ROLE OF MIDDLEWARE FOR INTERNET OF THINGS: A STUDY

Soma Bandyopadhyay, Munmun Sengupta, Souvik Maiti and Subhajit Dutta

Innovation Lab, TATA Consultancy Services Ltd. Kolkata, India
soma.bandyopadhyay@tcs.com, munmun.sengupta@tcs.com,
s.maiti@tcs.com, subhajit.dutta@tcs.com

ABSTRACT

Internet of Things (IoT) has been recognized as a part of future internet and ubiquitous computing. It creates a true ubiquitous or smart environment. It demands a complex distributed architecture with numerous diverse components, including the end devices and application and association with their context. This article provides the significance of middleware system for (IoT). The middleware for IoT acts as a bond joining the heterogeneous domains of applications communicating over heterogeneous interfaces. First, to enable the better understanding of the current gap and future directions in this field a comprehensive review of the existing middleware systems for IoT is provided here. Second, fundamental functional blocks are proposed for this middleware system, and based on that feature wise classification is performed on the existing IoT-middleware. Third, open issues are analyzed and our vision on the research scope in this area is presented.

KEYWORDS

Internet of Things, middleware, semantic model, context-awareness, ubiquitous computing.

1. INTRODUCTION

Internet of Things (IoT) is a combined part of future Internet and ubiquitous computing. It demands interactions with the heterogeneous raw sensors, aggregators, actuators and diverse domain of context aware applications, preserving the security and privacy. It comprises two definite components Internet and things. Internet is a global network infrastructure with self configuring, scalable, dynamic expansion capabilities based on standard and interoperable communication protocols whereas "things" are physical objects/devices or virtual-objects/devices/information having identities, physical attributes, and virtual personalities and use intelligent interfaces. "Things" are heterogeneous in nature and seamlessly integrated into the information network.

In order to meet the above said demand IoT will require a software platform defined as middleware, fundamentally providing abstraction to applications from the things, and offering multiple services. Development of middleware in the domain of IoT is an active area of research. There have been a lot of researches towards building up this middleware addressing interoperability across heterogeneous devices serving diverse domains of applications, adaptation, context awareness, device discovery and management, scalability, managing a large data volumes and, privacy, security aspects of the said IoT environment. Therefore there is a strong need to understand how the existing IoT-middleware systems work and address the different requirements of ubiquity as well as IoT [14] and [15], and most importantly the existing issues and gaps.

In this article focus has been given to study the existing IoT-middlewares, understanding its functional components and categorizing and comparing them as per the various features. The

DOI : 10.5121/ijcses.2011.2307

basic functional building blocks of IoT-middleware are proposed and discussed. Based on this review existing issues and gaps, and future research scope are also analyzed and presented.
The remainder of this article is organized as follows. First, the related work in IoT-middleware is presented, followed by descriptions of the essential functional blocks and the system architecture of the IoT-middleware system. The feature wise classification of theirs along with the different interfaces and syntax and semantics strategies are described in detail in the next section. The final section concludes this article with future research scope, and analyzes the gaps of the existing IoT-middleware system.

2. RELATED WORK

The common goal of all the middleware development initiatives is to develop a framework which can enable an adaptation layer in a plug-n-play mode. In recent past, many reviews have been made on different middleware, and on their basic features to support the important needs of the respective domains.

Various kinds of middlewares based on their supported functionalities like adaptability, context-awareness and application domains like Wireless Sensor Network (WSN), Radio Frequency Identification (RFID) are studied. The surveys performed in [4] and [5] have studied the middleware based on context-awareness feature. The survey in [4] is based on the architectural aspects and provides taxonomy of the features of a generic context-aware middleware. Survey reported in [5] evaluates several context-aware architectures based on some relevant criteria from ubiquitous or pervasive computing perspective. In [6] middleware for WSN has been reviewed and a detailed analysis of the approaches and techniques offered by the middleware to meet the requirements of the WSN has been presented. It also discusses generic components of the middleware and reference model of WSN based middleware. In [7], middleware for WSN has been classified depending on their approaches, which can be database oriented, tuple space approaches, event based and service discovery based approaches [13]. It also narrates the challenges of WSN middleware and provides some suggestions for solving them. In [8] middleware has been surveyed from adaptability perspective. This review also presents taxonomy for adaptive middlewares, their application domains and provides details for one of each middleware categories.

