Resource-Bounded Context-Aware Applications: A Survey and Early Experiment

Ijaz Uddin[1], Hafiz Mahfooz Ul Haque[1], Abdur Rakib[1(✉)], and Mohamad Rafi Segi Rahmat[2]

[1] School of Computer Science, The University of Nottingham, Malaysia Campus, Semenyih, Malaysia
{khyx4iui,khyx2hma,abdur.rakib}@nottingham.edu.my
[2] School of Applied Mathematics, The University of Nottingham, Malaysia Campus, Semenyih, Malaysia
mohd.rafi@nottingham.edu.my

Abstract. The recent advancement of mobile computing technology and smartphones have changed the way we live, communicate, interact, and understand the world. Smartphones have various salient features that make them promising system platforms for the development of context-aware applications, e.g., embedded sensors in smartphones make them more convenient to be used for making context-rich information available to applications. Although the state of the art development of smartphones has endued developers to build advanced context-aware applications, many challenges still remain. Those are mostly due to the limited resources available in the mobile devices including computational and communication resources. This paper surveys the recent advances in context-aware applications in mobile platforms, and proposes a decentralized context-aware computing model that makes use of the smartphone platform, a P2P communication model, and declarative rule-based programming.

Keywords: Context-aware · Resource-bounds · Rule-based reasoning · Distributed reasoning · Android SDK

1 Introduction

The rapid growth of the cell phones across the world creates a platform for new computing systems. Within a decade the basic idea of cell phones has changed from a mobile phone towards a smartphone. Smartphones are capable to assist us in our daily routine tasks as well as providing basic communication features and connections to the wide services of the internet [3]. The exponential growth of smartphone products, softwares and communication ease have a tremendous effect on human lives. People use smart technology to connect and share experiences with each other including social networking, VoIP and other freely available messaging and call services [23].

P.C. Vinh et al. (Eds.): ICTCC 2016, LNICST 168, pp. 153–164, 2016.
DOI: 10.1007/978-3-319-46909-6_15

In recent years, smartphones are equipped with wide range of sensors. For example, the global positioning system (GPS), shake sensors, accelerometers, proximity sensors are now basic sensors to be found on such devices [13]. These kind of sensing devices generate enough information about the users, such as location, time, movement, and so on. That is, such devices and technologies could be used to sense the surrounding environment of a user, acquire user's contexts and act accordingly. While a suitable communication mechanism would help to enrich the user interaction with the application and between these devices [16]. In principle, context-aware computing systems which include multiple interacting devices and human users can often be usefully modelled as multi-agent systems. Non-human agents in such a system may be running a very simple program, however they are increasingly designed to exhibit flexible, adaptable and intelligent behaviour. A common methodology for implementing the latter type of agents is implementing them as rule-based reasoners. In the literature, ontology-based context-modeling and rule-based reasoning have already been used in the field of context-aware systems [6,8]. In previous work [18,20], we have shown that ontological approach is a good way for modeling context-aware systems, and it allows us to model context-aware systems as rule-based agents. In [18], we developed a logical model for resource-bounded context-aware multi-agent systems which handles inconsistent context information using non-monotonic reasoning, but we have not explored the practical implementation yet. Rule-based systems and traditional rule engines have found significant application in practice, though mostly for desktop environment where the resources are abundantly available compared to smartphone devices. The main issue with those engines is that they cannot be easily used on smartphones or resource bounded devices due to platform differences and different hardware profiles. Some rule engines, which are discussed in Sect. 4, have already been tested for porting into mobile environment but the results were not satisfactory or the porting were only partially successful. In view of the above, there is a need to develop a decentralized context-aware computing model that makes use of the smartphone platform, a suitable communication model and declarative rule-based programming as a preferred development language. By developing a pure smartphone compatible context-aware framework, any kind of domain specific context-aware applications can be developed, e.g., elder care system, hospital critical situation, traffic control and office security, among others.

