2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing

Data Management for Internet of Things: Challenges, Approaches and Opportunities

Meng Ma
School of Electronics Engineering and Computer Science
Peking University, Beijing, China
mameng@pku.edu.cn

Ping Wang
National Engineering Research Center for Software Engineering
Peking University, Beijing, China
pwang@pku.edu.cn

Chao-Hsien Chu
College of Information Sciences and Technology
The Pennsylvania State University, University Park PA, USA
chu@ist.psu.edu

***Abstract*—Internet of Things (IoT) is an important part of the new generation information technology. Data management for IoT plays a crucial role in its effective operations and has become a key research topic of IoT. Much work has been done to enable effective and intelligent data processing and analysis when IoT is evolving from Radio Frequency Identification (RFID), Wireless Sensor Network (WSN) and other related technologies. In this paper, we start from the core definition and architecture of IoT, aiming at examining current research effort to derive a holistic view of existing literatures. We present a layered reference model for IoT data management and elaborate the related research topics and solutions in each layer. Based on our analysis, we identify research challenges and opportunities for future work.**

Keywords—Internet of Things; data management; data cleaning; event processing; data storage; big data; middleware

I. INTRODUCTION

With the rapid development of sensing, communications, and analytics technologies, information is being generated, collected, managed and analyzed automatically rather than followed the traditional information processing process. Emerging technologies such as Wireless Sensor Network (WSN), Radio Frequency Identification (RFID), and Complex Event Processing (CEP) make real-time accurate perception, monitoring and reaction to the physical world possible. The extension of physical world networking and development of information systems have promoted a new type of network, called Internet of Things (IoT).

IoT refers to an Internet-like structure designed to connect anything for pervasive object identification, data capture and intelligent information processing at any time, in anywhere and for anyone [1][2]. This intelligent identification, positioning, tracking, monitoring and management system has been put into applications in various fields [3]. As shown in Figure 1, we suggest a layered architecture of IoT which is consisted of four layers.

- *Sensing Layer* is a connection between physical and cyber worlds. It contains various devices for information collection such as RFID reader, infrared sensor, GPS system, monitoring camera, etc. Ubiquitous perception is an important feature for what IoT is different from other networks. With the help of sensing layer, IoT can achieve a real-time monitoring and management on the objects properties or behavior patterns.
- *Network Layer* is the media to convey the collected data to upper layers for further processing and higher-level abstraction. The usage of IPv4/IPv6 based Internet, 3G/4G wireless networks and private networks provides IoT a seamless control over pervasive objects.
- *Middleware Layer* is a crucial part in IoT architecture. IoT middleware is an effective integration of several major functions including communications management, devices management, data processing, semantic reasoning, information security and service interfaces. IoT middleware provides service interfaces for applications allowing users to get rid of the trivial details in sensing and network layers.
- *Application Layer* is established based on the three lower layers and provides domain-oriented IoT applications for end-users in various application domains.

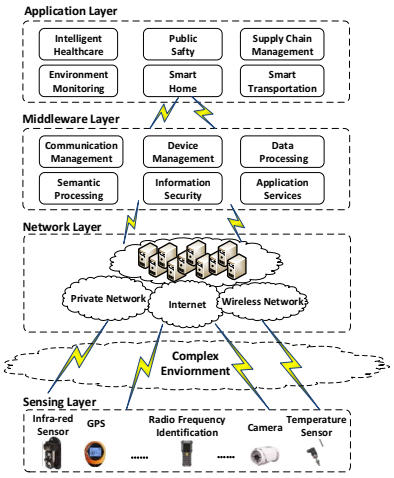

Figure 1. Layered Architecture of IoT

According to this layered architecture, how to manage data effectively and intelligently is the core research challenge in IoT development. IoT data management involves the process of data collection, processing, storage and analysis. The first and primary purpose is to collect (and clean) accurate and reliable data from multiple sources and manage heterogeneous data structures. Secondly, high-level information needs to be extracted, abstracted and inferred from raw data for decision supports. Eventually, how to store and analyze large amount of data is also a key research issue for IoT applications.

Much work has been done to enable effective and intelligent data processing and analysis when IoT is evolving from RFID, WSN and other related technologies. There are several papers surveyed related research in various aspects. Different IoT data types and characteristics are surveyed in

978-0-7695-5046-6/13 $26.00 © 2013 IEEE
DOI 10.1109/GreenCom-iThings-CPSCom.2013.199

the view of database management [4]. Challenges and approaches have also been concluded from the perspective of data-centric and energy-efficient respectively [5][6]. However, the overall management structure and key topics such as CEP and middleware research have not been closely examined. Although there are studies on the IoT data management framework [7][8], the detailed approaches not analyzed. Therefore, a comprehensive survey and analysis on this topic is still needed. In this paper, we review and examine current research effort to derive a holistic view of related works, from which we identify research challenges and discuss possible directions for future work.

This paper is organized as follows: We analyze the data characteristic in IoT systems and then present a reference model for IoT data management in Section II. Section III details the analysis on existing research for IoT data management and also the research opportunities. We conclude the research and the future works in Section IV.

II. IoT Data Characteristics and Management Reference Model

In this section, we start with an analysis of IoT data characteristic and then present a data management reference model for IoT.

