**Introduction**

Internet of Things is a network established by uniquely identifiable commodity objects or devices equipped with some sensing system [1]. Internet of Things paradigm enables the objects, which then communicates with other objects for data exchange through an existing physical network assembly. Therefore, Internet of Things promotes a seamless connection between the smart devices, scatter around us, and the physical world to ensure full automation that eventually improves human life. heart monitoring implants, automobiles with embedded sensors, firefighter’ devices, smart thermostat systems, and Wi-Fi enabled washer/dryers are examples of Internet of Things-enabled commodity devices. As the arena of Internet of Things is expanding, the number of Internet of Things-enabled applications is also rapidly growing, which results in massive growth of smart devices. This swift increase in the number of sensing things is responsible for generating and storage of a plethora amount of diversified data at a much faster rate. The Internet of Things objects sense and collect the data from the highly sparse geographical environments. The data is exchanged with remotely stationed peer devices for numerous quick and efficient operations. As a result, the cloud computing technology has emerged as one such innovation that have been invented to efficiently tackle the growing Internet of Things issues [1].

Internet of Things paradigm is increasingly encouraging the universal connectivity of the intelligent objects within the internal or external world. The rapid growth of Internet of Things-enabled objects and storage technology, have resulted into the massive amount of heterogeneous digital footprints and sizeable traces. A huge amount of data is being generated by various sensing sources every day. It is observed that the primary sources of Internet of Things are sensor-enabled devices, unlike the traditional Big Data, where social media is the major contributor in data collection as compared to the sensing systems. Therefore, Internet of Things can be seen as a subset of traditional Big Data. The actual pattern and nature of such data is indistinct, but is certainly large, complex, structured and unstructured. Some important attributes of Internet of Things includes volume, variety, and velocity and some core elements like sensor-embedded devices, intelligence for quick decision making, and connectivity for data sharing. Apparently, to obtain constructive insights from Internet of Things, efforts are required for Internet of Things modeling in contrast to that of traditional data. Also, the rapid growth of sensing devices under Internet of Things is generating such a large scale complex and heterogeneous data that the available computing capacity of the existing systems are unable to successfully match up the data challenges. Today, this has emerged as one of the core issues for the data science community. The storage capacity and also the processing power of the existing data computing systems are failed in handling the data stress. As Internet of Things and its applications are majorly impacting the human life, the scientific communities contemplate a broader outreach from the processing and sharing of Internet of Things across the variety of the several commodity devices around us. Consequently, the development of new capable technologies is encouraged to cater the current data processing needs [1].

**What is Data Distribution Service?**

Data Distribution Service is a protocol for the Internet of Things which enables network interoperability for associated machines, enterprise systems, and mobile devices. It provides the needed scalability, performance, and Quality of Service required to support Internet of Things applications. DDS can be deployed in platforms ranging from low-footprint devices to the Cloud and supports efficient bandwidth usage, as well as agile orchestration of system components. It provides a global data space for analytics and enables flexible real-time system integration [2].

DDS simplifies software systems, and reduces risk and costs through development, integration, deployment, and lifetime maintenance of distributed software systems. Historically, DDS has been used in large Distribution of Data systems to satisfy open architecture requirements for extensibility, maintainability, composability, and interoperability, but only in the larger computer components of these systems. Now, with the availability of small-footprint implementations, many other applications can benefit from standardized publish subscribe communications, including Android apps. The data distribution service standard contains an easy to use, well defined Application Programming Interface. APIs allows the developer to write portable codes, that will work with any compliant Data Distribution Services implementation. The DDS standard references the Real Time Publish Subscribe (RTPS) Wire Protocol standard which defines the wire protocol for DDS communications. This allows applications built with different DDS implementations to communicate, or interoperate, with each other. Users of DDS do not tie themselves to a particular vendor, but to a standard, and can change or intermix vendors throughout the development and deployment cycles [3].

Each application communicating over DDS contains the DDS API and provides the discovery, and other required communication details. DDS simplifies communications processes among different system types, making distributed development easier, faster, and more reliable. A DDS Communications Middleware simplifies Android project from development through initial deployment and maintenance over the life of the system [3].

In a distributed system, middleware is the software layer that lies between the operating system and applications. It enables the various components of a system to more easily communicate and share data. It simplifies the development of distributed systems by letting software developers focus on the specific purpose of their applications rather than the mechanics of passing information between applications and systems [4].

**How does Data Distribution Service work?**

Data Distribution Systems are in charge of transferring information: Information is transferred from publishers (producers and senders of messages) to subscribers (consumers and receivers of messages). Even while operating on unlike platforms or operating systems subscribers and publishers using DDS can and still communicate with one another. Exchanges can happen between tens of thousands of devices at the simultaneously, each one of which can be publishers, subscribers, or both simultaneously. Systems using DDS to communicate do not rely on each other’s systems to send and process information. A publisher can publish data even if there is no subscriber looking for the data. Subscribers can obtain data from other publishers if the original publisher it was getting data from fails. DDS automatically recognizes how to send and receive messages with other DDS users. Data Distribution Service is able to conclude which users should receive messages, where these users are located, and what to do if the receiver is unavailable. This simplifies data distribution, reduces the code required to achieve message delivery and hence saves time [3].

Due to the Data Distribution Service API, DDS participants can be on the same machine or across a network and communicate. Adding an additional communication participant on any operating system or hardware platform becomes an easy task, because there is no need to know or configure IP addresses, or take into account the variances in machine designs. Each version of DDS can perform the same minimum set of functions in the same way with the same results, this system description is called an “open standard system”. System components from different manufacturers can be substituted and take over for each other with minimal or no changes to the larger systems in which they operate. This saves costs and avoids vendor lock-in [3].

Data Distribution Service middleware uses logical QoS policies, as set by the applications at runtime, to balance efficiency and determinism. Data differ in priority, reliability, timing, and other non-functional properties. DDS balances use of scarce resources to distribute data at the right time. Real-time systems interact with the real world. Data must be delivered on time, the right data too late is a failure. The QoS contracts ensure these timing relationships. For example, if a subscriber requires an update every 10ms and its matched publisher does not deliver, the system declares an error, enabling remedial action. QoS policies cover many characteristics, including urgency, importance, reliability, persistence, and liveliness [5]. DDS addresses data in a manner similar to relational databases. It can manage data by both structure related topics, using key-fields, and allow ad-hoc queries and filters on content and time so applications can extract specific data as needed. The DDS standard wire protocol implements reliable multicast over plain UDP sockets, allowing systems to efficiently benefit from modern networking infrastructures. Unlike message-centric products, DDS offers explicit application support for information lifecycle awareness. For instance, it detects, communicates, and informs applications about first and last appearances of data updates. This facilitates timely responses to new and outdated information [5].

**Data Analytics of Data Distribution**

Data analytics (DA) is the process of examining data sets in order to draw conclusions about the information they contain, increasingly with the aid of specialized systems and software. Data analytics technologies and techniques are widely used in commercial industries to enable organizations to make more-informed business decisions and by scientists and researchers to verify or disprove scientific models, theories and hypotheses [6].