Chapter II

WIRELESS SENSOR NETWORKS AND SOFTWARE AGENTS: A PREFACE

2.1 INTRODUCTION

The promising field of WSNs amalgamates sensing, computation, and communication into a device more commonly known as mote. Numerous such motes are connected to form a marine that extends the reach of cyberspace out into the real world. This revolutionary technology finds its way in the huge plethora of applications like disaster relief, habitat monitoring, health care, home networks, detecting chemical, biological, radiological, nuclear, and explosive materials, just to name a few [30,37,98,128,141,143].

Software agents on the other hand are proactive intelligent objects capable of performing the assigned task. Mobile agents are unique subset of software agents that have the capability to migrate from machine to machine in a heterogeneous network. They can defer their execution at an arbitrary point, migrate themselves to a different machine and may resume execution at a new location. A collection of such autonomous entities is being referred to as Multi Agent System (MAS) [26,74]. Available literature reflects that MAS is growing exponentially and is being applied in personnel management, e-commerce, search engines, electronic gadgets, manufacturing and production processes etc. [41, 43, 104].

Owing to the huge number of applications that software agents and distinctively mobile agents cater, it is feasible to exploit and use the power of mobile agents into these self configuring, self healing networks. This thesis contributes towards integrating mobile agents into WSNs so as to increase their lifetime and to provide an intelligent substitute to the conventional client server paradigm for routing the data from source to destination.
The upcoming section presents a meticulous study of wireless sensor networks and its applications in various domains. A portrayal of software agents is provided in the subsequent sections.

2.2 WIRELESS SENSOR NETWORKS: THE BACKGROUND

WSN is a special class of adhoc networks which has boomed up as a result of current advances in networking, semiconductor, and material science technologies. The heart of these self deployed networks is a dedicated diminutive entity known as sensor.

![Figure 2.1: Components of a Sensor Node](image)  

As shown in Figure 2.1, the sensing unit is capable of monitoring various kinds of data like acoustic, visual, seismic and temperature data. The communication module is a kind of radio unit capable of short range communication (up to tens of meters), while the processing unit contains small memory and a processor with limited size and processing speed. A sensor node functions using a non-rechargeable battery contributing to one of the major drawbacks of sensors [56, 64]. In order to address the problem of inability of recharging, developers are working towards a mass production of nodes, which will significantly lower the per device cost, and to deploy them liberally as disposable devices.
Before further discussing the architecture and applications of WSNs, a short description related to its evolution is given in the next section.

### 2.2.1 Origin of WSN

As stated above, WSN is an association of compact micro sensors with wireless communication capabilities. Like many advance technologies, WSN owe its root in military and heavy industrial applications. The first wireless network that is inline with the latest WSN is the Sound Surveillance System (SOSUS) developed on submerged acoustic sensors. Sensors in SOSUS were distributed in the Atlantic and Pacific oceans.

Stimulated by the developments pertaining to Internet in 1960s and 1970s to develop the hardware for today’s Internet, Defence Advanced Research Projects Agency (DARPA) initiated the Network (DSN) program in 1980[21]. The motive was to explore the design challenges related to WSN. With the birth of DSN and its penetration into education through Carnegie Mellon University and the Massachusetts Institute of Technology, WSN technology could find its base in household, education and civilian scientific research.

Very soon, public and private communities started deploying sensors to monitor air quality, detect forest fire, forecast weather, prevent natural disaster etc. The sensors however at that time were bulky, expensive and made use of proprietary protocols. The use of such WSNs thus weighed down the industry which used it. This disproportionate relation of high cost with low volume of sensors declined their pervasive use.