The remainder of the paper is organized as follows. In Sect. 2, we briefly introduce context-aware computing and its limitations and challenges in resource-constrained settings. Section 3 presents a brief survey on general theoretical context aware resource bounded frameworks, while Sect. 4 focuses on discussing Android based theoretical and ported context aware frameworks. Section 5 presents the proposed solution which includes a framework design for a specific platform, a communication model along with a proposed Android based application for context-aware programming. Section 6 concludes and discusses some suggestions for future work.

2 Context-Aware Computing: Limitations and Challenges

In a survey [14], it has been revealed that many definitions of *context* in different views exist. According to Dey et al. [2]: *"Context is 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"*. In the same paper context-aware is defined as, *"A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task"*. Context can be further specified by its entities [24]. For example, a person's context entities can be his identity, location, mood and his surrounding, among others. The context-aware computing emerged in early 1990s when small mobile devices were introduced. In 1992, Olivetti Labs active badges used the infrared badger assigned to staff members for tracing their locations in office and according to the locations calls were forwarded [27]. Further developments in the context-aware systems lead to the development of various frameworks to support such systems including for example Georgia Tech's Context Toolkit [21]. In recent years, more research has been carried out and advanced context-aware systems exist [4], and the contributions to research and development over the years promise a bright future of such systems. Although various context-aware frameworks have been developed over the years, however, their functions remain primitive. This is because these systems are more complex due to the mechanism for sensing and reasoning about contextual information and their changes over time, and they often run on tiny resource-bounded devices in highly dynamic environments. Many challenges might arise when these context-aware devices perform computation to infer implicit contexts from a set of given explicit contexts and reasoning rules, and perhaps exchange information via messages. We list some constraints those often arise while designing and developing context-aware systems.

- **Space constraint:** The memory space available for storing contexts is often limited on most of the mobile devices. These devices usually are not privileged to store all the contexts acquired by itself or received via interaction with other agents.
- **Communication constraint:** Context-aware devices often acquire contextual information from other smart devices via messages. These devices communicate among themselves in a highly dynamic environment, they exchange information via message which causes quick reduction of the battery energy level. And most of the smart devices are not specifically designed to support this feature due to battery power constraints.
- **Time constraint:** Mobile devices often have limited computational power and simultaneous execution of multiple programs make this process more slower.

Due to the above constraints, execution of some applications on smart mobile devices may be impossible or may not be able to produce expected behavior, execution of some applications may need more energy or space than available in a mobile device.

3 Context-Aware Resource-Bounded Frameworks

Recent developments in the field of context-aware systems have led to a renewed interest in probing different approaches in developing different kinds of context models and reasoning techniques, which heightened the significance of the applications used by different resource-bounded mobile devices. In the literature [5,6,8,17], several approaches have been proposed for context modeling and reasoning techniques considering specific architectures and home health monitoring as an exemplar system. In [5], a ontology-based context management (GCoM) model is presented to facilitate context modeling and reasoning considering (user defined and ontological) rules and their semantics. This context modeling approach shows how a context can be acquired, manipulated, stored and expressed. This model is designed for dynamicity and re-usability in different domains where resource limitation is a crucial issue.

The authors in [8] present an ontology based programming framework for rapid prototyping of context-aware application development. The design goal of the authors is to support a wide user's category and cooperation and collaboration in the applications development. The framework further emphasizes that being a collaborative environment, users have to agree on shared conceptualization of the domain. The authors also targeted three categories of users based on their technical abilities into High level, Middle level, and Low level users who can use the framework in different environment. The main components of the framework are context providers, the context manager, programming toolkits and the resource sharing server. The framework although has various options to cater users from diverse technical skills, however the use of resource sharing server suggests limitation on distributed approach, and also the Android limitations demands a more compact and Android compatible framework.

In [17], an ontology-based framework has been presented to show how context-aware systems can be modeled as resource-bounded rule-based systems using OWL 2 RL and Semantic Web Rule Language (SWRL). Emphasize is given in the distributed problem-solving for the systems of communicating context-aware rule-based agents and specify bounds on time and number of messages exchanged to derive a goal. However, memory bound was not considered. Memory requirement is an important factor for reasoning because context-aware systems often run on resource limited devices. In [20], a logical framework $\mathcal{L}_{\mathcal{OCRS}}$ is presented for modeling and verifying context-aware multi-agent systems where agents reason using ontology-driven first order horn clause rules. In this model, authors have considered space requirement for reasoning in addition to time and communication resources. This work is based on monotonic reasoning where beliefs of an agent cannot be revised based on some contradictory evidence.

