-based context reasoning platform for mobile computing. The platform design is based on a critical review of existing solutions. The rest of the article is organized as follows: In Section 2 a motivation for the research is provided. In Section 3 a review of context modeling approaches is presented. The most common architectures for context-aware systems are described in Section 4. Selected frameworks targeted at mobile devices are briefly characterized in Section 5. In Section 6 a custom architecture and modeling language for context-aware mobile devices are proposed. Section 7 provides a preliminary evaluation of this architecture, and in Section 8 some representative use cases are described, including a social system supporting crime detection and investigation. The summary and directions for future work are given in Section 9.

2. Motivation and Related Work

Research on context-aware systems resulted so far in many different approaches and frameworks. However, the diversity of the field, as well as the rapid development of the hardware used requires further development. This is especially true for context-aware applications using ubiquitous mobile devices. To provide a full support for all of the challenges that we believe are crucial for todays mobile computing (e.g. smartphones or tablets) new solutions need to be provided. Not only the issue of context modeling and classification needs to be addressed, but more importantly an appropriate context-based reasoning layer has to be created. The most severe challenges are:

1. Energy efficiency – most of the sensors, when turned on all the time, decrease the mobile device battery level very fast. This has impact on usability of the system and ecological aspects regarding energy saving.
2. Data privacy – most of the users do not want to send information about their location, activities, and other private data to external servers. Hence, the context reasoning should be performed by the mobile device itself.

3. Resource limitations – although mobile phones and tablets are becoming computationally powerful, the context aware system has to consume as low CPU and memory resources as possible in order to be transparent to the user and other applications.
4. System responsiveness – in mobile environment context changes very fast, hence no delays are admissible in processing contextual data.
5. Context data distribution – in mobile pervasive environments many devices produces huge amount of contextual information, hence the quality measures should be developed and distribution methods designed to fit characteristics of such unstable and dynamic network [29,5].

All of these require from the modeling language and inference engine to be simple and lightweight. On the other hand, the model should be powerful enough to allow not only solving simple context identification tasks but also more advanced context processing and reasoning. This gives motivation for evaluating existing architectures w.r.t. the requirements of mobile computing. New approaches, tailored to the needs of such devices should be provided.

Aforementioned challenges were usually approached by the programmers at the very last phase of the development of context-aware application, or were not approached at all. We believe that solutions to these challenges should be provided by an appropriate framework architecture. This will allow the programmer to design and build context-aware application in an efficient way, making the development easier and less error prone.

Finally, the *portability* of the application needs to be considered. This is a complex but important issue. While todays landscape of operating systems for mobile devices seems to be balanced, with a split between Android and iOS, it may rapidly change in the near future, e.g. with the wide introduction of Tizen from Samsung. Moreover, Android devices are quite diverse, so even though Android provides an abstraction layer for the sensors they have, the characteristics of specific devices need to be considered. While this issue is not explicitly considered in this paper, it is included in the general motivation for our approach, as it will be discussed in the evaluation.

3. Modeling And Reasoning Approaches

In this section several approaches for context modeling are presented. The summary and the comparison is also given in Table 1. We took into consideration following aspect of context modeling methods, that we believe are crucial for efficient designing and developing of context-aware applications and systems:

1. formalization,
2. simplicity,
3. expressiveness,
4. support for inference,
5. handling of uncertainty, and
6. existing tools that support design.

Below, the main approaches are briefly discussed.

	Key-Value	Logic	Rules	GM	PGM	Ontologies	Processes
Formalism	None	High	Med	Low	Low	High	Low
Simplicity	High	Med	Med	Med	Low	Low	Med
Expressiveness	Low	Med	Med	High	Med	High	High
Reasoning support	Low	Med	High	Low	Med	Med	Low
Handling uncertainty	Low	Med	Med	Med	High	Low	Low
Design tools support	None	Low	High	High	Med	High	High

Table 1. Comparison of context-modeling techniques.

Key-Value Representation One of the simplest way to define context is to use pairs of a form: `key-value`. The `key` is usually a name that defines a context property. For instance it can denote location (e.g. room) or time (e.g. daytime). The complex contexts can be represent as a union of several `keys`. The `value` represents current state of the context property (e.g. kitchen, etc.)

The ActiveBadges [53] based system called *Watchdog* described in [50] uses the key-value context representation. A simple example might be:

```
badge location event-type action
```

The context in the Watchdog system is represented by three keys: `badge` denoting an electronic ID, `location` denoting a location of a person wearing a badge and a `event-type` that describes activity of a person. Remaining parameter represents an action that should be performed when previous three keys match actual context. An example of a context state might be following:

```
Coffee Kitchen arriving "play -v 50 /sounds/rooster.au"
```

We can read this as a statement: *When a person that wears badge Coffee arrives to the Kitchen, then play sound*.

Another example of a system that uses *key-value* modeling is FAWIS [16]. It is a framework for representation and translation of context information in adaptive Web-based applications. In FAWIS, context is a collection of profiles, that can be user account, network contention type, hardware or user browser.

The reasoning in key-values models is usually supported by a simple matching engine. When the keys match the actual context values, an action is triggered. Key-value model does not provide formalization and visualization of the model, nor provide design tools. It does not incorporate hierarchy nor any sophisticated structure into model which is flat. The inference is supported usually by very simple matching algorithm that does not allow form more sophisticated reasoning. However, it is very simple to implement.

Logic-based Models These approaches are widely used in context-aware systems. They enable automated inductive and deductive reasoning to be done on contextual information and due to their strong formalization, allow for verification and validation of context models. There are several approaches that use logic to represent context. First order logic allows for an expressive description of context using boolean operators and existential and universal quantifiers [38,48]. Fuzzy logic and probabilistic logic is used to handle uncertainty of the environment and to deal with the imperfections of the data [47]. Context

lattices approach is used to represent low-level semantics on domain knowledge and sensor data, and to derive high-level semantics on human activities [56]. Description logic is usually used in combination with ontologies. It models concepts, roles and individuals, and their relationships. It also provides simple reasoning capabilities that resolves classification tasks [25].

An example of a system that uses first order logic to describe context can be found in [48]. The approach presented in that paper uses Prolog programming language for context representation. An example of a context description statement is presented below in pseudo code:

```
#People(Room 2401, ">=",3) AND
   Application (PowerPoint, Running)
 => RoomActivity(20401,Presentation)
#People(Room 2401, ">=",3) AND
   NOT EXSIST x Application(x, Running)
 => RoomActivity(20401,Meeting)
```

Logic-based models provide strong formalization, though their flexibility might be limited. There is also lack of tools that provide visualization of models defined in logic languages. There exist a lot of programing languages and reasoners for expressing and processing knowledge encoded with logic based languages. However, dedicated reasoners are rarely available for mobile platforms.

Rules for Context Representation Rule-based systems have been in use for several decades in various branches of engineering. Hence, they have also been used in context-aware applications, both as a representation of models and as a support for reasoning [17,21,51]. A rule-based system consists of three main elements: knowledge base, fact base and inference engine. Knowledge base is considered as a set of rules, usually of a form of production rules: `IF <conditions> THEN <action>`. Fact base contains information used to check which rules conditions are satisfied. Inference engine implements mechanism that allow for processing of rules within a knowledge base.