Realising the potential of the network, industry and academia joined hands to solve the engineering challenges associated with sensors and lead to the production of modern sensors: low cost miniature size sensors, having simplified development and maintenance tasks.
2.2.2 Communication Architecture of WSN

Communication between the nodes of WSN is made possible when number of motes are deployed covering a given geographical area. Larger is the number of sensors covering the geographical region, higher would be the accuracy. Figure 2.2 shows a schematic diagram of the communication architecture of WSN. Here, each sensor senses the data (usually sensors are application specific) related to the event. These dispersed sensor nodes collect and transmit data to base station routed through the number of hops. A base station may be a fixed or mobile node capable of connecting this network to an existing communications infrastructure offering the services to the end user [30, 56, 64].

![Communication Architecture of WSN](image)

Figure 2.2 : Communication Architecture of WSN

Though the above described architecture is well suited to almost all the applications of WSN, communication in WSN provides an interplay between the energy which is spent on communication vis a vis on computation. The goal is to understand the impact of these constraints on the overall usefulness of the sensor networks.

Since past few years, rigorous research that addresses the potential of collaboration among sensors in data gathering, processing, coordination and management of the sensing activity is being conducted. In most applications, sensor nodes are inhibited in terms of energy and communication bandwidth. Thus, innovative techniques to eliminate energy inefficiencies that shorten the lifetime of the network and efficient
use of the limited bandwidth are vital especially in non-deterministic areas like remote jungles, undersea areas etc. Such constraints united with a typical deployment of large number of sensor nodes pose many challenges to the design and management of WSNs and necessitate energy-awareness at all levels. For example, one of the main objective is to find ways and means for energy-efficient routing and reliable transmitting of data from the sensor nodes to the sink so that the lifetime of the network is maximized [35]. There is a definite need to enhance the power of each sensor individually so that they can carry out the task of sensing dedicatedly and for a much longer span of time.

*The core challenge is thus to synthesize the capacity of the sensor without causing any extra burden on either the hardware or network as a whole.*

### 2.2.3 Applications of WSN

Research in WSN supports abundant applications in almost all domains ranging from household appliances to military surveillance. The huge plethora of such applications is majorly classified into three main categories depending upon the initiative taking entity. Each of these categories is described as follows:

- **Event Driven Model**

  The applications in this model are initiated by source and the source reports to the base station immediately on occurrence of an event. Event driven applications are generally delay intolerant and interactive. This model is majorly applicable to mission critical applications [18,24,28]. The interaction is between the sink and the group of nodes deployed to monitor the event in an unattended area. The data thus sensed is likely to be highly correlated and redundant. Also, the data traffic may be either of low or high intensity depending on it being sensed by a single sensor or by a set of sensors. This phenomenon of more than one sensor sensing the same event is known as event showers. Examples of such applications include earthquake, forest fire, military based applications etc.[7,19,100,125].
• **Query Driven Model**

Also termed as sink initiated model, applications in this model reports on the status of an area when requested. They have similar characteristics as that of event driven applications except that the data is pulled by the sink in the former case. For example, wind stream during storm [128].

• **Periodic Model**

Applications in this model periodically monitor the event and then report to the base station. The model is also referred to as continuous data delivery model. Applications include traffic control, weather tracking etc.[107].

The subsection below describes some of the applications of WSN belonging to one or more of the category listed above.

**A) Military based Applications**

As described earlier, US military sowed the seeds of the ubiquitous WSNs which we experience in today’s world. Military based applications are so closely related to WSN that it is very difficult to ensure that whether motes were developed because of military and air-defence needs or whether they were invented independently and were subsequently applied to army services. WSN finds its usage in military in numerous ways like military situation awareness, detection of enemy unit movements on land and sea and battle field surveillance (target classification) etc.

Sensors have replaced the mines in the army areas as mines are obsolete and can be hazardous to civilians. These sensors, being connected wirelessly form a network, detect any intrusion of hostile units and alarm the army. Then, the prevention of intrusion will be the response of the army. The use of WSN ensures complete visibility of the field and makes communication over the long radius possible. Several applications related to this scenario of detection and classifications of objects have been developed. One such application is being demonstrated by Ohio State
University. The name of this project is “A line in the Sand” [117] and refers to the deployment of ninety nodes which are capable of detecting metallic objects. The ultimate objective is the tracking and classification of moving objects with significant metallic content and specifically the tracking of vehicles and armed soldiers. Other beings (for example, civilians) were ignored by the system.