A. Data Characteristics

In general, IoT Data shares five distinct characteristics:

- *Heterogeneity*: IoT uses different properties in various data types to describe the status of the "things". These data types vary from integer to character, also including semi-structured and unstructured data such as audio and video streams [9].
- *Inaccuracy*: One of the primary factors limiting the wide spread adoption of IoT is the inaccuracy of the data produced [10]. For example, experiments show that RFID systems can only capture 60% to 70% correct data caused by unreliable readings which leads to difficulties for direct use [11]. The same situation exists in most other sensing technologies.
- *Massive Real-Time Data*: IoT is designed to connect thousands of objects in large scale. Communications between different entities in dynamic networks generate a large volume of heterogeneous data in the form of real-time, high-speed, uninterrupted data streams. Scalable storage, filtering and compression schemes are essential for efficient big data processing [12].
- *Implicit Semantics*: Unprocessed IoT data is of low-level with weak semantics [13]. In order to support higher-level applications, such as smart home and intelligent healthcare, complex semantics need to be abstracted in event-driven perspective from the mass of low-level data.

B. The IoT Data Management Reference Model

According to these mentioned characteristics, we present a three-layered reference model for application design (see Figure 2). The function of each layer is described as follows:

- *Data Cleaning Layer*: As the reliability of current IoT systems is far from practical usage, data cleaning is essential for the correct interpretation and analysis of IoT data. The primary target of this layer is to diminish the data unreliability. Given the enormous volume of information, heterogeneous sources of errors, and real-time processing requirements, it can be a challenging task.
- *Event Processing Layer*: Cleansed IoT data is of high reliability and usability. But raw data only provides simple information, which is not directly related to business processes. The event processing layer inferences and transforms raw data into higher-level business logic. Semantic-based CEP technique is applied for this purpose which will be explained in Section III.
- *Data Storage and Analysis Layer*: Having the data in the form of business logics, IoT systems should provide services in various aspects and store them with proper schema. Such a schema should compress the amount of data and support analysis in a variety of hierarchies.

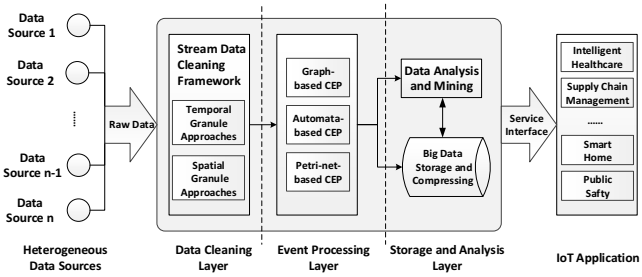

Figure 2. IoT Data Management Reference Model

In the rest of this paper, we will survey the existing research, classify them according to these three layers and analyze the challenges and opportunities for future work.

III. Challenges and Approaches of IoT Data Management

In this section, we proceed with typical issues encountered in each layer and the corresponding approaches. With in-depth analysis, we identify the research opportunities for each issue.

A. Data Cleaning Layer

The concept of data cleaning was firstly introduced in date warehouse research, which aims to clean up incomplete, erroneous and duplicated data, finally to solve the data quality issues in database or file system [14]. However, unlike database or file system, the unreliability of data produced by IoT sensing devices, namely "dirty data", manifests itself in four general forms: *False Negative*, *False Positive*, *Invalid* and *Redundancy* [15]. False negative refers to the data loss in IoT sensing devices. The reason for this problem includes interference signal between sensors and complex environment. False positive is also known as noise. Besides the correct data, system may also collect additional noise which is not expected. It is also due to the factors of the sensor itself and communication environment. This will cause another problem called invalid data showing the value deviates from the normal range. In order to ensure the signal coverage, there are usually more than one sensor covers the same object. This policy causes the data redundancy. Data

cleaning technology in IoT has been widely studied. Approaches to reduce and eliminate data quality issues include stream data cleaning framework, temporal granule and spatial granule approaches.

1) Stream Data Cleaning Framework

A data cleaning framework refers to an organizational structure of data cleaning approaches in different processing layer and perspectives. A flexible pipeline structure to clean sensor data, named ESP (Extensible Sensor stream Processing), is proposed in [16]. ESP is consisted of five stages: *Point*, *Smooth*, *Merge*, *Arbitrate* and *Virtualize*. The stage of Point filters the single data value for the quality problem like outlier. Smooth and Merge stages are designed to process data respectively in temporal and spatial granule. Logic fault will be eliminated in stage Arbitrate and the data from multiple sources are fused in the state of Virtualize. This framework provides an efficient and flexible organization of data processing. However, how to select the specific temporal and spatial granule for different application scenarios remain an unsolved challenge. Because the algorithm cost and performance of different cleaning approaches have not been considered in ESP, a cleaning framework for minimizing computational cost is proposed in [17]. This cost-conscious framework is based on the idea of classifier which defines cleaning/error cost for RFID data cleaning and results in a decision tree with minimum cleaning cost from machine learning tests. Although these data cleaning frameworks for IoT can organize different algorithms effectively from different aspects, they are based on sequential algorithm combination. Bagging and boosting approaches in data mining can be introduced into framework research for a better performance [18].

2) Temporal Granule Approach

Temporal granule data cleaning approach is mainly based on the concept of time window smoothing which is first proposed in [19]. As illustrated in Figure 3, the idea of the algorithm is to output a reading at the end of a time slot if there exists at least one positive reading in this window.

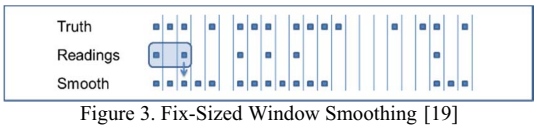

Figure 3. Fix-Sized Window Smoothing [19]

Although the fixed-sized window smoothing is effective and simple for implementation, it cannot ensure the data completeness and dynamics at the same time. To solve this problem, SMURF introduces an adaptive window scheme by viewing data streams as a statistical sample of readings [20]. Many improving works on SMURF are presented in rapidly movement object readings cleaning [21], high confidence node weighting [22] and proximity readers based statistical cleaning [23]. Temporal granule approaches can be mapped into the stage of *Smooth* in ESP framework. It can clean data based on the statistical characteristics. Although SMURF and its improving algorithms achieve a good balance between completeness and dynamics, temporal granule approach is a low-level clean-up mechanism. A higher level data cleaning requires additional spatial and logic information to achieve better results.