One of the most popular tools for context-aware applications that implements rule-based approach is Context Toolkit [19]. Example of a rule written in a Context Toolkit notation looks as follows:

```
<Reference name="Off">
   <Query name="lightOff">
      (OR
         (EQUAL presence 0)
         (GREATER brightness brightnessThreshold))
   </Query>
   <Outcome outAttribute="light">0</Outcome>
   <ServiceInput service="LightService" function="lightOff" />
</Reference>
```

The example rule can be read as follows: *If there is no person in a room or brightness exceeds some threshold, then turn out the light in the room.*

Context Toolkit uses custom rule language and inference engine. However, there are several commonly used rule-based environments that provides advanced reasoning mechanisms and complex rule languages. Examples of such are: Clips [2], Jess [3], Drools [4]. Although there were attempts to use these tools for context-aware applications [7,21], they are still not popular in this area.

Rules incorporates more powerful reasoning mechanisms than those available in key-value approach. They allow for assertions of new facts to knowledge base that can later be used as an input for other rules making the knowledge base more dynamic. They provide self-explanation mechanism that is crucial for implementing ineligibility of a system [18]. Rule-based systems provide more advanced methods for selecting rules that should be processed, improving efficiency of the system.

Graphical models They use a graphical notation to express knowledge. They play an important role in software engineering, e.g. UML diagrams are widely used for supporting software development, ERD diagram are irreplaceable in designing relational databases schemes, and BPMN is used in process modeling. In context aware systems, a Context Modeling Language (CML) [24] developed by Henricksen *et al* is an example of such approach. CML is based on Object-Role Modeling language which was developed for conceptual modeling of databases. It provides a graphical representation of different classes and sources of context facts, relations between them and uncertainty of information. The underlying formalism is based on first order predicate logic and set theory. An example of a model designed with CML is presented in Figure 1.

CML approach allows for reasoning about situations that are derived from simple facts. It also allows for mapping its models to relational database and thus allowing for SQL-like queries on context data. Although the representation is human readable and expressive, it can be very complicated, especially when number of entities and relations between them grows.

Probabilistic graphical models They use a graph-based representation as the basis for compactly encoding a complex probability distribution over a high dimensional space. These interpretable models can be constructed and learned automatically and then manipulated by reasoning algorithms, what makes them one of the most important machine learning tool for modeling and reasoning about uncertain data [35].

In the area of context-aware systems, they are often used to model human behavior, activities [32], transportation routines [36], and other aspects that are characterized by high entropy of data. Bui *et al* [10] use a mult-layer Bayesian dynamic structure, called an Abstract Hidden Markov Model, to track an object and predict its future trajectory. Han-Saem Park [46] employs Bayesian networks to automatically recognize high-level contexts like activity or emotion of the user, based on mobile device logs. An example of a probabilistic graphical model for human activity recognition is presented in Figure 2.

Probabilistic graphical models provide a very effective way for modeling and reasoning on uncertain information. There are many tools supporting both visual modeling and reasoning tasks. However, exact inference in complex probabilistic models can be NP-hard task, and thus is not always tractable.

[2] http://clipsrules.sourceforge.net/

[3] http://herzberg.ca.sandia.gov/

[4] http://www.jboss.org/drools/

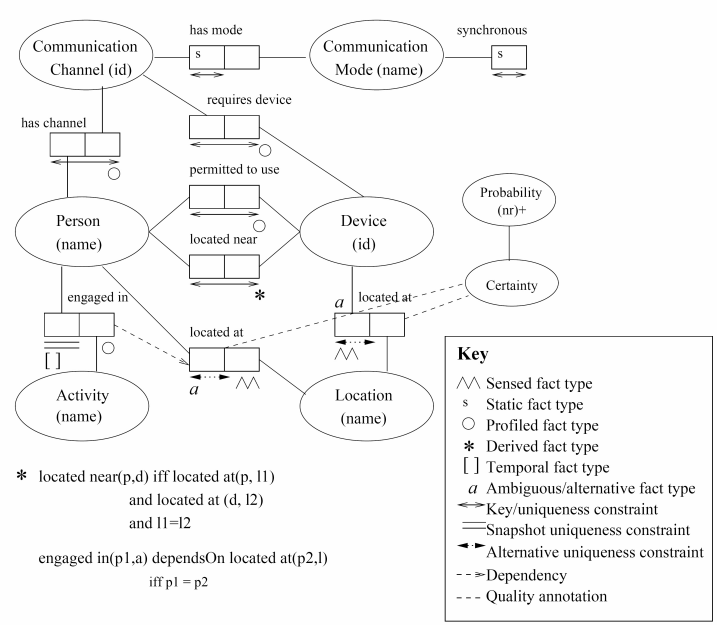

Fig. 1. An example of CML model [24]

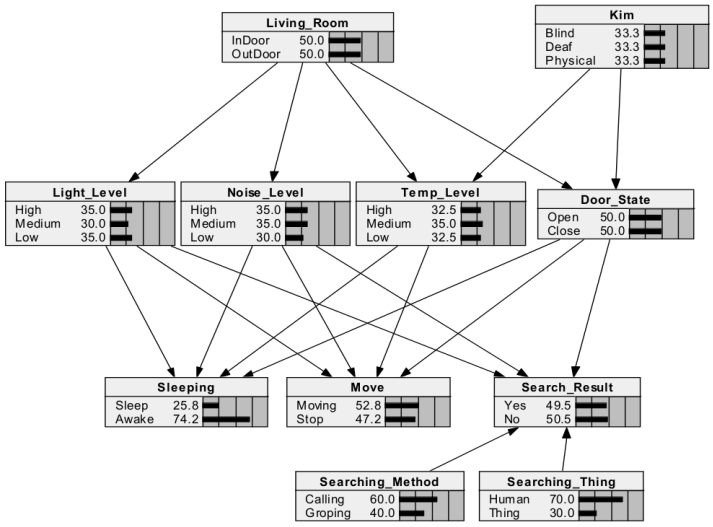

Fig. 2. An example of Bayesian network for human activity reasoning [34]

Ontologies An ontology is a formal explicit specification of a shared conceptualization [22]. Ontologies support a set of modeling primitives to define classes, individuals, their attributes and their relations. This approach is most often used to model context hierarchy and dependencies within a context space. The main advantage of ontological

context models is that they form a separate, independent layer in context-aware system. Due to standardized languages for serializing ontologies like OWL, or RDF, it is possible to reuse some well-defined models in many context-aware applications.

There are many frameworks that provide such ready-to-use ontological models. One of them is CONtext ONtology (CONON) [52], that provides an upper context ontology for smart home environments. It captures general concepts about basic context like users, locations, activities. Based on this context ontology, a logic reasoning can be applied to a model to check the consistency of context information, and to reason over low-level, explicit context to derive high-level, implicit context. The other example is SOUPA [13] – an ontology for modeling context in pervasive environments.