The survey presented in this article lists the overall features supported by the existing middleware in the IoT domain, with the proposal of the functional components and system architecture of IoT-middleware, and establishes the importance of middleware in IoT, whereas the rest of the surveys in this section depict how efficiently a particular feature is implemented in the middleware other than IoT domain. This is the basic difference between the present and the rest of the works. This article also analyses the issues and gaps on the existing middleware of IoT and explores the way forward on research scope.

3. FUNCTIONAL BLOCKS

Middleware for IoT is required for various reasons. The summary of reasons is as follows:

- Difficult to define and enforce a common standard among all the diverse devices belonging to diverse domain in IoT.
- Middleware acts as a bond joining the heterogeneous components together.
- Applications of diverse domains demand abstraction /adaptation layer.
- Middleware provides API (application programming interfacing) for physical layer communications, and required services to the applications, hiding all the details of diversity.

The above stated reasons generate the need for various functional components the IoT-middleware must support. The functional components of the middleware as proposed here are portrayed in the current section. The functional component of an IoT-middleware is depicted in Fig. 1. The inner most circle shows the required functional blocks. The second circle encompasses the further division of the functional blocks, and the outermost circle shows the important modules interacting with the various functional components, but not part of the middleware – example context processing, data storage and knowledge database. The functional components are as follows:

- Interoperation
- Context detection
- Device discovery and management
- Security and privacy
- Managing data volume

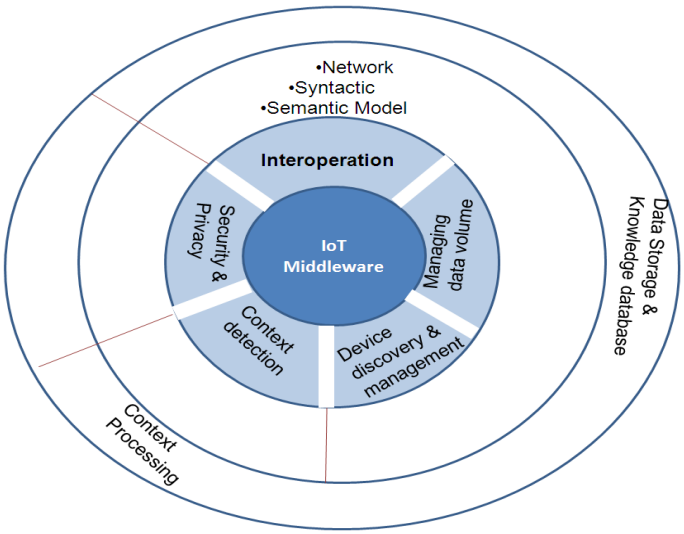

Fig. 1. Functional Components of IoT-Middleware

Functional components are discussed in the following subsections.

The system architecture is depicted in Fig. 2. It presents a layered view of the IoT middleware architecture. The fundamental layers are interface protocols, device abstraction responsible for providing interoperation, resolving the syntax and semantics associated with devices, central and management module is the core component, which performs the device discovery, management and context detection. Application abstraction module provides the interface with local and remote application. Local applications mainly run as event driven services. The other components like context analysis, knowledge database may reside in remote system essentially generating a need of distributed architecture.