To some extent we believe that inconsistency may occur in the agent's memory and context-aware agents take decisions based on the available information that may become unreliable at certain circumstances. To overcome this issue, another framework [18] has been proposed for resource-bounded context-aware multi-agent systems which handles inconsistent context information using non-monotonic reasoning. The resulting logic $\mathcal{L}_{DROCS}$ allows a system designer to describe a set of rule-based non-monotonic context-aware agents with bounds on space and communication resources. In [19], it has been shown how to state various qualitative and quantitative properties of resource-bounded context-aware systems and formally verify resource requirements of such systems using model checking techniques.

Although the works discussed above make novel attempts to improve modeling resource-bounded context-aware systems, however the practical implementation has not yet been studied in depth.

4 Context-Aware Rule-Based Frameworks for Mobile Devices

There are quite a few frameworks that are specifically designed for the context-aware resource-bounded devices. The authors of [11] argue that there does not exist any comprehensive design and development tool which covers all the aspects of context aware applications in mobile platform including e.g., methodology, language, inference engine and communication protocols. They further state that such development environment is essential and will benefit both the researchers and developers. In an attempt to address some of these issues researchers tried to port the existing desktop based framework into mobile platform, e.g., in [22] authors proposed to port the JADE framework to Android, however still it is an ongoing project. Another attempt to port JADE in Android system by extending the JADE agent classes also shares the same problem of an earlier attempt [26] that it is not purely distributed and services are provided by a server which acts as a back end while the mobile devices act as the front end of the platform. From the development point of view, a general purpose programming language such as Python, is widely used among the complicated system development languages. The reason to mention Python is that there is a tool available which can convert a Python code into Android or its equivalent of iOS code. Although it works well at the basic code level, things get complex when a user wants access to program the internal hardware or sensors of the Android device. In order to make it works, there is a three step turn around required which will make it possible to use the sensors of the devices and it is out of the scope to be discussed here which also makes it least desirable as when new sensors are available their support may not be readily available. Some researchers tried to develop android based frameworks as discussed in [28] which provides a mechanism based on expression for Android. An expression is a Boolean, in which axioms are the context condition on the context entities. Although the work is based on the Android framework, the framework doesn't have its own language. Furthermore, instead of reasoning, various scenarios are monitored using the evaluators (== $, >=, >, <, <==$, regular expression, distance). The authors intend to provide distributed environment compatibility in their future work.

Based on the literature, there does not exist any platform specific framework where it is completely Android compatible. In all of the above mentioned frameworks, one has to use the traditional desktop computers for agent programming and then agents are made Android compatible. One such (Prolog based) project is also an ongoing research work called HeaRT [25], which is also in its early development stage and needs effort and time to see the product quality. Although the rule text representation is written in Prolog code, the author intends to change it and makes its own parser and semantic analyzer for writing rules in its own language.

5 Proposed Context-Aware Application Framework

To the best of our knowledge, various theoretical rule-based context-aware resource-bounded frameworks are available and some of them are discussed above, however so far they have not been implemented, tested or deployed. Nevertheless, there are some frameworks available that are ported from desktop to the Android framework where usability cannot be guaranteed, as they have problems associated with the frameworks' differences. They may be able to facilitate the currently available device sensors and architecture but in the future if the Android architecture design changes or new sensors are added then a user has to wait for the support to be available for the ported frameworks.

Based on the literature studied, we propose that there should be an independent framework for resource-bounded devices, which may not depend on the traditional desktop computers from any aspects. A user may be able to develop an agent on a device and run it on the device itself. The next sections further elaborate our idea and the proposed implementation framework, communication model and the application.