Ontologies have been successfully incorporated into various context-aware systems like CoBrA [12] for building smart meeting rooms, GAIA [49] for active spaces or SOCAM [23] – a middleware architecture for building context-aware mobile services. An example of a part of the CONON ontology, serialized to OWL, is presented below. It defines three classes and relations.

```
<owl:Class rdf:ID="ContextEntity"/>
<owl:Class rdf:ID="Location">
  <rdfs:subClassOf rdf:resource="#ContextEntity"/>
</owl:Class>
<owl:Class rdf:ID="IndoorSpace">
  <rdfs:subClassOf rdf:resource="#Location"/>
  <owl:disjointWith rdf:resource="#OutdoorSpace"/>
</owl:Class>
```

Reasoning in ontological models is usually supported by Description Logic. This however allows only for resolving simple classification tasks. It does not provide mechanisms for inferring more complex information from existing data, like in the case of rule representation models.

Ontologies have become very popular due to the formalization and hierarchization of knowledge they provide. There are many tools supporting design of ontologies like for instance Protege [5]. However, design and implementation are usually far more difficult and time consuming than in other approaches.

Processes These are one of the most popular methods for modeling flow of information and/or control within a sequence of activities, actions or tasks. Jaroucheh *et al* model contextual data with processes [27], which he defines as a directed graphs of states. Those states denotes user current, past and possible future context. This, according to Jaroucheh *et al*, is crucial in determining full user context. E.g. *User is in the room – has he just came in, or is he about to leave?* This cannot be answered with traditional context models, but becomes possible with processes, where current user context can be described with respect to previous and possible future states. An example process of user leaving home in the morning is presented on Figure 3. Processes can be automatically obtained from sensors logs with a process mining techniques [1].

[5] http://protege.stanford.edu

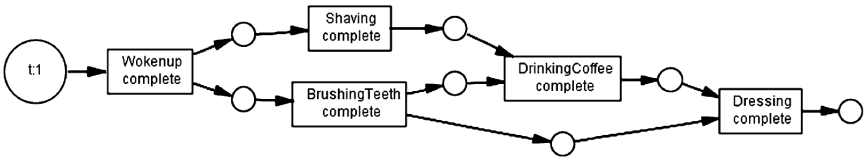

Fig. 3. An example of a Petri Net model of the context process [27]

Processes provide limited reasoning support, which focuses on simple tracking of a user current and future state according to the learned model. There are tools that allow modeling and automatic detection of processes e.g. ProM [6] or Disco [7].

Based on these different approaches, specific architectures are also made available.

4. Context-Based Systems Architectures

There are several commonly used approaches for modeling architecture of context-aware system. In this section a short overview or most common approaches will be given. Also a comparison of described architectures is presented in Table 2.

	Direct sensor Access	Middleware	Service Oriented	Centralized	Distributed
Energy efficiency	Low	High	Med	Med	Med
Privacy	High	High	Med	Low	Low
Resource efficiency	Low	Med	Med	Med	High
Responsiveness	High	High	Low	Low	Low

Table 2. Comparison of common context-aware architectures.

Considering the factors identified in Section 2, the following features are considered:

- **Energy efficiency** – most of the sensors, when turned on all the time, requires a lot of energy. This may reflect on usability of the system, e.g. on a mobile phone device, and ecological aspects regarding energy saving.
- **Privacy issues** – most of the users do not want to send information about their location, activities, and other private data to external servers. Hence, the architecture where there is a context server performing reasoning task would not be preferable.
- **System Responsiveness** – online work on mobile devices requires at least a soft-real time responsiveness where the results of the context identification and reasoning are provided to the user instantly, or with little delay.
- **Hardware resource limitations** – some platforms, like mobile devices, may have limited available CPU and memory resources. Although most of the electronic mobile devices are becoming computationally powerful, in some cases, the context aware

[6] http://www.processmining.org/prom/start
[7] http://fluxicon.com/disco/

system has to consume as low CPU and memory resources as possible in order to be transparent to the user and other applications.
- **Environment characteristic** – depending of the target environment of the context-aware system, different architecture will be used. For instance if we plan to incorporate context-aware application that will work in virtual world (e.g. Internet), the architecture choice may be different than for a context-aware system for mobile robots, or smart homes.
- **Context definition** – different types of context that are taken into consideration may require different architectural approaches. For instance, social context information will require different method for acquiring and share data, than a location-based context-aware application.

Below the previously discussed approaches are shortly characterized w.r.t to these features.

Direct sensor access It is the simplest architecture. It can be described as a system in which a part responsible for reasoning about a context, fetches information directly from sensors or other information source. This approach is usually not very energy efficient, however it preserver privacy issues, since no communication with external servers is usually needed, and the interpretation of the sensor data as well as reasoning is performed directly on the host device.

Middleware infrastructure It incorporates additional layer that encapsulates the sensor layer and is usually responsible for filtering and initial interpretation of data gathered by the latter. This approach eases extensibility and reusability of hardware dependent code responsible for sensing information. The middleware approach is one of the most common one in context-aware systems area. Comprehensive comparison of existing middlewares was presented in [31].

Service oriented architecture Such an architecture aims to allow building large-scale systems with loosely coupled elements. In context-aware applications this architecture is used mainly in pervasive environment, where variety of context information from many different sources has to be processed. This architecture usually does not preserves privacy nor energy efficiency issues since usually it assumes communication over the web between each of its elements. An example of a framework built in service oriented paradigm is SOCAM [23].

Centralized context server In a this architecture there is an entity that acts as a context server which may be responsible for gathering contextual data from clients, and reasoning with this data. This approach is especially useful when a context-aware system is composed of many mobile devices with limited resources. The server relieves mobile agents from performing reasoning tasks. On the other hand, one has to consider privacy issues connected with sending private contextual data to remote server, quality of service issues, etc. This approach is also characterized with rather low responsiveness that stems from a possible lack of network connection or communication delays. An example of framework using context server approach is described in [12].

Distributed context agents In a distributed architecture each component of the system acts as an autonomous context-aware agent that performs reasoning based on data gathered by itself and by other agents that at the moment it communicates. Agents may be implemented on different devices and can be given ability to represent different features of the environment in which they interact with other agents. Different agents may be designed to consider different contexts and can give different interpretation of the same contextual information. This type of architecture was described in details in [6,26].

5. Context-Aware Applications and Frameworks for Mobile Devices

In recent years, a lot of development was devoted to build applications that use mobile devices to monitor and analyze various user contexts. The availability of application distribution platforms for common mobile operating systems, e.g. Google Play for Android stimulated the popularity and adoption of such solutions. However, most of them focus only on a very narrow application area of context awareness. Most of them are end user applications, and not generic frameworks. Some selected representative cases are briefly described below.

The SocialCircuits platform [14] uses mobile phones to measure social ties between individuals, and uses long- and short-term surveys to measure the shifts in individual habits, opinions, health, and friendships influenced by these ties. Jung [28] focused on discovering social relationships (e.g., family, friends, colleagues and so on) between people. He proposed an interactive approach to build meaningful social networks by interacting with human experts, and applied the proposed system to discover the social networks between mobile users by collecting a dataset from about two millions of users. Given a certain social relation (e.g., isFatherOf), the system can evaluate a set of conditions (which are represented as propositional axioms) asserted from the human experts, and show them a social network resulted from data mining tools. Sociometric badge [45] has been designed to identify human activity patterns, analyze conversational prosody features and wirelessly communicate with radio base-stations and mobile phones. Sensor data from the badges has been used in various organizational contexts to automatically predict employee's self-assessment of job satisfaction and quality of interactions. Eagle and Pentland [20] used mobile phone Bluetooth transceivers, phone communication logs and cellular tower identifiers to identify the social network structure, recognize social patterns in daily user activity, infer relationships, identify socially significant locations and model organizational rhythms.