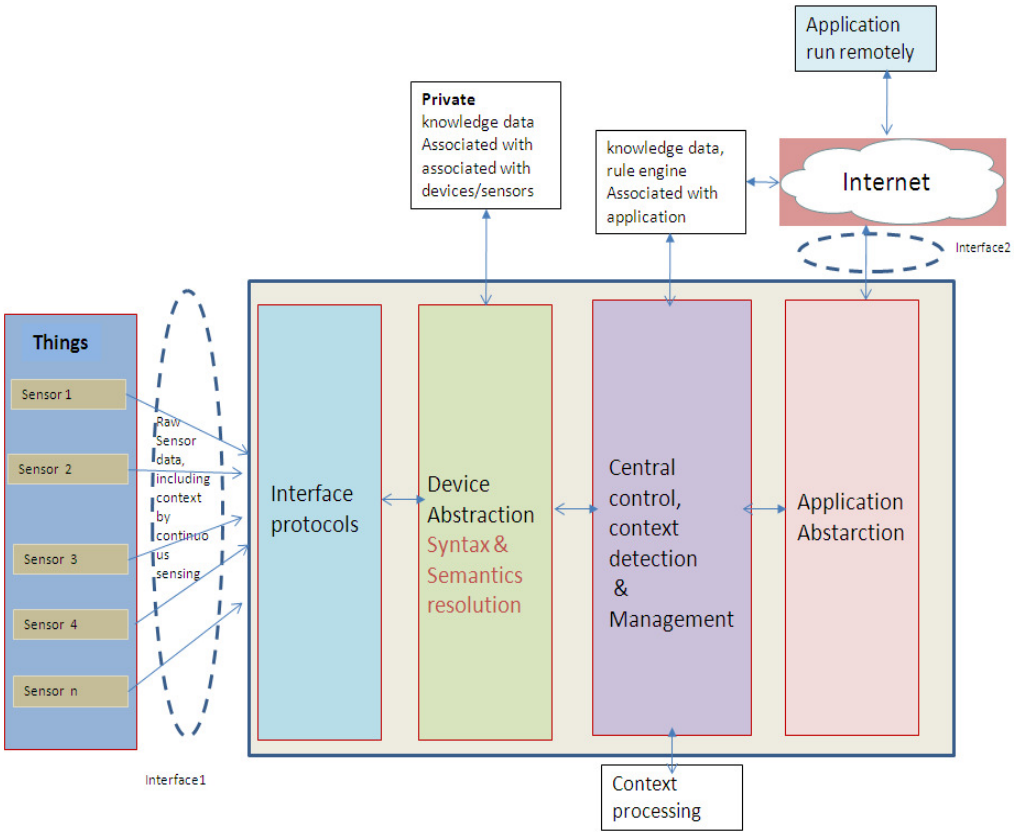

Fig.2. Functional Components of IoT-Middleware

3.1 Interoperation

Interoperation shares information and uses the same across diverse domains of applications using diverse communication interfaces. It can be further classified under three different categories like network, syntactic and semantics [11]. Network interoperation defines protocols for exchanging information among the various things across different communication networks, without considering the content of information. It covers the basic connectivity issues in physical and data-link to network, transport, session and sometimes application layer of TCP/IP stack. Syntactic interoperation deals with the format and structure of the encoding of the information exchanged among things. It includes the presentation and application of TCP/IP stack. Semantic interoperation defines the rules for understanding the meaning of the content of information, and creates a domain specific information model, known as semantic model.

IoT-middleware exposes multiple APIs to perform these interoperation functionalities. SOA-based architecture [9], [18] and Ubiquitous Service-Discovery Service (ubiSD-S) [22] are some of the common approaches adapted by the available IoT-middlewares for semantic interoperation whereas Home Audio/Video Interoperability (HAVi) [3] addresses network interoperation. IoT middleware particularly connecting the diverse sensors with the Internet uses sensorML (Sensor Model Language) [27] which provides standard models and XML schema for interoperating with sensors. This defines multiple models for various functionalities of the sensors, like actuation, aggregation, detection.

3.2 Context Detection

Context is responsible for characterizing the situation of an entity where an entity can be person, place, or object relevant to the interaction between a user and an application, including the user and applications themselves. IoT-middleware must be context aware for working into smart environments. Context awareness can be achieved by context detection and context processing. Context detection collects data and identifies the factors that have significant impacts on the response. Context processing extracts the context data, processes it and performs or takes decision based on that. Both these functional modules are depicted in Fig. 1. A knowledge database is required for setting up a closed feedback path between these blocks to evaluate the effectiveness of context-aware systems and make some possible improvements.

Among these, context-detection is a core component of IoT-middleware. The other two functional blocks residing outside the core components of IoT-middleware are mostly interacting with the IoT applications. Context detection and context processing have been implemented in different ways in existing IoT-middlewares.

Each service supported by the middleware [9] can be registered with a context-dependent identifier. The context-dependent identifier gets changed with the change in end-point mobility. The network manager would maintain a data-structure like IDTable with the information of identifiers [10].