5.1 Intended Implementation Platform

In order to make a purely resource-bounded context-aware framework implementable, a specific platform from the leading platforms may be targeted. The state of the art available platforms are Android, iOS, Windows mobile, Black Berry and Symbian, among others. The frameworks used for the rule-based systems, to name some, are JADE, JARE JESS [15] and many more use platform independent Java for their frameworks and implementation. These also add the power of the Java programming into their rule-based platform. The downside of using Oracle Java is that it has some compatibility issues with the smart devices platforms such as Android or iOS which have their own development tools. The main language for Android development is using Google implementation of Java using the Android Software Development Kit (SDK) and the Android Studio. There are differences between Java programming for desktop systems and Java programming for the Android systems. As for the Java, the syntax of the language remains the same. The basic difference lies between their low level machine code generations. The desktop systems use JVM to translate Java code into machine understandable code or byte code while in Android system the applications are executed using the Dalvik Virtual Machine (DVM). DVM is a compact VM and is used in resource-bounded devices including cell phones, smart TVs and tablets, and is used to run programs on resource-bounded devices [9]. The Java language that can be used for Android does not support all the classes of Java as it has been optimized for the use with the resource bounded devices, so the obvious choice left is to use the Google Android SDK to program any application or software for the Android framework. Besides the focus of this study is for the resource bounded devices and the Android itself is made up resource bounded device and its programming is mainly done using the Google Android SDK, which can be used over Eclipse IDE or directly from the Google provided Android studio which is Officially supported by Google. According to a survey from Business Insider, the Google Android has the major user base as of 2015 report [1]. We chose the Google Android SDK to implement resource-bounded context-aware applications, however this choice does not restrict the research objective to Android only, and in the future we aim to develop a context-aware implementation framework that can be used to run application programs on multiple platforms seamlessly.

5.2 Proposed Communication Protocol: P2P

Since majority of the smart devices are wireless, there is also a need for efficient communication between devices whenever they interact with one another. Keeping in view the distributed scenario, one of the suitable protocols according to our proposed solution is Peer-to-Peer (P2P) protocol [12]. The P2P communication is a famous protocol widely available on the internet specially in the file sharing software like BitTorrent, Amule and Skype, among others. Skype uses it for their communication purpose with a slight changes. The P2P protocol is designed on the bases to provide a communication model without using any centralized server or computer, hence it is a natural fit for a distributed communication model where no central command is desired.

5.2.1 Open Peer

Open Peer is an open P2P signalling protocol. It is available for the Android as well as iOS platform. It lets the user find their peers by using various methods which include but not limited to social network websites, phone numbers, email ID and other ID authorization. It allows to communicate between two domains which are not on the same platform after providing strong identity validation. The communication mechanism is very secure and privacy is maintained. The open peer is designed to be stable on domains with high load of data. This protocol can use the signalling protocol in both encrypted and unencrypted formats. A message can be sent by any means of HTTPS/MLS/TLS/Message/RUDP/UDP or SCP protocols. Each request type must have its own ID and method of invocation and the same is for reply to include the sender ID. Another protocol that can be used for the same reason is Wifi-Peer to Peer, which provides almost all the functionalities that a standard protocol provides.

5.3 An Android-Based Context-Aware Application Example

The proposed solution is being developed for a complex context-aware application using the Google Android SDK, where smart devices sense the surrounding environments to acquire low level contexts and infer high level contexts based on the rules which are derived from a smart environment ontology domain [18,19]. In this system design, a number of agents (devices) have been considered which behave according to the use of the current contexts. The system is being implemented and run on smart devices and act as a dedicated system based on customized features. The communication among devices in based on messages passing and they exchange only facts (contexts). The application relies on rule-based systems where rules are distributed in a customized way and they don't change during execution. Each device has capability to produce implicit contextual information (derived facts) from given (or sensed) contexts. The core components of the application are elaborated in the following sections.