3) Spatial Granule Approach

Spatial granule approach can be classified as *Merge* stage in ESP framework. Its main purpose is to clean and aggregate data from different sources. A distributed RFID system architecture based on the inter-reader communications with secure channel is proposed in [24]. Similarly, as shown in Figure 4, IoT system deployment can be divided into regions. Inter-region communications can provide important information and supports for spatial granule data cleaning approaches.

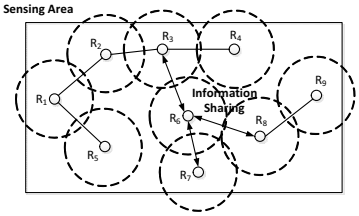

Figure 4. Distributed Model based on Inter-Region Communication [24]

P2P Collaboration is an RFID data cleaning method acting like the behavior of P2P network based on the inter-region communication architecture [25]. Through information sharing with forward and afterward nodes, the sensing node can judge and correct the fault with the help of surrounding nodes. Similarly, TopK-PDI [26] and bSpace [27] removes duplications and false positive data utilizing the data sharing among RFID readers. Unfortunately, this architecture has a higher requirement on inter-device communication capabilities and system topology control.

4) Summary

Table 1 summarizes major IoT data cleaning approaches and their characteristics.

Table 1. Classification of IoT Data Cleaning Approaches

	Target	Granule	ESP Stage	Features
ESP	FN,FP,I,R	T,S	P,S,M,A,V	Comprehensive Framework
Cost-Conscious	FN,FP	T	P,S,M	Minimum Cleaning Cost
Fix-Sized Window	FN,FP	T	P,S	Simple and Effective
SMURF	FN,FP	T	P,S	Balance Completeness and Dynamics
P2P Collaboration	FN,FP	T,S	P,S,M	P2P-like Cleaning Scheme
TopK-PDI	FP,R	T,S	P,S,M	Inter-region Communication
bSpace	FN,FP	T,S	P,S,M	Bayesian estimation

Note: FN-False Negative, FP-False Positive, I-Invalid, R-Redundancy, T-Temporal, S-Spatial, P-Point, S-Smooth, M-Merge, A-Arbitrate, V-Virtualize

Based on the above analysis, we observe that existing algorithms and frameworks are mostly designed for RFID systems. Extension of these studies from RFID to IoT systems will have rich research opportunities for IoT data management.

B. Event Processing Layer

Event is an occurrence of interest in certain time, which could be either primitive or complex event [13]. In IoT systems, cleansed data can be treated as primitive events. As depicted in Figure 5 that, in event processing layer,

complex semantic events are extracted from low-level primitive events to support higher-level applications, which is called Complex Event Processing (CEP) [28]. CEP is deployed to detect interested behavior patterns and provide services as event detection, monitoring and response.

Figure 5. Complex Event Processing in IoT [13]

According to our survey, the existing CEP models can be classified into three categories:

1) Graph-based CEP Model

In graph-based model, complex events are expressed in tree structure with its leaf nodes represent the primitive events. Once the corresponding event occurs, the occurrence will be stored in the leaf node and pushed to its father node. If father node successfully detects according to its operation semantics, the result will be further pushed to upper node until the root detects this complex event's happen. Snoop [29] is the first CEP system provides simple support of graph-based time model for complex event detection in active database system. READY [30] and YEAST [31] are other CEP systems based on the graph model. A major disadvantage of these approaches is the lack of in-depth support on complex temporal event constraints. RCEDA [32] provides a detailed description language on time constraints in complex event and proposes a concept of non-spontaneous event. Many events are non-spontaneous as they cannot detect their occurrences by themselves unless expired, if associated with interval constraints [32]. RCEDA defines an ECA-like description language to express complex event and uses a pseudo-event for non-spontaneous event processing.

2) Automata-based CEP Model

In this model, event is constructed into corresponding automata, set of states and transition functions in automata-based model. Complex event is successfully detected if the automata reaches an accept state according to the event history. In Ode database, finite state automata is firstly adopted to detect complex event which is descripted in logic based language [33]. Cayuga [34][35] and Esper [36] use automata engine to process events from data streams. However, non-spontaneous event is not supported for these approaches. SASE is an automata-based CEP engine supporting non-spontaneous event detection [37]. It provides a plan-based query framework to process event which includes five stages: Sequence Scan/Construction, Selection, Window, Negation and Transformation.

3) Petri-net-based CEP Model

In this model, complex event is transformed into corresponding petri net in petri-net-based CEP model. Input place represents the primitive event and output place indicates the occurrence of complex event. The input place will be marked when corresponding event occurs. If all the input places are marked, transition will be fired. A complex event is successfully detected when the model reaches the output place. Petri-net-based model is first adopted by SAMOS [38] in active database event detection. In [39], a CEP middleware, PRES, is proposed which provides a series of temporal constructor to describe and process the non-spontaneous event.

4) Summary

Table 2 summarizes the major IoT event processing models and their characteristics.