The middleware described in [11] performs context detection by collecting information from the sensor devices and extracts required context data by utilizing available data mining algorithms. Another approach towards context detection is incorporating semantic context-aware multimodal visualization approach [1]. This will enable users not only to get just raw data, but also information relevant to a particular context and also visualization in an adequate way. This type of framework contains context-aware A2H (Agent-to-Human) interaction which is termed as 4i (For Eye) technology.

Optimized message communication between middleware users can be another notion of context awareness [12] where any event-response can first understand the intended recipient and can communicate among them.

3.3 Device Discovery and Management

Device discovery and management enables any device in the IoT network to detect all its neighbouring devices and make its presence known to each neighbour in the network. Device ontology [26] is used for storing information about the heterogeneous devices. From IoT perspective, these modules need to be reliable, fault-tolerant, adaptive and optimized for resource consumption [21]. Few techniques adopted for device discovery and management of device information are listed below:

Middleware described in [16], [8], [9], tries to extend the syntactic interoperability to semantic interoperability in application layer. This is done by combining the use of ontologies with semantic web services. Semantic Model Driven Architecture (Semantic MDA) is introduced to facilitate application development and to promote semantic interoperability for services and devices. It includes a set of models (Device ontology) and their usage in design time and run time. It introduces concept of semantic devices which are software representation of physical devices. Mapping of physical devices to semantic devices can be one-to-one or many-to-one depending on the application. Information and data about devices and device types are stored in device ontology. Semantic device description includes information regarding device capabilities, services, and device malfunction and security properties. Device Description includes information like device name, vendor details, hardware description and software description used to describe hardware and software resources of the device. Application

Ontology Manager provides interface for using Device Ontology. New devices can be included into the device ontology by adding sub-classes depending on specialized concepts and new properties. P2P (Point-to-Point) discovery is supported in various middleware systems. Middlewares described in [1] and [16] adopt this technique. As described in [12], peer-to-peer architecture is used in a system where devices support self-configurable services and scalability from tiny embedded devices to heterogeneous P2P networked systems.

Using device ontology, sensor nodes are abstracted as virtual sensor node in middleware described in [2]. The virtual abstracted unit contains metadata used for identification and discovery of sensor nodes. It also contains information regarding the data stream, functional properties related to lifecycle management etc. Concept of hardware abstraction layer which creates new adaptor when newer devices are added to the network is described in [13]. Agent based architecture is used in middleware [1] and [11].Here it is assumed that agents are associated with all devices present in the network. Middleware described in [1] follows IEEE FIPA model, including a Directory Facilitator in the system which maintains mapping between agents and their roles. It helps one agent to find other suitable agent/agents. This introduces severe bottleneck in the system. To improve the survivability of these systems, a significant part of DF knowledge gets stored in local knowledge databases of different platform agents when the system becomes fully operational. The combination of the local storages presents a kind of distributed directory. Peer-to-Peer (P2P) mechanism implemented on such a distributed directory is another mechanism complementary to central DF. Also, sometimes some of the services might not prefer to advertise themselves through the central DF for security reasons. P2P discovery can be the option in such scenario. The objective here is the design of mechanisms which will extend the scale of semantic resource discovery with P2P discovery. Such mechanisms have to enable an agents for discovering other agents playing a certain organizational role, to discover an agent /agents possessing certain needed information, and to discover resources (through its agents) of certain type or possessing certain properties. Here the request is sent to all or some of the agents on the contact list of the agent who wants to communicate. Those agents can forward the request to all/some of the agents on their lists, and so on.

Middleware in [11] includes agents for managing the repository of roles and scenarios. Agents are able to monitor data coming from the adapter about states of the resource, and take decisions depending on the data content. Agents also facilitate discovery of other agents in the environment. This approach facilitates service discovery, FIPA communication protocols utilization, and integration/composition of services. Agent-based layer includes an agent managing repository of roles and scenarios encoded in RDF-based Semantic Agent Programming Language (S-APL). Agents manage repository of atomic behaviours which are software components that agents can load depending on current scenario and the directory that facilitates flexible discovery of agents. S-APL [24] is a hybrid of semantics specification languages, semantic reasoners, and agent programming languages. It integrates the semantic description of domain resources with the semantic prescription of behaviour of agents in individual and collaborative level.