Beside research projects, there exist also a variety of application that are used for gathering information about context from mobile devices, like SDCF [3], AWARE [8], JCAF [4], SCOUT [55], ContextDriod [54], Gimbal [9]. These are mostly concerned with low-level context data acquisition from sensors, suitable fur further context identification. On the other hand, they do not provide support nor methodology for creating complex and fully customizable context-aware systems.

All of the approaches described in this section use their own dedicated methods for gathering and maintaining context. These methods are mostly not applicable for reuse,

[8] `http://www.awareframework.com/`
[9] `https://www.gimbal.com/`

or their functionality is limited to simple context matching. What is more, some of the systems do not provide any support for context modeling nor context reasoning, limiting their functionality only to identifying and collecting contextual information.

To solve the problem of reusability of context models, the two architectures were developed: CoBrA [11], and SOCAM [23]. They offer ontology approach for modeling context, however they do not take into consideration most important issues regarding development of context-aware applications for mobile, distributed systems, which have been listed in Section 2. Although SOCAM provides architecture for distributed mobile systems, it mostly solves problems of a low memory and CPU power of mobile agents, which nowadays is no longer a big issue for most of the mobile devices, like smart-phones or tablets. On the other hand, energy efficiency issue is still a big problem, which was not addressed by none of the solutions described in this Section.

Taking all above into consideration, we can argue that there are no tools dedicated to mobile platforms that will bind all the aspects of context-aware application development into a fully customizable framework that will provide methodology, modeling language, inference engine and communication protocols. Hence, the development of such framework will be a benefit for both researches and application developers. In the next section a proposal of a new approach is introduced.

6. Proposal of a Rule-Based Approach for Mobile Context-Based Applications

Considering the comparison of context modeling techniques presented in Section 3 and architectures described in Section 4, we propose a new, hybrid approach for designing and developing context aware systems for mobile devices. The proposed system is an approach that exploits strengths of service oriented and middleware architectures (See Figure 4). It incorporates an idea of a mobile device as an autonomous context-aware entity, equipped with intelligent middleware layer and context-based inference service. The architecture offers context-based reasoning to many applications at the same time, via a public (yet limited only to applications installed on the host device) API. The intelligent middleware, that act as a proxy between context sources and inference service, is able to learn sensor usage patterns and thus adjusting sampling rates to significantly improve energy consumption of the system (See Section 7.1).

The architecture consists of three main elements:

1. sensors service – responsible for gathering data from sensors and performing initial preprocessing of them,
2. inference service – responsible for context based reasoning and knowledge management, and
3. working memory middleware – acting as an intelligent proxy between sensors service and the inference service.

The *Sensor Service* gathers data directly from mobile device sensors. Due to the different possible sensor types (GPS, Accelerometer, Bluetooth), different methods for interpreting these data are required. Hence, each sensor has its own interpreter module that is responsible for initial preprocessing of the raw data. Data preprocessing is triggered by the Working Memory Middleware.

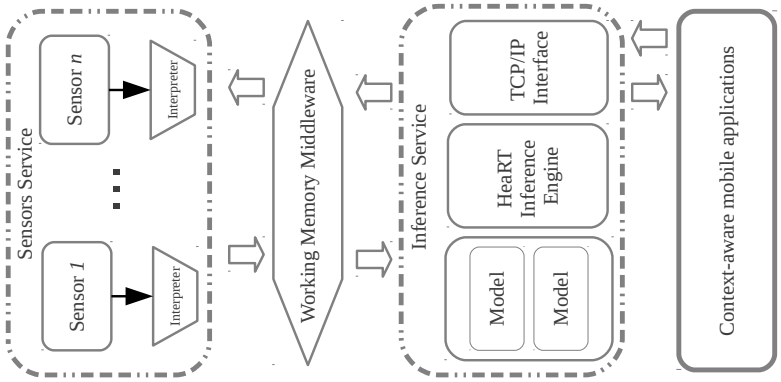

Fig. 4. Architecture of the mobile context aware framework

The *Inference Service* is responsible for performing reasoning, based on the model (knowledge base) and the working memory elements (facts). The service is capable of managing many models transparently switching between them. The reasoning task is performed by HeaRT. It is a lightweight rule-based inference engine that uses XTT2 [44] notation for knowledge representation. It is written in Prolog and can be installed on mobile device together with tuProlog [10] interpreter, providing autonomous inference service. Moreover, the HeaRT inference engine, in contrary to other rule-based reasoners, provides custom verification module that can be used for automatic optimization of a knowledge base. The inference service provides a TCP/IP interface for context-aware applications that may query HeaRT for important information.

The *Working Memory Middleware* is responsible for exchanging information between sensors service and inference service. The working memory is shared between all models stored within the inference service, acting as a *knowledge cache*. Therefore, it minimizes the number of required requests to the sensors service, improving power efficiency of the entire system.

The idea of separating Working Memory Middleware from the inference service is that it is able to learn sensors usage habits, and in consequence adapt itself to the individual device characteristic. It improves power efficiency of the system, since sampling rates are not fixed, but leaned from the usage patterns. The more details on that are presented in Section 7.1.

Most of the modeling languages discussed in Section 3 provides mechanisms for systems that are suppose to perform context identification tasks. To allow more advanced context reasoning, including context processing and derivation of new information based on actual context, we argue that rules are the best choice. The context aware framework presented in this section uses the XTT2 [44] notation for knowledge representation. It is a visual knowledge representation method for rule-based systems [37] where rules are stored in tables connected with each other creating a graph. XTT2 has a textual representation called HMR. An example of a rule written in HMR language is:

```
xrule Today/1: [day in [sat,sun]] ==>[today set weekend].
```

[10] See `http://alice.unibo.it/xwiki/bin/view/Main/`.

The rule is referenced in Figure 7, in table *Today*. The HMR representation is used by the HeaRT inference engine, which provides several reasoning techniques (inference modes), including: *Data-Driven* which finds all possible information that can be inferred from given data, and *Goal-Driven* which finds only a specific information that can be inferred from a specific subset of XTT2 tables. These inference modes allow the efficient reasoning in structured knowledge bases. Only the tables that lead to desired solution are fired, and no rules are fired without purpose, making the inference process less resource-consuming. Detailed description of the inference algorithms for XTT2 rule bases, can be found in [39].

HeaRT inference engine provides a callback mechanism that allows to query external sources for information. The external source could be: database, user, or in our case working memory middleware and any other source of data available to device. Callbacks are associated with attributes, defined as e.g.:

```
xattr [ name: day, class: simple, type: day, comm: in,
        callback: [ask_working_memory,[day]] ].
```

The *comm* element in the attribute definition determines behavior of a callback if it pushes or pulls the value of the attribute to/from an external source. More details about the callback mechanism can be found in [42].

7. Practical Evaluation

A prototype implementation of the approach has been provided for the Android 4 platform. Its two main aspects are described below.

