DoT(Database for IoT): Requirements and Selection Criteria

Trupti Gurav
Asst.Professor
Department of Computer Engg
SKN College of Engg
Pune, Maharashtra, India

R. A. Kudale
Asst.Professor
Department of Computer Engg
SKN College of Engg
Pune, Maharashtra, India

ABSTRACT
The Internet of Things (IoT) is the dynamic and global network infrastructure where smart interconnected objects continuously generate massive amount of data which then transmitted over internet. A lot of work is done on building low cost small devices, their connectivity and communication. However there is a little work done on data management as well as the type of database systems (DoT: Database for IoT) that must be used in IoT. DoT requirements are different than the traditional database requirements. WSN Data management solutions are applied to Iot but WSN is just a part of IoT and it needs another solution for data management. In this paper we will focus on different types of IoT data, data management and their challenges in IoT . This paper also considers database requirements for IoT and focuses on selection of DoT by considering type of devices and their capabilities.

Keywords
Database management challenges, Internet of Things, Database for IoT (DoT) .

1. INTRODUCTION
With a cell phone in every pocket, humans are now able to communicate with friends, coworkers and family like never before. However, the accessibility and portability of the internet has not just changed the way humans communicate — with the availability of inexpensive network cards and high coverage networks, machines are gradually catching up to human usage of the internet. This paradigm shift, from human to machine domination of Internet traffic, is termed the Internet of Things (IoT). The 'thing' in IoT could be a person with a heart monitor or an automobile with built-in-sensors, i.e. objects that have been assigned an IP address and have the ability to collect and transfer data over a network without manual assistance or intervention. The embedded technology in the objects helps them to interact with internal states or the external environment, which in turn affects the decisions taken. In IoT the devices, traditionally considered as unintelligent, have started communication through internet. For the end user, everything appears to happen automatically, but in reality it needs coordination of all actions and ensuring smooth operations of these devices through web services and databases. At any given time, the IoT ecosystem will have multiple devices, systems and tools that will generate significant volumes of data. IoT systems are either *closed system* , meaning the nature of the things making up the system is known or *wide- open means* creators are not able to anticipate the universe of "things" that could be connected, or their quantity [1] . IoT system generates massive amount of data which later on used for further processing. Storing, processing and retrieving data altogether is called Data Management. In the context of IoT, data management should act as a layer between the objects and devices generating the data and the applications accessing the data for analysis purposes and services.

It has been projected recently that there is a renewed interest in database systems research that focuses on alternate models other than the traditional relational model. The new database for IoT (DoT) needs some aspects such as the utilization of remote storage at the Things layer, non-structural data support, relaxation of the Atomicity, Consistency, Isolation, and Durability (ACID) properties to trade-off consistency and availability, and integration of energy efficiency as a data management design primitive [2].

IoT data management framework from the perspective of IoT architecture is proposed in [3]. According to [3] both offline and real-time data cycles need to be supported in an IoT-based data management system, to accommodate the various data and processing needs.

Section 1 discusses about data management in IoT and how it is different from traditional data management. Section 2 focuses on database requirements and selection criteria in IoT.

2. DATA MANAGEMENT
Data management in IoT systems takes place at two different levels.

First Data management takes place at device level (ie. Field deployed devices) where online, real time data is collected. This data is then propagated to other smart objects or to server/cloud. Second data management is at cloud/server level which handles the mass storage and in-depth analysis of IoT.

The first data management task is fundamentally simple: It involves collection of small subsets of data, analysing them, and giving respond with automated, real-time control actions in each case. As an embedded device is designed for a limited range of jobs, it becomes easy for developer to know quite a lot about its data, including the types, relationships, volume, and velocity of data in turn make it easier to assess and direct what actions are to be taken. The second operation at an aggregation point on the cloud or in communication with it must collect and manage data from a large number of field devices, analyse it, and determine if it has actionable intelligence that must either be responded to quickly or passed along for analysis or action in the future.