A comprehensive study done in the field of military based applications reveals that most of the research efforts has been undertaken with reference to wartime scenarios. Peacetime applications such as homeland security, property protection and surveillance, border patrol, etc. are activities that perhaps in future sensors networks will undertake.

(B) Forest Fire Detection

Detection of forest fire is one of the prime examples of event driven applications which can be handled using WSNs. This application is of prime concern because in addition to causing the tragic loss of lives and a great hazard to ecologically healthy grown forests, they cause an irreparable damage to the atmosphere and environment(30% of carbon dioxide in the atmosphere comes from forest fires)[15].

The main source of fire in these abandoned/unmanaged areas is that they are filled with leaves, dry and parching wood etc. which together form an extremely combustible material and represent the perfect context for initial-fire ignition and act as fuel for later stages of the fire. The use of WSN for forest fire detection has gone beyond all its previous approaches(like manned observation towers, camera surveillance systems, satellite imaging technologies etc.). WSN Technology integrated with other technologies or networks has been used for detection and prevention of wood fire systems. Lloret et al. [7] has deployed a mesh network of sensors provided with internet protocol (IP) cameras where sensors detect the fire at the beginning and send an alarm signal to the base station. Son et al. [15] proposed a project for fire detection in South Korean Forest Fire Surveillance System (FFSS) which makes use of Minimum Cost Forwarding(MCFA) protocol [36] to sense humidity, temperature, and illumination to forward it to the base station node and then to the gateway. Hartung et al. [19] have used FireWxNet which is a multi-tiered
portable wireless system for monitoring weather conditions in rugged wild-land fire environments. FireWxNet provides the fire fighters with the ability to measure fire and weather conditions. Conrad et al. [7] have given a business case for the Enhanced Forest Fire Detection System with a GPS project in Pennsylvania. They proposed to create a fire detection system using fire sensors and GPS devices.

(C) Traffic Control

Wireless Sensor networks are being extensively used for vehicle traffic monitoring and control. These applications make use of either overhead or buried sensors to detect vehicles and control traffic lights. Furthermore, video cameras are frequently used to monitor road segments with heavy traffic, with the video sent to human operators at central locations. The wireless capability of sensors clubbed with their low cost can contribute to revolutionise the way in which traffic monitoring and patrolling can be done. Cheap sensors with embedded networking capability can be deployed at every road intersection to detect and count vehicle traffic and estimate its speed. The sensors will communicate with neighbouring nodes to eventually develop a “global traffic picture” which can be queried by human operators or automatic controllers to generate control signals. Another more radical concept [101] is to attach the sensors to each vehicle. As the vehicles pass each other, they exchange summary information on the location of traffic jams and the speed and density of traffic, information that may be generated by ground sensors. These summaries propagate from vehicle to vehicle and can be used by drivers to avoid traffic jams and plan alternative routes.

(D) Environmental Monitoring

Another major area of research is monitoring and control of the environment. The most beneficial aspect of this application is the ability to produce a big picture of the environment being monitored. For example, use of WSN to track the mating habits of seabirds. Similar to vehicles, sensors are directly attached to animals particularly to large mammals. This type of arrangement allows sensors to exchange information when animals are near to each other. Two sensor applications which have used this
approach are the SWIM project for monitoring whales and the ZebraNet project for monitoring Zebras[117]. Another application is monitoring river currents. The flow of currents in a river depends in part on the quantities and temperatures of water flowing from and to different tributaries. Positioning sensor nodes throughout the river can give the detailed information of the river currents and flow and mixtures of water from different tributaries. The information gained can also be used to track information about water ways.