7.1. Working memory middleware

We implemented a prototype of Working Memory Middleware that learns user habits based on the usage of device sensors (in this case a GPS sensor). This allows automatically adjust sampling rate of sensors form sensor service, depending on the probability of sensor usage, and thus for minimizing energy consumption of the system.

We used trigonometric approximation and logistic regression machine learning algorithm to learn sensor usage. For the number of m observations the algorithm will take as an input two vectors of length $m - X$ and Y. X_i is the time of i-th observation (a continuous value in range of $< 0; 24)$ and corresponding value:

$$Y_i = \{^{-1\ for\ inactive\ state}_{1\ for\ active\ state}$$

A function describing probability in time we will call a hypothesis. Desired characteristics of a hypothesis function $h(t)$ are: (1) continuous, (2) defined in range $< 0; 24 >$ corresponding to time of a day, (3) values contained in range $< 0; 1 >$ - given its a probability

We can introduce another helper function $F(t)$, closely resembling what can be found in trigonometric approximation, mainly linear combination of independent trigonometric functions.

$$F(w,t) = \omega_0 + \sum_{i=1}^{n} \left(\omega_{2i+1} \cos\left(\frac{i * t * 12}{\Pi} \right) + \omega_{2i+2} \sin\left(\frac{i * t * 12}{\Pi} \right) \right)$$

We are now ready to define our hypothesis function $H(t)$ as: $H(w,t) = \theta(F(w,t))$, where θ is defined as sigmoid function of a following form:

$$\theta(x) = \frac{1}{1+e^{-x}}$$

With problem posed in that way we will search for a probability density function $h(t)$ which most likely produced the learning data X, Y. Our overall goal will be to maximize the combined probability of all observations over vector of parameters w:

$$\max_{w} \prod_{i=0}^{m-1} P(Y_i|X_i)$$

Maximizing an expression is equivalent to maximizing its logarithm:

$$\min_{w} -\frac{1}{m} \ln \prod_{i=0}^{m-1} P(Y_i|X_i)$$

Thus we have reduced the problem to a well known issue of convex optimization, solved easily by such methods as gradient descent:

$$\min_{w} \frac{1}{m} \sum_{i=0}^{m-1} \ln \frac{1}{Y_i * \theta(F(w, X_i))}$$

The results of approximation for a GPS sensor usage, based on data collected during one week is shown in Figure 5.

We made experiments on two identical devices carried by the same person. Our working memory approach allowed for 50% battery saving than in case of the device without the algorithm implemented. Figure 6 presents battery level over time for these two cases.

7.2. Inference service

We are prototyping light version of HeaRT reasoner, that will be working on an mobile device as a local inference service. The architecture of HeaRT inference engine is designed to allow communication via TCP/IP protocol and callback system. The callbacks system is used to communicate with Working Memory Middleware. It is energy efficient and allows for fast, immediate access to sensor data. Communication between the inference service and context-aware applications is realized via TCP/IP protocol. This allows to make all the reasoning, and sensor data interpretation transparent to the engineer who designs context-aware application. The inference service is based on an HeaRT rule-based engine. This allows for more complex reasoning than just a simple classification tasks provided by most of the methods presented in Section 3.

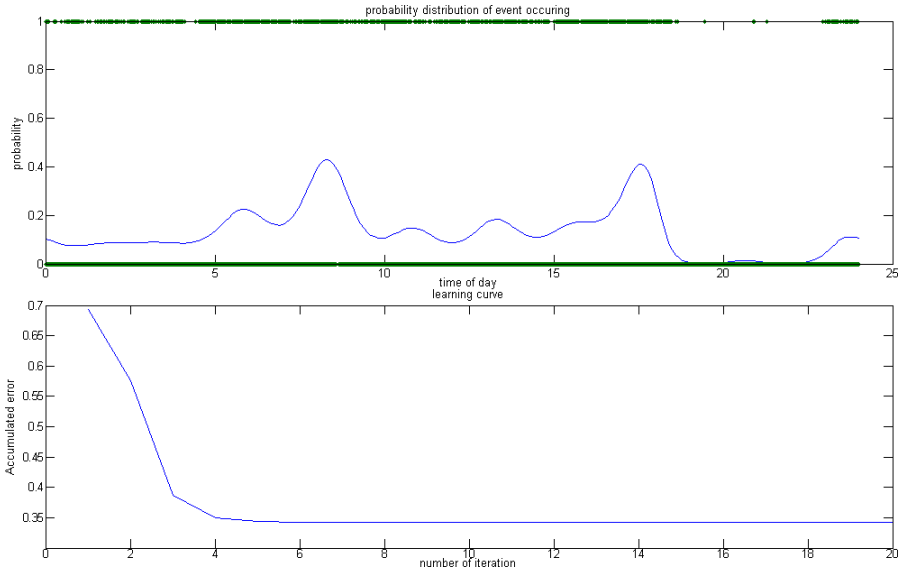

Fig. 5. Prediction of GPS sensor usage based on samples from five days.

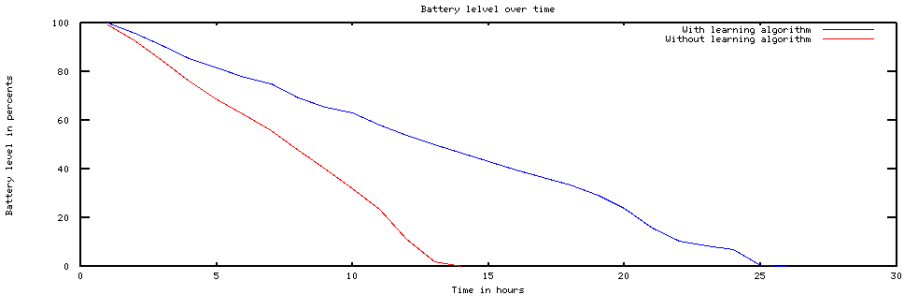

Fig. 6. Difference in power consumption for device with and without learning algorithm implemented.

8. Usecase Scenarios

In this section we introduce two use cases, for the proposed architecture. The first is a social system supporting crime detection and investigation that has been extended for mobile devices. The second one, which is a work in progress, includes the adaptation of social applications for mobile devices to explore the flexibility given by opportunistic network infrastructures.

8.1. Social Threat Monitor

Social Threat Monitor (STM) is a system developed to build a semantically enriched environment for collaborative knowledge management [2]. Using it, local communities

are able to share information about road traffic dangers and threats, i.e. closed roads, holes in the pavements and streets, dangerous districts or events that impede a normal traffic. STM was aimed to be a community portal that allows citizens to participate and cooperate in order to improve the security in the urban environment.

Being in a given situation, and location the user can be automatically notified by their mobile device about the relevant threats and the situation. Relevance to the person may be related to their role defined in the STM system, as well as the context, e.g. a person who walks should not be bothered by warnings relevant only to drivers in the same location. The use of data fusion from the sensors and multimodal interfaces of the mobile device allows to limit the amount of data the user would have to provide to the system. In fact, we propose a major paradigm shift on the front-end side. Whereas the original interface of STM was mostly query-based, here we propose a push-based UI where the user is automatically notified only about the information relevant to him. The system automatically uses the context data from the mobile device, as well as the data acquired from the STM server to perform reasoning for the user.