Modest amount of data is managed at field-deployed devices, they do propagation of query requests and result to and from sensors and smart objects. But huge amount of data is handled at an aggregation point (the cloud being the most obvious example).In short first level of data management is communication intensive whereas second level is storage intensive.

2.1 IoT data Characteristics

IoT systems have massive and growing number of data sources; sensors, RFIDs, embedded systems, and mobile devices. Data is generated by millions of these data sources connected over network and hence the amount of data generated is massive and heterogeneous in nature.

Cooper, J., & James[2] have categorized the data into the following areas: RFID, address/unique identifiers, descriptive data, positional and environmental data, sensor data, historical data, physics models, and command data.

Based on purpose of IoT system, devices are periodically sending observations or keeping track of some abnormal behaviour. Thus data gathered is geographically-dispersed real-time data. Along with all this data every object or thing has its own metadata like its identification, location, processes and services provided by that thing etc.

2.2 Lifecycle of IoT data

The lifecycle of data within an IoT system—illustrated in Figure 1—proceeds from data production to aggregation, transfer, optional filtering and preprocessing, and finally to storage and archiving. Querying and analysis are the end points that initiate (request) and consume data production, but data production can be set to be "pushed" to the IoT consuming services [4]. Production, collection, aggregation, filtering, and some basic querying and preliminary processing functionalities are considered online, communication-intensive operations. Intensive preprocessing, long-term storage and archival and in-depth processing/analysis are considered offline storage-intensive operations.[3]

2.3 Data-management Challenges

By knowing the purpose of IoT and the type of devices used in the IoT following are the data management challenges.

1. Data diversity: Internet of Things involves weaving together multiple connected devices, the data requirements are defined by weaving together disparate data sources. It requires the integration of customer data, billing information, device data, web services data (e.g. weather and traffic data), and more.

2. Data volume and velocity: The two primary characteristics of the data flood are volume and velocity. Devices generate a flood of data that must be ingested, evaluated for trend data and anomalies, and used to trigger various actions. The velocity challenge puts a massive load on data management technologies, because the data may be pouring in at a rate of millions of elements a second. The data must be stored to provide a historical context for evaluating trends over time.

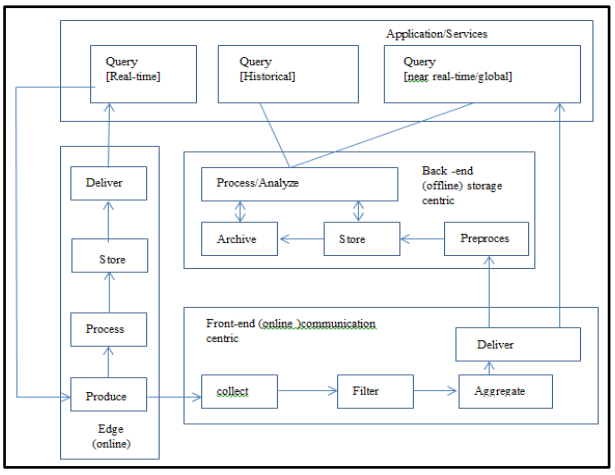

Fig 1: IoT data Lifecycle

3. Real-time data processing: Internet of Things system has the ability to respond to users actions, more or less in real-time. Since this response is often an orchestrated effort leveraging a variety of device inputs and other data, the ability to process the data and respond in real-time puts an extra load on the data infrastructure.

4. Data integration: IoT data has the multiple use cases. Device data may be used operationally—triggering certain actions in real-time—and it may also be leveraged to glean analytic insight regarding trends and for predictive purposes. These demands typically require very different tools, and introduce a significant data integration challenge.

3. DATABASE FOR IOT (DoT)

3.1 Dot Requirements

The Internet of Things success relies on the data that applications create; however, vetting a database for IoT apps requires criteria totally different from traditional enterprise databases**.** Database system should be able to handle all different unknown data. Following are the DoT requirements.