Service discovery based approach is used to find services available in the system and then correspondingly binding to that target provider Middleware described in [22] introduces a service discovery protocol called ubiquitous Service-Discovery Service (ubiSD-S) which provides dynamic, interoperable, context-aware service discovery. The middleware mentioned in [3] uses devices profile for Web Services (WS) which includes WS-Addressing, WS-Discovery, WS-Metadata exchange, and WS-Eventing for devices. The framework also includes device metadata management, and device lifecycle management containing information regarding the various devices in the network.

Monitoring and Inventory module of the middleware described in [18] includes sub-modules like Device Repository, Device Monitor, Middleware Historian and Discovery. Device Repository stores the static information about devices present in the network. For this purposes it clubs similar devices into Device Types. Device Type contains two fields: Properties and Features which describe common characteristics of a class of devices. Function of Device Monitor is to monitor devices and detect events like malfunction/faulty devices. As the devices are of varied nature, set of fault detection techniques are needed. These techniques can be driven by events generated by the devices or by invoking specific methods on the devices. New techniques can be used by implementing another monitoring method. Middleware Historian acts as an event archive for the architecture. Data from the devices (due to events or invocation), middleware system information, debugging messages, etc. can be stored there and are accessible for correlation of events. Discovery sub-module helps to find devices present on a network and retrieves information about them and their hosted services. Two functional modes are supported for discovery sub-module,"passive discovery" and "active discovery". The"passive discovery" listens to the network for announcements from new devices that connect to the network, dynamically retrieves their metadata and notifies the systems about the devices and their hosted services. The" active discovery", is responsible for dynamic search of specific devices or services, and is particularly suited for contexts where new devices with unknown capabilities continuously connect to the system.

3.4 Security and Privacy

Security and privacy are responsible for confidentiality, authenticity, and non-repudiation. Security can be implemented in two ways – (i) secure high-level peer communication which enables higher layer to communicate among peers in a secure and abstract way and (ii) secure topology management which deals with the authentication of new peers, permissions to access the network and protection of routing information exchanged in the network.

Security has been considered in all the functional blocks of the existing IoT-middleware, from the user level application to the various parts of the functional blocks [12] with a view to build a trustworthy design [9] and [10]. Other approaches to implement security and privacy in IoT-middleware are as follows. Middleware in [11] depicts the semantic ontology-based approach to build a universal trust management system, here trust descriptions are interpretable and processable by autonomous trust management procedures and modules, trust data should be given explicit meaning via semantic annotation. Semantic trust concepts and properties will be utilized and interpreted using common trust ontology. Trust information can be incorporated as part of semantic resource descriptions and stored in dedicated places within the platform. Communication and retrieval of trust information will be accomplished through corresponding agent-to-agent communication. Here agents represent communicating resources and required to be configured appropriately to handle all necessary trust management activities between the corresponding communication parties. Trust management procedures can be realized as a set of specific business scenarios in the form of agent configuration plans. Device authentication [18], integrity service and access control [2] are other techniques deployed for IoT.

3.5 Managing Data Volumes

Managing data volumes is an integral part of IoT-middleware. It is believed that there will be trillions of objects which will be part of this enormous network and hundreds of Exabytes [20] will be stored or exchanged among the objects. In other words there will be "Exaflood" or "Data deluge", i.e. explosion of the amount of data collected and exchanged. Therefore it is imperative to get novel methods to find, fetch, and transfer data. Here challenges involve in querying, indexing, process modelling, and transaction handling. These data can be identification data, positional data, environmental data, historical data and descriptive data as presented in [23].

That is why managing these voluminous data is an important module of IoT-middleware. In addition storage and knowledge database plays an assistive role to the module as depicted in Fig. 1. The data volume management of various IoT-middlewares is discussed in the following sections.

In [10] and [19] the Storage Manager module realizes the persistent storage and administration of information in the middleware. It can integrate any kind of storage into the middleware, address those storages as virtual devices and it stores the data as string. The main constraint of this middleware is that the storage device should also be running the same middleware. On the other hand, in [17] there is an existence of a RDBMS (Relational Data Base Management System). The Information Sharing Module of the middleware is responsible for collecting, filtering, storing and extracting queried data from the database.