We proposed an context-aware enhancement to STM based on the context-aware architecture presented in this paper. Detailed description of the system was described in [8]. An exemplary XTT2 model (see Figure 7) presented in this section allows to alert users about threats in a context-aware way. The system takes into consideration spatial (localization of the user) and temporal (time of a day) contexts, as well as user activities. This allow the intelligent threats filtering. For instance, the model will prevent from warning a user who is driving a car about threats that are applicable to pedestrians only. This is achieved by selecting only these rules that are valid in current context.

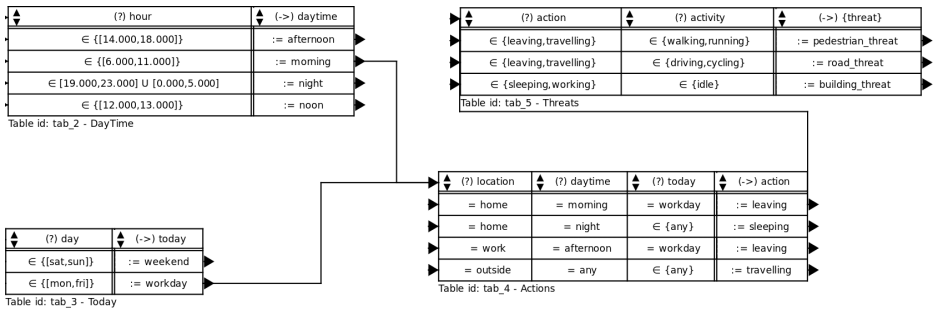

Fig. 7. Example of the XTT2 rule model for a mobile threat monitor

Information about threats is fetched from Social Threat Monitor system via callbacks using the STM WEB API (see [2] for details). Information about user localization, time of a day, and user activities is pulled from a working memory middleware via callback mechanism. The working memory middleware obtains this information from sensors interpreters (for example: location from GPS sensor interpreter, activity like walking or running from accelerometer sensor interpreter, etc.).

Taking into consideration an example from Figure 7, and assuming that it is Monday, 8 a.m., and the user is driving a car, the system will process following rules: (1) rule 1

from *DayTime* table, (2) rule 2 from *Today* table, (3) rule 4 from *Actions* table, (4) and rule 2 from *Threats* table.

This inference chain will trigger several callbacks, including one that is assigned to `road_threats` in the *Threats* table. The callback will fetch all road threats from Social Threat Monitor system that are located near the user and assign it to `road_threats` attribute.

The application that implements Mobile Threat Monitor interface will be able to pull all information about threats from inference service via TCP/IP API and display it to the user. On the other hand, the system could exploit the social aspects of mobile users, to automatically reason on threats and feed STM with these data. This could be done by monitoring users activity on social networks portals (like Tweeter, Facebook, Google+) and interpreting data gathered there. for instance, if a lot of tweets are generated about a shooting, robbery, or any other incident in some location, the system could automatically report these incidents to STM and alert appropriate services and other users. The application could also report anomalies in user behavior to the STM system. When similar anomaly will be reported by many users, an alert will be raised. For instance, if a number of users start running from a building, there is probably a fire.

8.2. Opportunistic Networks

Opportunistic networks emerge as a networking paradigm that leverages both the use of various communication interfaces on mobile device and the mobility of users. The advantages of opportunistic networks include potentially high capacity, low cost, localized communication, full decentralization, and independence of any network infrastructure [9]. Therefore, opportunistic networks have been considered as an appealing and critical alternative solution when traditional cellular networks are unavailable or inaccessible; for example, a crowded social event, or disaster-affected areas [57]. However, the lack of stable topology of opportunistic networks brings challenges in disseminating, sharing, and collaborating on network resources. For example, what information should be shared with which user will depend on the relation between the information and the user (context and semantics), the relation between and the information producer and the user (social sensing and privacy), the prediction of the user's future movement and the understanding of the social implication of mobility (social context and mobility analysis), the status (memory space, battery level, or communication bandwidth) of the user's device (context).

Taking that into consideration we propose a mobile, context aware system based on the architecture presented in this paper, that will take an advantage of all the contextual information such as: social relationships, user profiles with respect to the Internet content, and prediction of the user's future activity, to allow for improvement of routing algorithms in opportunistic networks. The real challenge in this approach will be automatically building and reasoning on user profiles, defined as user interests in the network content. The human interaction through the Web generates both implicit and explicit knowledge. Explicit, however unstructured knowledge may be contained in articles on Wikipedia, writing blogs or tweets etc. An example of an implicit contribution is a correlation between a search query that the user posted on the Web and the explicit knowledge that the user found valuable to him or her within search results. Such information may allow for modelling of user interests.

The Web is searched every day for various data from many different domains. However, it is possible to distinguish groups of users that usually search for information in similar sources, or on similar topic. For instance, it is highly probable that computer researchers will be more interested in websites containing scientific papers in computer science, or technical documentation than in websites containing information about ancient history. Thus, the information about the group that user belongs to, can be a significant factor in deciding which content he or she will be interested in in the future. A similar approach can be found in [28], where a mobile user context is gathered dynamically from its social network.

Working memory middleware discussed in Section 6, can not only improve power efficiency of the system, but also help predicting future sensor readings. By analyzing the sensor usage profile we are able to predict where the user will be in a nearest future, what social context will he be in (crowded places, not crowded areas) and what communication protocols (Wifi, Bluetooth, GSM) will be available to him. We can assume that a user with WiFi enabled, whose usage profiles assumes browsing the news websites in a nearest future, will have lower bandwidth priority than a person who does not have Wifi connection, depend only on Bluetooth ad-hoc transfers and his usage profiles assume video streaming. Such context fusion can be a valuable information for the inference service, that can help in determining the Internet connection quality in the nearest future. This information combined with the knowledge about user content interest (profiles) can be used to infer bandwidth priorities for opportunistic network routing algorithms. Finally, the security profiles of the different users on the network could also be represented by rules, see [43].

9. Summary and Future Work

In this paper we presented an overview of common approaches for modeling context, and we compared them according to certain critical factors. This led us to the conclusion that most of the languages available nowadays offer limited support for advanced inference tasks. To allow more advanced context reasoning, including context processing and derivation of new information based on actual context, we argue that rules are the the best choice.

Most common context-based architectures were also presented and compared according to such factors as: (1) energy efficiency, (2) privacy issues, (3) resource consumption, (4) responsiveness. We argue that most of the existing solutions are not fully applicable to mobile architectures. Hence, a new original approach for modeling and developing context-aware mobile applications was presented in the paper. The presented framework is designed as a hybrid of service oriented architecture and intelligent middleware architecture that includes: *inference service*, that uses HeaRT inference engine to provide on-line efficient reasoning, preserving the factors (2), (3) and (4); *working memory middleware*, that works as an intelligent knowledge cache that is able to learn sensor usage habits and adapts itself to individual device usage patterns. It minimizes the number of required requests to the sensors service, improving power efficiency of the entire system, preserving (1) and (2); *sensor service*, that is responsible for gathering and initial preprocessing of the raw sensor data and though providing high responsiveness of the system (4). A prototype implementation of the discussed architecture was conducted, with sev-

eral services modeled. Using it two use cases were presented, including a social system supporting crime detection and investigation.