5.3.1 Internal Development Mechanism

The proposed model is based on the previous work [19], agents are programmed using defeasible rules and they use defeasible reasoning technique to infer implicit contextual information from given (or sensed) explicit contexts. The execution of rules accuracy also depends on the designer of rules, as rules have priority and the priorities are set by the domain expert. The proposed model is distributed, and every device acts as a module (see Fig. 1). We modularize the rule base for two reasons: (i) to reduce the amount of rules to be searched, and (ii) to make the rules specific to the role of a user e.g., patient or a doctor.

In order to load the initial facts the agents has pre-loaded facts in its static memory which cannot be changed. As these facts are the pre requisites for an agent to start its activity. As the agents are believed to have its inference so it will generate new facts from its inference and will store them accordingly. For that reason the dynamic memory is used which can be changed as required by the agent itself. But keeping the resources minimum, the dynamic memory is also limited and some facts can

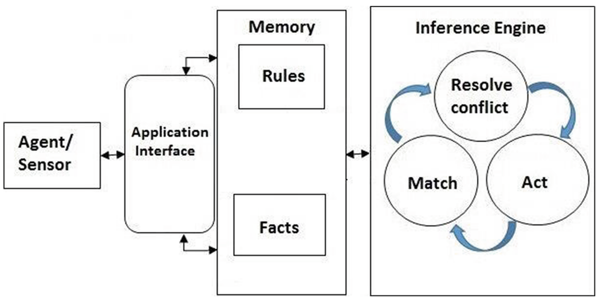

Fig. 1. System overview (single agent's perspective)

be stored. To overcome the problem of storing newly derived facts the logical model uses a mechanism to overwrite existing facts in two condition: (i) only if the memory is full to store new facts, and (ii) when a contradictory context arrives in the memory [19]. In the first case if there is no contradictory context residing in the memory, it will randomly overwrite any memory slot to store the newly derived context. While in the second case even if the memory is not full, it will check first for any contradictory context, if there exists a contradictory context it will be replaced with the new context.

It is pertinent to mention that in resource bounded devices where memory space is at a premium, the well known pattern matching RETE algorithm [7] may not be a good choice while implementing rule engines [10]. In our prototype implementation, the rules of an agent are logically divided into two categories, the rules that are fired frequently are stored separately from those which are fired rarely. The rarely fired rules are gathered by using a counter with every rule, if a rule counter increases frequently it will be stored in the frequently fired rules category. And the rules with very low counter rate can be stored in the simple rule set. Threshold cannot be determined as the frequency may vary from role to role and device to device. Instead, the difference is based on the counters associated with the rules. The reason for this division is to restrict the rule engine to traverse the rules which are more likely to fire, saving time in computation, and memory space.

5.4 The User Interface

The prototype of Android based experimental results are shown in Fig. 2. These user interfaces reflect the system behavior based on the set of horn-clause rules