The agents in middleware, as proposed in [11], acquire knowledge by utilizing available data mining algorithms and dynamically reconfigure the data management architecture on the basis of this knowledge. These agents infer (also collaboratively) new configuration plan based on this acquired knowledge. Whereas in [2], storage layer is in-charge of providing and managing persistent storage for data streams. Query processing is done by the Query Manager (QM) which includes the query processor and query repository for efficient management of the data. The notification manager handles the delivery of events and query-results to the registered clients.

3. CLASSIFICATION OF THE IOT-MIDDLEWARE

This section classifies the different IoT-middleware based on the various features like interoperation, device management, platform portability, context awareness, security and privacy, and the support of various interface protocols. Table 1 and Table 2 depict the classifications of various IoT-middleware systems based on the various features and interface protocol support respectively.

Table 1. IoT-middleware comparison.

IoT Middleware	Features of Middleware				
	Device Management	Interoperation	Platform Portability	Context Awareness	Security and Privacy
HYDRA	✓	✓	✓	✓	✓
ISMB	✓	×	✓	×	×
ASPIRE	✓	×	✓	×	×
UBIWARE	✓	×	✓	✓	×
UBISOAP	✓	✓	✓	×	×
UBIROAD	✓	✓	✓	✓	✓
GSN	✓	×	✓	×	✓
SMEPP	✓	×	✓	✓	✓
SOCRADES	✓	✓	✓	×	✓
SIRENA	✓	✓	✓	×	✓
WHEREX	✓	✓	✓	×	×

All the listed middlewares support device discovery and management. Context aware functionality is supported by HYDRA, UBIWARE, UBIROAD and SMEPP. On the other hand, SOCRADES, SMEPP, GSN, UBIROAD and HYDRA are some examples of middleware implementing security and user privacy in their architecture. Based on platform portability, syntactic resolution, HYDRA, SMEPP and ASPIRE are OSGi compliant, UBIROAD uses

JAVA and XML, UBISOAP uses J2SE and J2ME, GSN uses XML and SQL, SIRENA and SOCRADES use DPWS while SOCRADES also uses SAP NetWeaver [25] platform and ISMB uses any JAVA compliant platform. WhereX [28] is developed using J2EE architecture and is integrated with Oracle Application Server 10g.It also uses Rhino rule engine which is implementation of Java Script.

Table 2. IoT-middleware Interfaces.

IoT Middleware	Interface protocols				
	Zigbee	RFID	WiFi	Bluetooth	Sensor (others)
HYDRA	✓	✓	✓	✓	✓
ISMB	×	✓	×	×	✓
ASPIRE	×	✓	×	×	×
UBIWARE	×	✓	✓	×	✓
UBISOAP	×	✓	✓	×	✓
UBIROAD	×	✓	✓	✓	✓
GSN	×	✓	✓	×	✓ IEEE-1451
SMEPP	×	×	✓	✓	✓
SOCRADES	×	✓	×	×	✓
SIRENA	×	✓	×	✓	✓
WHEREX	✓	✓	✓	✓	✓

4. CONCLUSION

In this article the role of middleware system in IoT is presented. It has proposed the functional blocks of IoT-middleware, and discussed feature wise classifications among the existing IoT-middleware. It has also presented the system architecture of IoT middleware.

The current state-of-the-art of the middleware for IoT explores different approaches to support some of the functionalities to operate in IoT domain. But no one covers the full set of functionalities to meet the requirement of IoT-middleware as analyzed here for any smart or ubiquitous environment, except in [29].

Middlewares have several short comings or open issues. They are available for respective domains separately. ASPIRE, ISMB etc. address the RFID domain. GSN addresses the sensor networks in general. UBIROAD addresses smart vehicular systems. There exists no generic middleware which can be applicable across all possible smart environments- like smart home, smart vehicle, smart city[30] etc. including RFID domain, and can be customized as per the domain specific requirements. It has been observed from this study that to resolve scalability issues IPv6 is proposed but not yet resolved completely. Support for context detection and processing have not been achieved fully. Support of semantic modelling and managing of data volumes also fall in the open issues, particularly handling the crowd sourcing of diverse domain. There is a scope for research work in making a generic IoT-middleware system, which is applicable across all domains by making all the functional components reusable and can be added as add-on to the middleware system. The development of generic interfaces, as shown in interface1 and interface2 in Fig.2, are also open issues in this domain.