1. **Scalability:** A database for IoT applications must be scalable. In Iot massive and growing number of data sources collects massive amount of data which the traditional RDBMS cannot be scaled to manage. Data will need to be managed via responsible local ownership. Local owners will decide which data and services to make available to global network[2]. DBMS used on devices and/or gateways must be able to handle the workload.

2. **Real time data handling and processing:** Sensors used in IoT system gathers some real time data for monitoring all sorts of environmental phenomena, for example, weather, temperature, and noise. Decisions are made based on captured data. Sensors and grid technology can be used to capture vast amounts of data very quickly, but querying and mining these can be problematic, particularly when the analysis must be achieved in real-time. Big data stores like Hadoop provide a low-cost high-volume data warehouse, but it doesn't address high-velocity data or ad hoc analytic processing.

3. **Capability to handle heterogeneous data:** DBMS must be able to handle data diversity, data volume

and velocity. Big data database supports multiple formats of data, facilitates faster storage of data, meets scalability requirements and ensures data consistency. Given the characteristics of data generated in IoT, big data databases are potentially the suitable options to consider. However, while the big data appliances like Hadoop meet requirements on analytics such as the trend analysis quite well, it do not meet up to the demands and rigor of online data processing. NoSQL databases are not designed to handle the analytic processing of the data or joins, which are common requirements for Internet of Things applications.

If a new device gets added into IoT system and database system has to accommodate

New and unforeseen data then the obvious challenges are:

1. The ability The ability to ingest new data that has a previously unknown structure
2. The ability to execute analytics on #1
3. The ability to integrate analytics on #1 with analytics on previously known data

#1 is handled well by NoSQL DBMSs. Or RDBMS via DDL. RDBMS handle structure of data through the database dictionary whereas NoSQL DBMSs handle this through different meta data.

4. **Transactional integrity:** In traditional database systems transaction management mechanisms guarantee the ACID properties in order to enforce overall data integrity [4]. ACID properties are atomicity, consistency, isolation, and durability. A transaction must complete in its entirety or not at all, a transaction must leave the database in a consistent state, transactions should not show other transactions, and intermediate results and changes made by a transaction must be permanent. Dynamic nature of IoT systems makes it difficult to maintain ACID properties. If IoT system is *closed* (used for specific purpose) then these traditional methods can apply. But for *wide-open* IoT systems these ACID properties must be relaxed. IoT sensors generate data rapidly, they do not entail the same kinds of transactions one finds in traditional enterprise business applications. This reduces the need for ACID transactions. The execution of distributed queries is based on the transparency principle, which dictates that the database is still viewed logically as one centralized unit, and the ACID properties are guaranteed via the two-phase commit protocol. Sometimes innovative applications and services may require location and context awareness. In this case transparency may not even be required in IoT.
5. **High availability:** If a node in the database cluster is down, it should still be able to accept read and write requests. DoT must be built to eliminate single points of failure and are optimized to ensure that the end user does not experience an interruption in service or degradation in user experience when hardware or networks fail. High availability can be ensured with redundancy features as done in most big data databases. Another approach to ensuring high availability with regards to writes is to use a distributed messaging system such as Apache Kafka *or* Amazon Kinesis, which is based on Apache Kafka[5]. These systems can accept writes at high volumes and store them persistently in a publish-and-subscribe system. If a server is down or the volume of writes is too high for the distributed database to ingest in real time, data can be stored in the messaging system until the database processes the backlog of data or additional nodes are added to the database cluster.
6. **Spatiotemporal scalability:** IoT systems gather real world data which shows spatial as well as temporal characteristics. Spatiotemporal phenomena have become an important aspect of many of the real world applications. Many data objects in real world have attributes related to both space and time, and managing them using existing RDBMS is complex and in-efficient, as these objects which show spatio-temporal behavior are multi-dimensional in nature.
7. **Fast and reliable:** Users of IoT systems needs different data as per the context. A casual user may need some information whereas an expert user needs data for analysis. So the query language for IoT must produce this data at very fast rate. The quantities of data are so vast that it would be unrealistic to expect any sort of uniform structure except for closed systems. Current popular query languages like SQL rely on structured data. Semi-structured data on internet can be represented in Extensible Markup Language (XML) It is a well-accepted technology that supports interoperability at a technical rather than a semantic level. XQuery, a language for querying XML, can combine documents, web pages, and links to relational databases[6]. Query languages for semi-structured data usually adopt an underlying hierarchical data model, for instance a unidirectional graph. There are, however, inherent problems with hierarchical data models, such as, difficulty in representing many-to-many relationships. So query languages for semi-structured data in IoT must be good enough to provide fast response to users.