The framework presented in this paper may be extended for additional functionalities. The first direction concerns intelligibility, since a rule-based system have a high capabilities of self-explaining their decisions. According to Dey [18], this is crucial factor of the system usability. The framework could provide mechanisms that will allow explaining its decision and asking user for corrections. The second one is related to the modelling of context with the use of rules combined with business processes, see [41,33,30]. In such a case processes could provide menas to handle the history of context and its processing. Another is related to the automatic model optimization, since HeaRT inference engine provides a verification plug-in that allows detecting anomalies such as: rules subsumption, redundancy and contradiction. Hence, a mechanism that will perform automatic optimization of an existing XTT2 model could be implemented [40]. An important part of the practical development will be concerd with the portability of the framework for different mobile platforms.

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer Publishing Company, Incorporated, 1st edn. (2011)
2. Adrian, W.T., Ciężkowski, P., Kaczor, K., Ligęza, A., Nalepa, G.J.: Web-based knowledge acquisition and management system supporting collaboration for improving safety in urban environment. In: Dziech, A., Czyżewski, A. (eds.) Multimedia Communications, Services and Security: 5th International Conference, MCSS 2012: Kraków, Poland. May 31-June 1, 2012. Proceedings. Communications in Computer and Information Science, vol. 287, pp. 1–12 (2012)
3. Atzmueller, M., Hilgenberg, K.: Towards capturing social interactions with sdcf: An extensible framework for mobile sensing and ubiquitous data collection. In: Proc. 4th International Workshop on Modeling Social Media. ACM Press (2013)
4. Bardram, J.: The java context awareness framework (jcaf) Ð a service infrastructure and programming framework for context-aware applications. In: Gellersen, H.W., Want, R., Schmidt, A. (eds.) Pervasive Computing, Lecture Notes in Computer Science, vol. 3468, pp. 98–115. Springer Berlin Heidelberg (2005)
5. Bellavista, P., Corradi, A., Fanelli, M., Foschini, L.: A survey of context data distribution for mobile ubiquitous systems. ACM Comput. Surv. 44(4), 24:1–24:45 (Sep 2012)
6. Benerecetti, M., Bouquet, P., Bonifacio, M., Italia, A.A.: Distributed context-aware systems (2001)
7. Biegel, G., Cahill, V.: A framework for developing mobile, context-aware applications (2004)
8. Bobek, S., Nalepa, G.J., Adrian, W.T.: Mobile context-based framework for monitoring threats in urban environment. In: Multimedia Communications, Services and Security: 6th International Conference, MCSS 2013: Kraków, Poland. June 6-7, 2013. Proceedings (2013), accepted
9. Boldrini, C., Conti, M., Delmastro, F., Passarella, A.: Context- and social-aware middleware for opportunistic networks. J. Network and Computer Applications 33(5), 525–541 (2010)
10. Bui, H.H., Venkatesh, S., West, G.: Tracking and surveillance in wide-area spatial environments using the abstract hidden markov model. Intl. J. of Pattern Rec. and AI 15 (2001)
11. Chen, H.: An Intelligent Broker Architecture for Pervasive Context-Aware Systems. Ph.D. thesis, University of Maryland, Baltimore County (2004)
12. Chen, H., Finin, T.W., Joshi, A.: Semantic web in the context broker architecture. In: PerCom. pp. 277–286. IEEE Computer Society (2004)

13. Chen, H., Perich, F., Finin, T.W., Joshi, A.: Soupa: Standard ontology for ubiquitous and pervasive applications. In: 1st Annual International Conference on Mobile and Ubiquitous Systems (MobiQuitous 2004), Networking and Services, 22-25 August 2004, Cambridge, MA, USA. pp. 258–267. IEEE Computer Society (2004)
14. Chronis, I., Madan, A., Pentland, A.S.: Socialcircuits: the art of using mobile phones for modeling personal interactions. In: Proceedings of the ICMI-MLMI '09 Workshop on Multimodal Sensor-Based Systems and Mobile Phones for Social Computing. pp. 1:1–1:4. ICMI-MLMI '09, ACM, New York, NY, USA (2009)
15. Coutaz, J., Crowley, J.L., Dobson, S., Garlan, D.: Context is key. Commun. ACM 48(3), 49–53 (2005)
16. De Virgilio, R., Torlone, R.: Modeling heterogeneous context information in adaptive web based applications. In: Proceedings of the 6th international conference on Web engineering. pp. 56–63. ICWE '06, ACM (2006)
17. Dey, A.K.: Understanding and using context. Personal Ubiquitous Comput. 5(1), 4–7 (Jan 2001)
18. Dey, A.K.: Modeling and intelligibility in ambient environments. J. Ambient Intell. Smart Environ. 1(1), 57–62 (Jan 2009)
19. Dey, A.K.: Providing architectural support for building context-aware applications. Ph.D. thesis, Atlanta, GA, USA (2000), aAI9994400
20. Eagle, N., (Sandy) Pentland, A.: Reality mining: sensing complex social systems. Personal Ubiquitous Comput. 10(4), 255–268 (Mar 2006)
21. Etter, R., Costa, P., Broens, T.: A rule-based approach towards context-aware user notification services. In: Pervasive Services, 2006 ACS/IEEE International Conference on. pp. 281 –284 (june 2006)
22. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. Int. J. Hum.-Comput. Stud. 43(5-6), 907–928 (Dec 1995)
23. Gu, T., Pung, H.K., Zhang, D.Q., Wang, X.H.: A middleware for building context-aware mobile services. In: In Proceedings of IEEE Vehicular Technology Conference (VTC (2004)
24. Henricksen, K., Indulska, J.: Developing context-aware pervasive computing applications: Models and approach. Pervasive Mob. Comput. 2(1), 37–64 (Feb 2006)
25. Hu, B., Wang, Z., Dong, Q.: A modeling and reasoning approach using description logic for context-aware pervasive computing. In: Lei, J., Wang, F., Deng, H., Miao, D. (eds.) Emerging Research in Artificial Intelligence and Computational Intelligence, pp. 155–165. Communications in Computer and Information Science, Springer Berlin Heidelberg (2012)
26. Hu, H., of Hong Kong, U.: ContextTorrent: A Context Provisioning Framewrok for Pervasive Applications. University of Hong Kong (2011)
27. Jaroucheh, Z., Liu, X., Smith, S.: Recognize contextual situation in pervasive environments using process mining techniques. J. Ambient Intelligence and Humanized Computing 2(1), 53–69 (2011)
28. Jung, J.J.: Contextualized mobile recommendation service based on interactive social network discovered from mobile users. Expert Systems with Applications 36(9), 11950–11956 (2009)
29. Jung, J.J.: Service Chain-based Business Alliance Formation in Service-oriented Architecture. Expert Systems with Applications 38(3), 2206–2211 (2011)
30. Jung, J.J.: ContextGrid: A Contextual Mashup-based Collaborative Browsing System. Information Systems Frontiers 14(4), 953–961 (2012)
31. Jung, J.J.: Cross-lingual Query Expansion in Multilingual Folksonomies: a Case Study on Flickr. Knowledge-Based Systems 42, 60–67 (2013)
32. van Kasteren, T., Krose, B.: Bayesian activity recognition in residence for elders. In: Intelligent Environments, 2007. IE 07. 3rd IET International Conference on. pp. 209–212 (2007)
33. Kluza, K., Maślanka, T., Nalepa, G.J., Ligęza, A.: Proposal of representing BPMN diagrams with XTT2-based business rules. In: Brazier, F.M., Nieuwenhuis, K., Pavlin, G., Warnier, M.,