Table 2. Classification of IoT Event Processing Approaches

	Model	Language	Logic Constructor	Temporal Constructor	Passive Event
Snoop	Graph	ECA Rules	or, any, etc.	Provided	Not Supported
YEAST	Graph	Yeast Specification Language	Basic Logic	Not Provided	Not Supported
RCEDA	Graph	ECA-Like Rules	$\wedge$, $\vee$, etc.	Provided	Supported
Cayuga	Automata	Cayuga Event Language	Basic Logic	Not Provided	Not Supported
SASE	Automata	SASE Event Language	Basic Logic	Provided	Supported
Esper	Automata	EPL	and, or, not, etc.	Provided	Not Supported
SAMOS	Petri-net	ECA Rules	Basic Logic	Not Provided	Not Supported
PRES	Petri-net	PRES Event Language	&, |,$\rightarrow$, etc.	Provided	Supported

Large amount of data from heterogeneous sources with complex semantic is a great challenge for IoT event processing, and the lack of semantic support in the processing makes it more difficult for real world applications. Accurate semantic definition and information processing, however, can achieve a wide range of incompatibility between information systems. Meanwhile, semantic support is still a missing part for existing solutions. Ontology which is a formal specification of a shared conceptualization can be an important tool to construct the concept and relationship between "things" in IoT [40]. We believe that ontology-based IoT semantic event processing is a promising research topic.

C. Data Storage and Analysis Layer

Data exchange, storage, compress and mining for IoT are implemented in Data Storage and Analysis Layer. How to manage the heterogeneous massive real-time IoT data is a challenging and key research topic.

1) Information Exchange Language

Data needs to be shared between different IoT devices, regions and systems. Due to the presence of heterogeneous platform, we need a cross-platform information exchange language for efficient communications. XML has been applied for Web information exchange in many systems [41]. As a result, many IoT information exchange languages are designed based on XML. PML (Physical Markup Language) [42] is one of the IoT information exchange standards designed by EPCGlobal for communication description between physical environments and objects. Its supported applications include inventory tracking, automatic transaction, supply chain management and object-to-object communications. IEEE 1451 is another important interface standard for intelligent sensors and actuators [43]. SensorML [44] is proposed by OGC (Open GIS Consortium) providing standard models and XML encoding to describe sensors and measurement processes based on IEEE 1451.

Other standards, including TML [45], EXDL [46], M2MXML [47] and BITX [48] are summarized in Table 3.

Table 3. Classification of IoT Information Exchange Languages

	Standard	Domain	Content
PML	XML	Logistics, Transportation, Healthcare, Manufacturing	EPC System Data
SensorML	IEEE 1451	Sensor Applications	Sensor Data Sharing and Interoperation
TML	XML	Environmental, medical, Industrial, Security, etc.	Data Exchange between Sensor and Application
EXDL	XML	Medical, Smart City	Emergency Message
M2MXML	XML	Intelligent Device Management	Data Exchange between Intelligent Devices
BITXmI	XML	Intelligent Device Management	Command and Communication between Intelligent Devices

These languages focus on different areas of applications, which make it difficult to share information between IoT systems. Semantic retrieval and processing crossing domains is also hard to implement because they do not have the same basic concept ontology. Therefore, defining a unified ontology concept and information exchange language covering different application domains is an important future work.

2) Data Compressing and Warehousing

As is analyzed in Section II, large volume of data is a main challenge for IoT data management. There are several solutions proposed to compress warehouse data in previous studies. A bitmap data structure is proposed to compactly represent a collection of EPC data [49]. This data type utilizes a key observation that it is more efficient by tracking EPC data in group than individually. It is easy to implement in DBMS and a series of operations are defined for efficient conversion, maintenance and calculation. Beside the EPC data, an effective path encoding scheme used to encode the flow information for logistic products monitoring is propose in [50]. This encode scheme is based on the *Chinese Remainder Theorem* and *Unique Factorization Theorem* [51]. Time information is encoded into a tree structure similar to XML region coding [52]. It can retrieve tracking and path oriented queries efficiently. For on-line analytical processing, the concept of RFID data warehouse is introduced in [53]. The storage schema named RFID-Cuboids is designed based on the observation that RFID objects usually move in large groups and data analysis takes place at different abstraction levels. The star storage schema is constructed centered with the fact table named *Stay* and linked by other supporting tables. Although only temporal and spatial dimension analysis is supported by RFID-Cuboids, as shown in Figure 6, we can extend additional data types making analysis in other dimensions possible. The above mentioned compress and storage scheme are established based on the assumption that sensing objects are moved in group so that the data is of high similarity. If not, the efficiency of using these storage schemes will be relatively low.

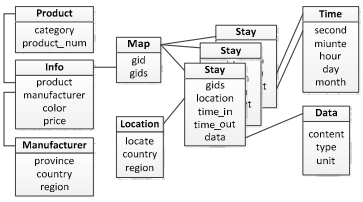

Figure 6. Extended Storage Schema for RFID-Cuboids [53]

3) Big Data Storage and Processing

The fundamental purpose of IoT is to have smart response and decision-making in domain applications based on accurate perception and intelligent data processing. IoT is also one of the typical emerging domains in which data is accumulating in a high speed and amount. As a hot research topic, big data processing is one of the important parts for IoT data management. Big data is a large and complex data set or stream which is difficult to process using traditional database management tools or processing applications [54]. The fast development of sensing technology makes IoT data management more complex, especially for the large amounts of unstructured data [55][56]. Table 4 summarizes and compares the characteristics of big data in IoT with other application domains.

Table 4. Typical Domains of Big Data Processing

Domain	Application Examples	User Scale	Data Speed	Data Scale	Data Type
Internet of Things	Sensor Data	Big	Fast	TB	Complex
Scientific Computing	Bioinformatics	Small	Slow	TB	Simple
Financial Analysis	Transactions	Big	Fast	GB	Simple
Social Network	Web2.0	Very Big	Fast	PB	Complex
Mobile Network	Smartphone	Very Big	Fast	TB	Complex
Multimedia	Audio Video	Very Big	Fast	PB	Simple

There are two computational paradigms for IoT big data processing: *Data Stream* and *Map-Reduce* [57]. In data stream model, data arrives in stream and the processing algorithm is tasked with data without explicitly storing it. Temporal granule data cleaning and time window event processing are data processing approaches based on data stream model. Typical real-time big data stream processing systems which can facilitate include the Twitter Storm [58], Yahoo's S4 [59] and LinkedIn Kafka [60]. In Map-Reduce model, data is distributed in slave machines and computations are performed in the sequence of map and reduce operations. The cloud computing has become the standard platform for big data analytics. There are many related research on data warehouse [61][62][63], distributed data processing [64] and data mining on the cloud computing platform [65][66].