3.2 Selection of DoT

In [1] characteristics of field deployed devices and the cloud are considered for database selection. Selection of DBMS will take place at two ends, first DBMS on field-deployed devices and second cloud DBMS. If IoT is *closed* system it won't need much scaling and database system would not necessarily be different. But if IoT system is *wide open* , DBMS should be able to handle data which may be unknown and database system on field-deployed devices may be different from the database system in the cloud. In order to properly handle IoT data and its requirements, it is critical to find the right IoT database. There are many factors to keep in mind when choosing a database for an IoT application, and they do not always align with the needs of other more traditional enterprise databases.

Field-deployed devices in the IoT are resource constraint. For such devices memory is less and OS is a less resource hungry. So if we are selecting a database system for field-deployed devices the database system should:

1. Have small code size.
2. Use little stack
3. Preferably allocate no heap memory.
4. Have no or minimal external dependencies.
5. Have built-in ability to replicate data(to a gateway or directly to cloud)
6. Replication should be "open" ,meaning be able to replicate to a different database system
7. Have built-in security features.
8. Nice to have:
    a. Built-in analytics to aggregate data prior to replicating it.
    b. Ability to define schema
    c. Ability to operate entirely in memory.

A database system for the cloud is expected to handle heterogeneous data at a large scale, as well as execute analytics and will, therefore, need ample resources. So cloud based DBMS should be able to take maximum advantage of the resources available, including being able to scale horizontally (across cores, CPUs, and servers).

Databases like Oracle, MS SQL Server and IBM DB2 have scalability, spatial, temporal and in-memory technology and also support big data. However, client will have to pay a higher price to get enough IoT scalability out of them. Most NoSQL databases including MongoDb and TempoDB claim that they can handle the time stamped data, scalability, rich query and index support but they do not have required features in terms of transactional integrity features.[7]

4. CONCLUSION

The Internet of Things generates new streams of data previously unimaginable, both in variety and quantity. DoT should be scalable, fast and reliable. Current open source big data and NoSQL databases have little support for these requirements and traditional databases also have issues with the same. Based on understating of the IoT requirements, correct databases must be selected. We have also mentioned selection criteria for DoT. DBMS at field deployed devices needs to operate in a constrained environment. A cloud DBMS needs to be able to effectively and efficiently utilize the ample resources available to it. There is no one specific database of IoT systems as the emerging database technologies have to handle complex IoT operations. It is important to decide database architecture for IoT systems. It needs identifying the data storage requirement and then defining the relevant database architectures to cater to future IoT related requirements. The various client/server, distributed, or embedded in-memory/in-process DBMS architectures are targeted for roles at different levels in the IoT, on the devices, or cloud level for data management. Thus at the end we can say that no one database system will be able to span the range of applications that will emerge, requiring developers to not depend on just one but several, and pick database as per the requirement.

5. REFERENCES

[1] Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems. .

[2] Ding, W. and Marchionini, G. 1997 A Study on Video Browsing Strategies. Technical Report. University of Maryland at College Park.

[3] Fröhlich, B. and Plate, J. 2000. The cubic mouse: a new device for three-dimensional input. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems

[4] Tavel, P. 2007 Modeling and Simulation Design. AK Peters Ltd.

[5] Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[6] Forman, G. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (Mar. 2003), 1289-1305.

[7] Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE.

[8] Y.T. Yu, M.F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions", Journal of Systems and Software, 2005, in press.

[9] Spector, A. Z. 1989. Achieving application requirements. In Distributed Systems, S. Mullender