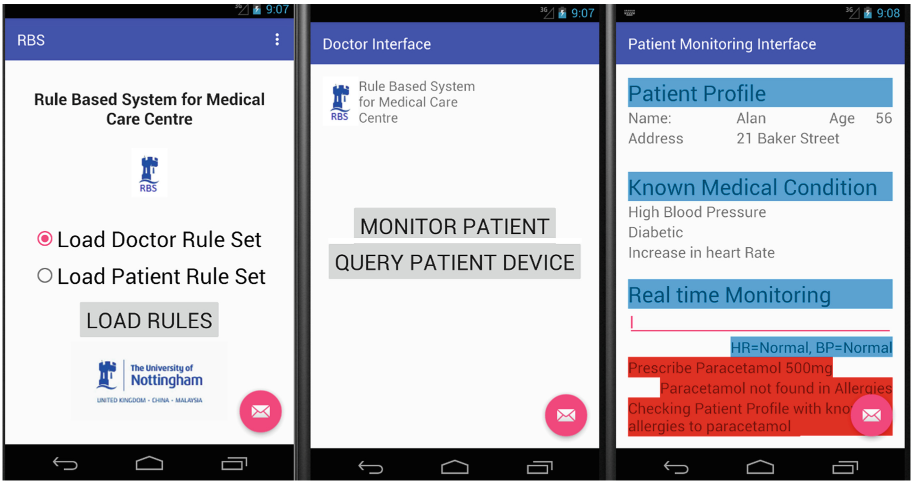

Fig. 2. Proposed application interfaces

distributed to the agents. As all the above mechanisms are running in the background, a good interface is as important as the code efficiency. We have developed an Android prototype that upon start ask for the role of either a doctor or a patient. Upon choosing one it will install the rules associated with the role selected and proceed accordingly. The step by step snapshots are provided in Fig. 2. The end product would be flexible enough to implement intelligent behaviour and at the same time this would work efficiently with limited processing power and on memory bounded devices. Hence, the proposed system is first verified using formal verification techniques before its implementation.

6 Conclusions

In this paper, we surveyed context-aware resource-bounded frameworks from both theoretical and practical points of view. The survey shows that purely Android platform based context-aware resource-bounded frameworks are not available yet. We propose a concrete Android platform based solution for context-aware resource-bounded systems, which makes use of the smartphone platform, a P2P communication model and rule-based programming language. In the future work, we will study the social impacts of state-of-the-art prototypes, which will be tailored to the specific needs of exploration of state-of-the art technologies to improve human lives; and particularly be suitable for development of systems for smart spaces.

References

1. Android is the world's largest mobile platform but it has to overcome these massive hurdles to keep the lead - business insider, October 2015. http://www.businessinsider.my/how-android-is-biggest-mobile-platform-ecosystem-google/?r=US&IR=T#uuUfWCcZ8WDUhJTg.97
2. Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: Gellersen, H.-W. (ed.) HUC 1999. LNCS, vol. 1707, pp. 304–307. Springer, Heidelberg (1999). doi:10.1007/3-540-48157-5_29
3. Ballagas, R., Borchers, J., Rohs, M., Sheridan, J.G.: The smart phone: a ubiquitous input device. IEEE Pervasive Comput. **5**(1), 70–77 (2006)
4. Bardram, J.E., Nørskov, N.: A context-aware patient safety system for the operating room. In: Proceedings of the 10th International Conference on Ubiquitous Computing, pp. 272–281 (2008)
5. Ejigu, D., Scuturici, M., Brunie, L.: An ontology-based approach to context modeling and reasoning in pervasive computing. In: Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007, pp. 14–19. IEEE (2007)
6. Esposito, A., Tarricone, L., Zappatore, M., Catarinucci, L., Colella, R., DiBari, A.: A framework for context-aware home-health monitoring. In: Sandnes, F.E., Zhang, Y., Rong, C., Yang, L.T., Ma, J. (eds.) UIC 2008. LNCS, vol. 5061, pp. 119–130. Springer, Heidelberg (2008). doi:10.1007/978-3-540-69293-5_11