D. Middleware for IoT Data Management

Middleware is a type of software that provides services to applications, enabling communications and management of data in distributed systems such as IoT [67]. In the report released by ISO/IEC [68], the functions of middleware for sensor networks include information collection, filtering, analysis, decision-making etc. Middleware systems for IoT data management generally fall into the following categories [69][70]:

1) Virtual Machine based Model

Virtual machine based model is consisted of virtual machines (VMs) and script interpreter. They usually provide a set of programming instructors in assembly-like language. User instructors are interpreted and loaded to VMs for execution. Typical middleware solutions include Mate [71], SensorWare [72] and MagnetOS [73]. This model has a high flexibility in interface to facilitate programmers to develop under heterogeneous hardware. However, it disadvantages in the dependency on code interpreter and a high system overhead. Furthermore, programming on the assembly-like low-level language is relatively difficult to use.

2) Database Model

Database model allows a simple, declarative style of querying submitted to the system and returns the result like a database. The middleware manages the network as a virtual database and the nodes as distributing data objects. Solutions like TinyDB [74], Cougar [75] and SINA [76] are implemented in database model and they can provide the management and optimization on nodes, data and queries. This high-level abstraction guarantees the independence of data processing applications with a set of simple query interfaces. Nonetheless, the database model needs to establish and maintain an overall abstraction of distributing nodes and system topology. This limits the extensibility and loses the in-depth control on the system.

3) Event-Based Model

Event-based model is a message-oriented paradigm leveraging Publish-Subscribe mechanism allowing asynchronous message exchange loose coupling between sender and receiver. This paradigm is suitable for pervasive scenarios with large amount of events such as healthcare and financial data. Middleware systems like Mires [77], DSWare [78] and PSWare [79] all provide the functions of complex event composition and detection. ALE is the application level events specification for communications on the interest of event between applications and EPC data sources [80]. There are a lot of industrial solutions based on the ALE specification. Although the event-based middleware has many advantages and can be developed quickly, as compared with the CEP technology, the real-time event detection, service reliability and information security issues in event-based middleware still requires further study.

4) Service-Oriented Model

Service-Oriented IoT middleware, such as $(SI)^2$ [81], SensorAct [82] and Hydra [83], is based on the Service-Oriented Architecture (SOA) [84]. With the support of SOA, standard services are provided to achieve flexible and scalable programming and simplify the application development process. This model is promising as it can merge and take advantages from other models such as database and event-based approaches. The collaboration and integration with cloud computing, information privacy issue and Quality of Service (QoS) are the major concerns for research.

5) Other Models

Other major approaches of IoT middleware include self-adaptive and mobile agent based middleware. MiLAN [85] and TinyCubus [86] are self-adaptive middleware systems which can continuously adapt the network configuration according to the feedback from applications. Mobile agent based middleware can intelligently migrate and clone agents to desired locations along with network changes. TinyLime [87] and Agilla [88] are typical solutions of this model.

6) Summary

In Table 5, we survey major middleware solutions for their characteristics and supported functions, especially in semantic, security and QoS. As is shown in the table, intelligent data processing is the first concern and opportunity in middleware research. Related issues and approaches have been analyzed in Section III. Moreover, the QoS problem, such as how to ensure the usability of services and the reliability of processing time, is another key research concern. Additionally, research on the data security and privacy management are still not enough.

Table 5. Comparison between Surveyed Middleware Solutions

Model	Middleware	Platform	Semantic	Security	QoS	Features
Virtual Machine	**Mate**	TinyOS	X	X	X	Mate instruction set and interpreter
	SensorWare	ARM Platform	X	X	X	Tcl-based script language
	MagnetOS	Java VM	X	X	X	Flexible deployment based on component architecture
Database	**TinyDB**	TinyOS	X	X	X	Distributed data flow architecture
	Cougar	MICA, Unix	X	X	X	Distributed database-like query language
	SINA	Sensor Node	X	X	X	Leveled node cluster structure
	SwissQM	Java VM	X	X	X	Combination of SwissQM VM and database-like query
Event	**Mires**	TinyOS	--	X	X	Publish-Subscribe data exchange paradigm
	DSWare	Sensor Node	--	X	X	Introduction of confidence function to deal single node unreliability
	Impala	ARM Platform	--	X	X	Lightweight event distribution framework
	PSWare	Java VM	√	X	X	Flexible complex event composition
	ALE	EPC	--	--	--	Wide support of industrial manufacturers
Service	**$(SI)^2$**	SOA	--	X	X	Service interface for smart devices
	SensorAct	Java VM	√	√	X	Lightweight RESTful API, message privacy protection mechanisms
	Hydra	SOA	√	√	√	Comprehensive development environment for IoT
Other	**MiLAN**	Sensor Node	X	X	√	Self-Adaptive network protocol stack
	TinyCubus	TinyOS	X	X	√	Dynamic system and data processing structure
	TinyLime	TinyOS	X	X	X	Agent migration and clones
	Agilla	TinyOS	√	X	X	Linda-like tuple space for inter-agent communication and context discovery

Note: √ Function Supported X Function Not Supported -- Not Explicitly Defined

IV. CONCLUSION AND FUTURE WORK

In this paper, we review the background of Internet of Things and analyze five distinguishing characteristics of its data. We then present a layered reference model for IoT data management. According to the model, we elaborate the existed research and solutions for each layer. The research challenges and opportunities are then identified and discussed to promote further works on promising topics.