Badica, C. (eds.) Intelligent Distributed Computing V. Proceedings of the 5th International Symposium on Intelligent Distributed Computing – IDC 2011, Delft, the Netherlands – October 2011, Studies in Computational Intelligence, vol. 382, pp. 243–248. Springer-Verlag (2011)
34. Ko, K.E., Sim, K.B.: Development of context aware system based on bayesian network driven context reasoning method and ontology context modeling. In: Control, Automation and Systems, 2008. ICCAS 2008. International Conference on. pp. 2309 –2313 (oct 2008)
35. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
36. Liao, L., Patterson, D.J., Fox, D., Kautz, H.: Learning and inferring transportation routines. Artif. Intell. 171(5-6), 311–331 (Apr 2007)
37. Ligęza, A., Nalepa, G.J.: A study of methodological issues in design and development of rule-based systems: proposal of a new approach. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1(2), 117–137 (2011)
38. Loke, S.W.: Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective. Knowl. Eng. Rev. 19(3), 213–233 (Sep 2004)
39. Nalepa, G., Bobek, S., Ligęza, A., Kaczor, K.: Algorithms for rule inference in modularized rule bases. In: Bassiliades, N., Governatori, G., Paschke, A. (eds.) Rule-Based Reasoning, Programming, and Applications. Lecture Notes in Computer Science, vol. 6826, pp. 305–312. Springer Berlin / Heidelberg (2011)
40. Nalepa, G., Bobek, S., Ligęza, A., Kaczor, K.: HalVA - rule analysis framework for XTT2 rules. In: Bassiliades, N., Governatori, G., Paschke, A. (eds.) Rule-Based Reasoning, Programming, and Applications. Lecture Notes in Computer Science, vol. 6826, pp. 337–344. Springer Berlin / Heidelberg (2011)
41. Nalepa, G.J.: Proposal of business process and rules modeling with the XTT method. In: Negru, V., et al. (eds.) Symbolic and numeric algorithms for scientific computing, 2007. SYNASC Ninth international symposium. September 26–29. pp. 500–506. IEEE Computer Society, IEEE, CPS Conference Publishing Service, Los Alamitos, California ; Washington ; Tokyo (september 2007)
42. Nalepa, G.J.: Architecture of the HeaRT hybrid rule engine. In: Rutkowski, L., [et al.] (eds.) Artificial Intelligence and Soft Computing: 10th International Conference, ICAISC 2010: Zakopane, Poland, June 13–17, 2010, Pt. II. Lecture Notes in Artificial Intelligence, vol. 6114, pp. 598–605. Springer (2010)
43. Nalepa, G.J., Ligęza, A.: Designing reliable Web security systems using rule-based systems approach. In: Menasalvas, E., Segovia, J., Szczepaniak, P.S. (eds.) Advances in Web Intelligence. First International Atlantic Web Intelligence Conference AWIC 2003, Madrid, Spain, May 5-6, 2003. Lecture Notes in Artificial Intelligence, vol. 2663, pp. 124–133. Springer-Verlag, Berlin, Heidelberg, New York (2003)
44. Nalepa, G.J., Ligęza, A., Kaczor, K.: Formalization and modeling of rules using the XTT2 method. International Journal on Artificial Intelligence Tools 20(6), 1107–1125 (2011)
45. Olguin, D., Waber, B.N., Kim, T., Mohan, A., Ara, K., Pentland, A.: Sensible organizations: Technology and methodology for automatically measuring organizational behavior. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B: CYBERNETICS pp. 43–55 (2009)
46. Park, H.S., Oh, K., Cho, S.B.: Bayesian network-based high-level context recognition for mobile context sharing in cyber-physical system. IJDSN 2011 (2011)
47. Ranganathan, A., Al-Muhtadi, J., Campbell, R.H.: Reasoning about uncertain contexts in pervasive computing environments. IEEE Pervasive Computing 3(2), 62–70 (Apr 2004)
48. Ranganathan, A., Campbell, R.H.: An infrastructure for context-awareness based on first order logic. Personal Ubiquitous Comput. 7(6), 353–364 (Dec 2003)
49. Ranganathan, A., McGrath, R.E., Campbell, R.H., Mickunas, M.D.: Use of ontologies in a pervasive computing environment. Knowl. Eng. Rev. 18(3), 209–220 (Sep 2003)

50. Schilit, B.N., Adams, N., Want, R.: Context-aware computing applications. In: Proceedings of the Workshop on Mobile Computing Systems and Applications. pp. 85–90. IEEE Computer Society (1994)
51. Wang, H., Mehta, R., Chung, L., Supakkul, S., Huang, L.: Rule-based context-aware adaptation: a goal-oriented approach. Int. J. Pervasive Computing and Communications 8(3), 279–299 (2012)
52. Wang, X., Zhang, D., Gu, T., Pung, H.K.: Ontology based context modeling and reasoning using owl. In: PerCom Workshops. pp. 18–22 (2004)
53. Want, R., Falcao, V., Gibbons, J.: The active badge location system. ACM Transactions on Information Systems 10, 91–102 (1992)
54. van Wissen, B., Palmer, N., Kemp, R., Kielmann, T., Bal, H.: ContextDroid: an expression-based context framework for Android. In: Proceedings of PhoneSense 2010 (Nov 2010)
55. Woensel, W.V., Casteleyn, S., Troyer, O.D.: A Framework for Decentralized, Context-Aware Mobile Applications Using Semantic Web Technology (2009)
56. Ye, J., Dobson, S.: Exploring semantics in activity recognition using context lattices. J. Ambient Intell. Smart Environ. 2(4), 389–407 (Dec 2010)
57. Ye, J., Dobson, S., McKeever, S.: Situation identification techniques in pervasive computing: A review. Pervasive and Mobile Computing 8(1), 36–66 (2012)

Grzegorz J. Nalepa is the corresponding author for this paper. He holds a position of assistant professor in AGH UST in Krakow, Poland and is a member of GEIST research team. His work is focused on design methods for intelligent systems, as well as knowledge representation and reasoning techniques. He formulated a new design approach for intelligent rule-based systems called XTT (eXtended Tabular Trees), as well as the Semantic Knowledge Engineering design and analysis methodology for knowledge-based systems. He has been involved in over 30 international conferences and workshops. Since 2008 he co-organizes the Knowledge and Software Engineering Workshop (KESE) at KI, the German AI conference.

Szymon Bobek holds a position of a research assistant at the AGH UST in Krakow, Poland. His main research interests are: knowledge representation, machine learning, ambient intelligence, and context awareness. He is the co-author of the HeaRT rule engine for the XTT rules.

Received: February 9, 2013; Accepted: November 15, 2013.