7. Forgy, C.L.: Rete: a fast algorithm for the many pattern/many object pattern match problem. Artif. Intell. **19**(1), 17–37 (1982)
8. Guo, B., Zhang, D., Imai, M.: Toward a cooperative programming framework for context-aware applications. Pers. Ubiquit. Comput. **15**(3), 221–233 (2011)
9. Jackson, W.: Android Apps for Absolute Beginners, 3rd edn. Apress, Berkeley (2014). ISBN13: 978-1-484200-20-9
10. Kim, M., Lee, K., Kim, Y., Kim, T., Lee, Y., Cho, S., Lee, C.G.: Rete-adh: an improvement to rete for composite context-aware service. Int. J. Distrib. Sens. Netw. **2014**, 1–11 (2014)
11. Nalepa, G.J., Bobek, S.: Rule-based solution for context-aware reasoning on mobile devices. Comput. Sci. Inf. Syst. **11**(1), 171–193 (2014)
12. Park, H., Izhak-Ratzin, R., van der Schaar, M.: Peer-to-peer networks - protocols, cooperation and competition. In: Streaming Media Architectures, Techniques, and Applications: Recent Advances. IGI Global (2010)
13. Pei, C., Guo, H., Yang, X., Wang, Y., Zhang, X., Ye, H.: Sensors in smart phone. In: Li, D., Liu, Y., Chen, Y. (eds.) CCTA 2010, Part II. IFIP AICT, vol. 345, pp. 491–495. Springer, Heidelberg (2011). doi:10.1007/978-3-642-18336-2_59
14. Perera, C., Zaslavsky, A.B., Christen, P., Georgakopoulos, D.: Context aware computing for the internet of things: a survey. IEEE Commun. Surv. Tutorials **16**(1), 414–454 (2014)
15. Petcu, D., Petcu, M.: Distributed jess on a condor pool. In: Proceedings of the 9th WSEAS International Conference on Computers, pp. 1–5 (2005)
16. Raento, M., Oulasvirta, A., Petit, R., Toivonen, H.: Contextphone: a prototyping platform for context-aware mobile applications. IEEE Pervasive Comput. **4**(2), 51–59 (2005)
17. Rakib, A., Faruqui, R.U.: A formal approach to modelling and verifying resource-bounded context-aware agents. In: Vinh, P.C., Hung, N.M., Tung, N.T., Suzuki, J. (eds.) ICCASA 2012. LNICST, vol. 109, pp. 86–96. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36642-0_9
18. Rakib, A., Haque, H.M.U.: A logic for context-aware non-monotonic reasoning agents. In: Gelbukh, A., Espinoza, F.C., Galicia-Haro, S.N. (eds.) MICAI 2014, Part I. LNCS (LNAI), vol. 8856, pp. 453–471. Springer, Heidelberg (2014). doi:10.1007/978-3-319-13647-9_41
19. Rakib, A., Haque, H.M.U.: Modeling and verifying context-aware non-monotonic reasoning agents. In: Proceedings of the 13th ACM-IEEE International Conference on Formal Methods and Models for System Design, pp. 453–471. IEEE (2015)
20. Rakib, A., Ul Haque, H.M., Faruqui, R.U.: A temporal description logic for resource-bounded rule-based context-aware agents. In: Vinh, P.C., Alagar, V., Vassev, E., Khare, A. (eds.) ICCASA 2013. LNICST, vol. 128, pp. 3–14. Springer, Heidelberg (2014). doi:10.1007/978-3-319-05939-6_1
21. Salber, D., Dey, A.K., Abowd, G.D.: The context toolkit: Aiding the development of context-enabled applications. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 434–441. ACM, New York (1999)
22. Sartori, F., Manenti, L., Grazioli, L.: A conceptual and computational model for knowledge-based agents in android. In: WOA@ AI*IA 2013, pp. 41–46 (2013)
23. Schrittwieser, S., Frühwirt, P., Kieseberg, P., Leithner, M., Mulazzani, M., Huber, M., Weippl, E.R.: Guess who's texting you? evaluating the security of smartphone messaging applications. In: 19th Annual Network and Distributed System Security Symposium (2012)

24. Sehic, S., Nastic, S., Vögler, M., Li, F., Dustdar, S.: Entity-adaptation: a programming model for development of context-aware applications. In: Proceedings of the 29th Annual ACM Symposium on Applied Computing, pp. 436–443. ACM (2014)
25. Slazynski, M., Bobek, S., Nalepa, G.J.: Migration of rule inference engine to mobile platform. Challenges and case study. In: Proceedings of 10th Workshop on Knowledge Engineering and Software Engineering (KESE 2010) co-located with 21st European Conference on Artificial Intelligence (ECAI 2014), Prague, Czech Republic (2014)
26. Ughetti, M., Trucco, T., Gotta, D.: Development of agent-based, peer-to-peer mobile applications on android with jade. In: The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, UBICOMM 2008, pp. 287–294. IEEE (2008)
27. Want, R., Hopper, A., Falcão, V., Gibbons, J.: The active badge location system. ACM Trans. Inf. Syst. **10**(1), 91–102 (1992)
28. van Wissen, B., Palmer, N., Kemp, R., Kielmann, T., Bal, H.: ContextDroid: an expression-based context framework for android. In: Proceedings of the International Workshop on Sensing for App Phones (PhoneSense) 2010, pp. 1–5 (2010)