Data security is an important aspect that we have not analyzed in this paper. With the relationship between devices, data and users become increasingly complex, data security and privacy problem need to be solved urgently. It will be a major challenge for the IoT industry.

REFERENCES

[1] I. Strategy and P. Unit, "ITU internet reports 2005: The internet of things," *Geneva: International Telecommunication Union (ITU),* 2005.

[2] E. Welbourne, L. Battle, G. Cole, K. Gould, K. Rector, S. Raymer, "Building the Internet of Things Using RFID: The RFID Ecosystem Experience," *Internet Computing, IEEE,* vol. 13, pp. 48-55, 2009.

[3] K. Ashton, "That 'Internet of Things' Thing," *RFiD Journal,* vol. 22, pp. 97-114, 2009.

[4] J. Cooper and A. James, "Challenges for database management in the Internet of Things," *IETE Technical Review,* vol. 26, p. 320, 2009.

[5] C. C. Aggarwal, N. Ashish, and A. Sheth, "The Internet of Things: A Survey from the Data-Centric Perspective," in *Managing and Mining Sensor Data*, Springer, 2013, pp. 383-428.

[6] N. A. Ali and M. Abu-Elkheir, "Data management for the Internet of Things: Green directions," in *Proceedings of the 2012 IEEE Globecom Workshops*, 2012, pp. 386-390.

[7] T. Fan and Y. Chen, "A scheme of data management in the Internet of Things," in *Proceedings of 2nd IEEE International Conference on Network Infrastructure and Digital Content*, 2010, pp. 110-114.

[8] S. Bin, L. Yuan, and W. Xiaoyi, "Research on data mining models for the Internet of Things," in *Proceedings of the 2010 International Conference on Image Analysis and Signal Processing,* 2010, pp. 127-132.

[9] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks,* vol. 54, pp. 2787-2805, 2010.

[10] R. Derakhshan, M. E. Orlowska, and L. Xue, "RFID Data Management: Challenges and Opportunities," in *Proceedings of the 2007 IEEE International Conference on RFID*, 2007, pp. 175-182.

[11] R. Cheng, D. V. Kalashnikov, and S. Prabhakar, "Evaluating probabilistic queries over imprecise data," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, 2003, pp. 551-562.

[12] C. C. Aggarwal, N. Ashish, and A. Sheth, "The Internet of Things: A Survey from the Data-Centric Perspective," in *Managing and Mining Sensor Data*, ed: Springer, 2013, pp. 383-428.

[13] F. Wang, S. Liu, P. Liu, and Y. Bai, "Bridging physical and virtual worlds: complex event processing for RFID data streams," in *Advances in Database Technology-EDBT 2006*, ed: Springer, 2006, pp. 588-607.

[14] Z. Guo, and A. Y. Zhou, "Research on Data Quality and Data Cleaning: a Survey," *Journal of Software*, 2002, pp. 2076-2082.

[15] S. R. Jeffery, G. Alonso, M. J. Franklin, H. Wei Hong, and J. Widom, "A Pipelined Framework for Online Cleaning of Sensor Data Streams," in *Proceedings of the 22nd International Conference on Data Engineering*, 2006, pp. 140-140.

[16] S. R. Jeffery, G. Alonso, M. J. Franklin, W. Hong, and J. Widom, "Declarative support for sensor data cleaning," in *Pervasive Computing*, ed: Springer, 2006, pp. 83-100.

[17] H. Gonzalez, J. Han, and X. Shen, "Cost-conscious cleaning of massive RFID data sets," in *Proceedings of the 23rd International Conference on Data Engineering*, 2007, pp. 1268-1272.

[18] J. Han, M. Kamber, and J. Pei, "Data mining: concepts and techniques, " Morgan kaufmann, 2006.

[19] A. Gupta, "Developing auto-id solutions using sun java system rfid software, " Oracle, Oct 2004.

[20] S. R. Jeffery, M. Garofalakis, and M. J. Franklin, "Adaptive cleaning for RFID data streams," in *Proceedings of the 32nd international conference on Very large data bases*, 2006, pp. 163-174.

[21] L. Meng and F. Yu, "RFID data cleaning based on adaptive window," in *Proceedings of the 2nd International Conference on Future Computer and Communication*, 2010, pp. V1-746-V1-749.

[22] Y. Zhuangt, L. Chen, X. S. Wang, and J. Lian, "A weighted moving average-based approach for cleaning sensor data," in *Proceedings of the 27th International Conference on Distributed Computing System*, 2007, pp. 38-38.

[23] X. Zhou and R. Chen, "Exploiting proximity readers for statistical cleaning of unreliable RFID data," in *Proceedings of the Third International Conference on Ubiquitous and Future Networks*, 2011, pp. 13-18.

[24] A. Solanas, J. Domingo-Ferrer, A. Martínez-Ballesté, and V. Daza, "A distributed architecture for scalable private RFID tag identification," *Computer Networks,* vol. 51, pp. 2268-2279, 2007.

[25] X. Peng, Z. Ji, Z. Luo, E. C. Wong, and C. Tan, "A P2P collaborative RFID data cleaning model," in *Proceedings of the 3rd International Conference on Grid and Pervasive Computing Workshops, The*, 2008, pp. 304-309.

[26] T. Jiang, Y. Xiao, X. Wang, and Y. Li, "Leveraging communication information among readers for RFID data cleaning," in *Web-Age Information Management*, ed: Springer, 2011, pp. 201-213.