We are continuing to carry researches on the above mentioned open issues to design and develop a layer based IoT-middleware for any smart environment. One of the contributions of this paper consists in this discussion on the open issues on middleware for IoT and that can be combined to define the future research scope on IoT-middleware.

REFERENCES

[1] Katasonov, A., Kaykova, O., Khriyenko, O., Nikitin, S., Terziyan, V.: Smart Semantic Middleware for the Internet of Things. In: 5th Intl. Conf. Informatics in Control, Automation and Robotics (ICINCO'08), pp. 169--178. Volume ICSO, (2008)

[2] Aberer, K., Hauswirth, M., Salehi, A.: Middleware Support for the Internet of Things. In: 5th GI/ITG KuVS Fachgespr˝ach "Drahtlose Sensornetze", pp. 15--21. (2006)

[3] Bohn, H., Bobek, A., Golatowski, F.: SIRENA - Service Infrastructure for Realtime Embedded Networked Devices: A Service Oriented Framework for Different Domains. In: International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL'06), pp. 43. IEEE Computer Society, Washington, DC, USA (2006)

[4] Kjær, K., E.: A Survey of Context-Aware Middleware. In: 25th conference on IASTED International Multi-Conference: Software Engineering, pp. 148--155. ACTA Press (2007)

[5] Miraoui, M., Tadj C., Amar, C. B.: Architectural Survey of Context-Aware Systems in Pervasive Computing Environment. In: Ubiquitous Computing and Communication Journal, vol. 3, no. 3 (2008)

[6] Wang, M., M., Cao, J., N., Li, J., Das, S., K.: Middleware for Wireless Sensor Networks: A Survey. In: Journal of Computer Science and Technology, vol. 23, no. 3, pp. 305—326. (2008)

[7] Henricksen, K., Robinson, R. A Survey of Middleware for Sensor Networks: State-of-the-Art and Future Directions. In: International Workshop on Middleware for Sensor Networks, pp. 60-65. Melbourne, Australia, November (2006)

[8] Sadjadi, S. M., McKinley, P.: A Survey of Adaptive Middleware. Technical Report MSU-CSE-03-35,Computer Science and Engineering, Michigan State University, East Lansing, Michigan (2003)

[9] Eisenhauer, M., Rosengren, P., Antolin, P.: A Development Platform for Integrating Wireless Devices and Sensors into Ambient Intelligence Systems. In: 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops (SECON Workshops '09), pp. 1—3. (2009)

[10] Badii, A., Khan, J., R., Crouch, M., Zickau, S.: Hydra: Networked Embedded System Middleware for Heterogeneous Physical Devices in a Distributed Architecture, Final External Developers Workshops Teaching Materials. (2010)

[11] Terziyan, V., Kaykova, O., Zhovtobryukh, D.: UbiRoad: Semantic Middleware for Context-Aware Smart Road Environments. In: Fifth International Conference on Internet and Web Applications and Services (ICIW), pp. 295--302, Barcelona (2010)

[12] Albano et. al., M.: Towards Secure Middleware for Embedded Peer-to-Peer Systems: Objectives and Requirements. In: RSPSI, Innsbruck (2007)

[13] Gelernter, D.: Generative Communication in Linda. In: ACM Transactions on Programming Languages and Systems (TOPLAS), vol. 7, issue 1, New York (1985)

[14] Liu, D. -L. Y. F., Liang, Y. -D.: A Survey of the Internet of Things. In: The 2010 International Conference on Electronic-Business Intelligence (ICEBI) (2010)

[15] Atzori, L., Iera, A., Morabito, G.: The Internet of Things: A Survey. In: Computer Networks, vol. 54, issue 15, pp. 2787—2805. (2010)

[16] Vision and Challenges for Realising the Internet of Things, http://ec.europa.eu/information_society/events/shanghai2010/pdf/cerp_iot_clusterbook_2009.pdf (2010)

[17] ASPIRE Architecture and Middleware, http://wiki.aspire.ow2.org/xwiki/bin/download/Main/Services/06%20ASPIRE%20Architecture%20and%20Middleware.pdf