[27] Y. Gu, G. Yu, X. L. Hu, and Y. Wang, "Efficient RFID Data Cleaning Model Based on Dynamic Clusters of Monitored Objects," *Journal of Software*, Vol.21, No.4, April 2010, pp. 632-643.

[28] D. C. Luckham, "The power of events," vol. 204: Addison-Wesley Reading, 2002.

[29] S. Chakravarthy and D. Mishra, "Snoop: An expressive event specification language for active databases," *Data & Knowledge Engineering,* vol. 14, pp. 1-26, 1994.

[30] B. Krishnamurthy and D. S. Rosenblum, "Yeast: A general purpose event-action system," *IEEE Transactions on Software Engineering*, vol. 21, pp. 845-857, 1995.

[31] R. E. Gruber, B. Krishnamurthy, and E. Panagos, "READY: A high performance event notification service," in *Proceedings of the 16th International Conference on Data Engineering*, 2000, pp. 668-669.

[32] F. Wang and P. Liu, "Temporal management of RFID data," in *Proceedings of the 31st international conference on Very large data bases*, 2005, pp. 1128-1139.

[33] N. Gehani and H. Jagadish, "Ode as an active database: Constraints and triggers," in *Proceedings of the Seventeenth International Conference on Very Large Databases (VLDB)*, 1991, pp. 327-336.

[34] L. Brenna, A. Demers, J. Gehrke, M. Hong, J. Ossher, B. Panda*, et al.*, "Cayuga: a high-performance event processing engine," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2007, pp. 1100-1102.

[35] A. Demers, J. Gehrke, B. Panda, M. Riedewald, V. Sharma, and W. M. White, "Cayuga: A general purpose event monitoring system," 2007.

[36] Esper Reference Documentation.
http://esper.codehaus.org/esper-4.9.0/doc/reference/en-US/pdf/esper_reference.pdf. Retrieved 1 May 2013.

[37] E. Wu, Y. Diao, and S. Rizvi, "High-performance complex event processing over streams," in *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, 2006, pp. 407-418.

[38] S. Gatziu and K. R. Dittrich, "SAMOS: An active object-oriented database system," *Data Engineering Bulletin,* vol. 15, pp. 23-26, 1992.

[39] W. Ye, W. Zhao, Y. Huang, W. Hu, S. Zhang, and L. Wang, "Towards Passive RFID Event," in *Proceedings of the 33rd Annual IEEE International Conference on Computer Software and Applications*, 2009, pp. 492-499.

[40] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge acquisition,* vol. 5, pp. 199-220, 1993.

[41] Extensible Markup Language (XML) 1.0 (Fifth Edition) Specification. http://www.w3.org/TR/REC-xml/. Retrieved 1 May 2013.

[42] D. L. Brock, "The Physical Markup Language," MIT Auto-ID Center, February 2001.

[43] IEEE Standard for a Smart Transducer Interface for Sensors and Actuators. IEEE Standards Association. Retrieved 1 May 2013.

[44] M. Botts, and A. Robin, "OpenGIS Sensor Model Language SensorML Implementation Specification," Open Geospatial Consortium Inc Tech Rep OGC 07-000, 2007.

[45] Transducer Markup Language (TML).
http://www.ogcnetwork.net/infomodels/tml. Retrieved 1 May 2013.

[46] Emergency Data Exchange Language (EDXL) Distribution Element, http://www.oasis-open.org. Retrieved 1 May 2013.

[47] M2MXML Specification Version 1.1.
http://m2mxml.sourceforge.net/specs.html. Retrieved 1 May 2013.

[48] BiTXml M2M Communications Protocol Rev 2.0.
http://www.bitxml.org. Retrieved 1 May 2013.

[49] Y. Hu, S. Sundara, T. Chorma, and J. Srinivasan, "Supporting RFID-based item tracking applications in Oracle DBMS using a bitmap datatype," in *Proceedings of the 31st international conference on Very large data bases*, 2005, pp. 1140-1151.

[50] C.-H. Lee and C.-W. Chung, "Efficient storage scheme and query processing for supply chain management using RFID," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 291-302.

[51] X. Wu, M. L. Lee, and W. Hsu, "A prime number labeling scheme for dynamic ordered XML trees," in *Proceedings of the 20th International Conference on Data Engineering,* 2004, pp. 66-78.

[52] C. Zhang, J. Naughton, D. DeWitt, Q. Luo, and G. Lohman, "On supporting containment queries in relational database management systems," in *ACM SIGMOD Record*, 2001, pp. 425-436.

[53] S. S. Chawathe, V. Krishnamurthy, S. Ramachandran, and S. Sarma, "Managing RFID data," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, 2004, pp. 1189-1195.

[54] Wikipedia Big Data. http://en.wikipedia.org/wiki/Big_data. Retrieved 1 May 2013.

[55] R. Bryant, R. H. Katz, and E. D. Lazowska, "Big-Data Computing: Creating Revolutionary Breakthroughs in Commerce, Science and Society," http://www.cra.org/ccc/docs/init/Big_Data.pdf. Retrieved 1 May 2013.

[56] Science. Special online collection, "Dealing with data,"
http://www.sciencemag.org/site/special/data. Retrieved 1 May 2013.

[57] R. Kumar, "Two computational paradigm for big data," *KDD summer school*, 2012.

[58] Storm. https://github.com/nathanmarz/storm. Retrieved 1 May 2013.

[59] L. Neumeyer, B. Robbins, A. Nair, and A. Kesari, "S4: Distributed stream computing platform," in *Proceedings of the International Conference on Data Mining Workshops*, 2010, pp. 170-177.