[18] Spiess, P., Karnouskos, S., Guinard, D., Savio, D., Baecker, O., Souza, L., Trifa, V.: SOA-Based Integration of the Internet of Things in Enterprise Services. In: IEEE International Conference on Web Services (ICWS), pp. 968—975. Los Angeles (2009)

[19] Reiners, R., Zimmermann, A., Jentsch, M., Zhang, Y.: Automizing Home Environments and Supervising Patients at Home with the Hydra Middleware: Application Scenarios using the Hydra Middleware for Embedded Systems. In: Proc. of the First International Workshop on Context-Aware Software Technology and Applications (CASTA), pp. 9—12. New York (2009)

[20] http://en.wikipedia.org/wiki/Exabyte

[21] Ahamed, S. I.,Zulkernine, M.,Anamanamuri, S.: A Dependable Device Discovery Approach for Pervasive Computing Middleware. In: First International Conference on Availability, Reliability and Security (ARES '06), pp. 66-73.Vienna (2006)

[22] Caporuscio, M., Raverdy, P. -G., Issarny, V.: ubiSOAP: A Service Oriented Middleware for Ubiquitous Networking. In: IEEE Transactions on Services Computing, IEEE computer Society Digital Library,(2010)

[23] Cooper, J., James, A., Challenges for Database Management in the Internet of Things. In: IETE Technical Review, vol. 26, no. 5, pp. 320--329. (2009),

[24] Katasonov, A., Terziyan, V.: Semantic Agent Programming Language (S- APL): A Middleware for the Semantic Web.In: IEEE International Conference on Semantic Computing, pp.504-511.Santa Clara (2008)

[25] SAP NETWEAVER, http://www.sap.com/platform/netweaver/components/index.epx

[26] FIPA Device Ontology Specification (Approved for Experimental, as on 2002/05/10), http://www.fipa.org/specs/fipa00091/XC00091C.pdf

[27] http://www.opengeospatial.org/standards/sensorml

[28] http://wendang.baidu.com/view/ad7040a1b0717fd5360cdc8a.html

[29] Soma Bandyopadhyay, Munmun Sengupta, Souvik Maiti, Subhajit Dutta "A Survey of Middleware for Internet of Things" CoNeCo 2011, Ankara, Turkey, June 26 - 28, 2011.

[30] http://www.sofia-project.eu/system/files/Unified+smart+city+environment+based+on+SOFIA.pdf

Authors

Soma Bandyopadhyay has more than 13 years of industry experience in the area of Embedded Systems, Digital Signal Processor, Protocol and Wireless Communications and ubiquitous computing. Since 2003 has been associated with Innovation lab of TATA Consultancy Services (TCS) as senior scientist. Presently her prime focus area is ubiquitous and sensor network and computation, wireless communication- 4G-next generation broadband wireless MAC and Physical layer design, development & research activity. She has contributed towards the IEEE standard body on behalf of TCS. At present she is leading the research and development activity in interoperability and adaptability aspects of ubiquitous computing. Academically she is an M.Tech & B.Tech in Computer Science & Engineering from the University of Calcutta, India. She did her graduation in Physics (Hons.) from the same university.

Munmun Sengupta is working as a researcher in Innovation Lab of TATA Consultancy Services, India, in ubiquitous computing domain, mainly in middlewares with having 7 years of Industry experience. She has worked on the development of various wireless mobile communication technologies like WCDMA, UMTS, LTE. She has received her Bachelor of Technology degree in Electronics and Communication engineering from Sikkim Manipal University of Health, Medical and Technological Sciences, India in 2004.

Souvik Maiti is working in the domain of media and informatics in TATA Consultancy Services, India, having 6 years of Industry experience. He has worked on various wireless communication technologies like LTE, WiMax, Bluetooth, 802.11n, and embedded platform development. He has received the Bachelor of Technology degree in Electronics and Communication engineering from the Sikkim Manipal University of Health, Medical and Technological Sciences, India and Diploma in Embedded System from the Jadavpur University in 2004 and 2005 respectively.

Subhajit Dutta is working in the domain of Performance Optimization in Business Framework in TATA Consultancy Services, having 3 years of experience. He has worked on IPv6, OSGi platform, and interoperation aspects of middleware. He has received the Bachelor of Technology degree in Computer Science and Engineering from West Bengal University of Technology in 2008.