[60] K. Goodhope, J. Koshy, J. Kreps, N. Narkhede, R. Park, J. Rao*, et al.*, "Building LinkedIn's Real-time Activity Data Pipeline," *Data Engineering,* vol. 35, pp. 33-45, 2012.

[61] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang*, et al.*, "Hive-a petabyte scale data warehouse using hadoop," in *Proceedings of the IEEE 26th International Conference on Data Engineering*, 2010, pp. 996-1005.

[62] A. Abouzied, K. Bajda-Pawlikowski, J. Huang, D. J. Abadi, and A. Silberschatz, "HadoopDB in action: building real world applications," in *Proceedings of the 2010 international conference on Management of data*, 2010, pp. 1111-1114.

[63] S. Chen, "Cheetah: a high performance, custom data warehouse on top of MapReduce," *Proceedings of the VLDB Endowment,* vol. 3, pp. 1459-1468, 2010.

[64] Y. Yang, X. Ni, H. Wang, and Y. Zhao, "Parallel implementation of ant-based clustering algorithm based on hadoop," in *Advances in Swarm Intelligence*, ed: Springer, 2012, pp. 190-197.

[65] S. Nair and J. Mehta, "Clustering with Apache Hadoop," in *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, 2011, pp. 505-509.

[66] Z. Ding, Q. Yang, and H. Wu, "Massive heterogeneous sensor data management in the Internet of Things," in *Proceedings of the 2011 International Conference on Internet of Things*, 2011, pp. 100-108.

[67] Wikipedia Middleware. http://en.wikipedia.org/wiki/Middleware. Retrieved 1 May 2013.

[68] ISO/IEC JTC 1 SGSN. Study on Sensor Networks (Version 3), 03 September 2009.

[69] M.-M. Wang, J.-N. Cao, J. Li, and S. K. Dasi, "Middleware for wireless sensor networks: A survey," *Journal of computer science and technology*, vol. 23, pp. 305-326, 2008.

[70] S. Hadim and N. Mohamed, "Middleware: Middleware challenges and approaches for wireless sensor networks," *Distributed Systems Online, IEEE,* vol. 7, pp. 1-1, 2006.

[71] P. Levis and D. Culler, "Maté: A tiny virtual machine for sensor networks," in *ACM Sigplan Notices*, 2002, pp. 85-95.

[72] A. Boulis, C.-C. Han, R. Shea, and M. B. Srivastava, "SensorWare: Programming sensor networks beyond code update and querying," *Pervasive and Mobile Computing,* vol. 3, pp. 386-412, 2007.

[73] A. Boulis, C.-C. Han, and M. B. Srivastava, "Design and implementation of a framework for efficient and programmable sensor networks," in *Proceedings of the 1st international conference on Mobile systems, applications and services*, 2003, pp. 187-200.

[74] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: An acquisitional query processing system for sensor networks," *ACM Transactions on Database Systems (TODS),* vol. 30, pp. 122-173, 2005.

[75] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," *ACM Sigmod Record,* vol. 31, pp. 9-18, 2002.

[76] C.-C. Shen, C. Srisathapornphat, and C. Jaikaeo, "Sensor information networking architecture and applications," *Personal communications, IEEE,* vol. 8, pp. 52-59, 2001.

[77] E. Souto, G. Guimarães, G. Vasconcelos, M. Vieira, N. Rosa, and C. Ferraz, "A message-oriented middleware for sensor networks," in *Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing*, 2004, pp. 127-134.

[78] S. Li, S. H. Son, and J. A. Stankovic, "Event detection services using data service middleware in distributed sensor networks," in *Information Processing in Sensor Networks*, 2003, pp. 502-517.

[79] S. Lai, J. Cao, and Y. Zheng, "PSWare: A publish/subscribe middleware supporting composite event in wireless sensor network," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, 2009, pp. 1-6.

[80] EPCglobal. The Application Level Events (ALE) Specification, version 1.1.1, Mar, 2009.

[81] J. Anke, J. Müller, P. Spieß, and L. W. F. Chaves, "A service-oriented middleware for integration and management of heterogeneous smart items environments," in *Proceedings of 4th MiNEMA workshop*, 2006, pp. 7-11.

[82] P. Arjunan, N. Batra, H. Choi, A. Singh, P. Singh, and M. B. Srivastava, "SensorAct: a privacy and security aware federated middleware for building management," in *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, 2012, pp. 80-87.

[83] M. Eisenhauer, P. Rosengren, and P. Antolin, "A development platform for integrating wireless devices and sensors into ambient intelligence systems," in *Proceedings of the 6th IEEE SECON Workshops* , 2009, pp. 1-3.

[84] Wikipedia Service-oriented architecture
http://en.wikipedia.org/wiki/Service-oriented_architecture. Retrieved 1 May 2013.

[85] W. B. Heinzelman, A. L. Murphy, H. S. Carvalho, and M. A. Perillo, "Middleware to support sensor network applications," *Network, IEEE,* vol. 18, pp. 6-14, 2004.

[86] P. J. Marrón, A. Lachenmann, D. Minder, J. Hahner, R. Sauter, and K. Rothermel, "TinyCubus: a flexible and adaptive framework sensor networks," in *Proceedings of the Second European Workshop on Wireless Sensor Networks*, 2005, pp. 278-289.

[87] C. Curino, M. Giani, M. Giorgetta, A. Giusti, A. L. Murphy, and G. P. Picco, "Tinylime: Bridging mobile and sensor networks through middleware," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, 2005, pp. 61-72.

[88] C.-L. Fok, G.-C. Roman, and C. Lu, "Rapid development and flexible deployment of adaptive wireless sensor network applications," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, 2005, pp. 653-662.