(E) Medical Applications/HealthCare Systems

The use of Wireless Sensors in the arena of Medicine has brought a revolution in the diagnosis system as a whole.WSN specifically designed for medical applications are often referred to as Wireless Medical Sensor Networks (WMSN)[119]. They are being envisioned as medical devices that are implanted on a patient’s body and can be used to closely monitor the physiological condition of patients. They monitor the patient’s vital body signs (for example, temperature, heart rate, blood pressure, oxygen saturation etc.) and transmit the data in a timely fashion to some remote location without human intervention. A doctor can interpret the sensor readings to assess a patient’s condition. The application of the WSN in healthcare systems basically deals with monitoring of patients in clinical settings, home & elderly care center monitoring for chronic and elderly patients, collection of long-term databases of clinical data.

Table 2.1 presents the classification of each of the above explained application in the models described above.
Table 2.1: Nomenclature of WSN Applications

<table>
<thead>
<tr>
<th>Sensor Network Application</th>
<th>Event Driven Model</th>
<th>Periodic Model</th>
<th>Query based Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forest Fire Detection</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(alarm for sudden fire detection)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Military Based Applications</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(for sudden danger awareness as enemy enters in the territory)</td>
<td>(monitoring or tracking the enemies)</td>
<td>(monitoring or tracking the enemies)</td>
<td></td>
</tr>
<tr>
<td>Medical Applications</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>(Alarms for sudden issues in patient’s health eg: high blood pressure or low blood pressure alarm, sudden increase in heart beat)</td>
<td>(Patient monitoring to collect data periodically or constantly and send to the doctor when patient is at home)</td>
<td>(Patient monitoring to collect data periodically or constantly and send to the doctor when patient is at home)</td>
<td></td>
</tr>
<tr>
<td>Traffic Control Applications</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(traffic rule violation alert , eg: vehicle crossing the red light)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Environmental Applications</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>(sudden rainfall alert or temperature raise exceeds limit in any factory)</td>
<td>(to forecast about floods , volcanic eruption , rainfall etc)</td>
<td>(to check temperature , humidity pressure at any time)</td>
<td></td>
</tr>
</tbody>
</table>

2.2.4 Taxonomy of Routing in WSN

Routing is the process of identifying a route from a source to a destination node for transmitting the sensed data and is achieved either by computing all routes before and restoring them or computing them when needed [33,55,120,140]. During this process, at least one intermediate node within the inter-network is encountered. The main design objective at the network layer is to develop energy efficient routing protocols which can contribute to prolong the lifetime of the network and prevent connectivity degradation by employing aggressive energy management techniques. Figure 2.3 illustrates the taxonomy of routing protocols developed and categorised on the basis of network structure and protocol operation and each one of these is described in short as follows.
Flat Routing Protocols

Flat routing protocols make use of a huge number of sensor nodes which collaborate to perform sensing. Because of this huge number, these nodes are not assigned any particular identification (id) and hence each node plays the same role. This leads to the usage of data-centric routing approach [56] which follows the request response concept [64,30]. The protocols belonging to this category are SPIN [58], Directed Diffusion [39], Rumour Routing [29], MCFA, GBR, COUGAR and ACQUIRE [56].

Sensor Protocols for Information via Negotiation (SPIN) [58] protocol is amongst the early work which overcomes the disadvantages of traditional flooding and gossiping mechanisms by negotiation and resource adaptation. It works by negotiating for meta-data with its neighbouring motes instead of data itself. There are three messages defined in SPIN to exchange data between nodes namely ADV, REQ and DATA. ADV allows a sensor to advertise a particular meta-data, REQ requests the specific data and DATA message carries the actual data. In addition to the conventional SPIN protocol, several variations have been proposed in literature like SPIN with energy consumption awareness (SPIN-EC) [58], SPIN for broadcast networks (SPIN-BC), and SPIN with reliability (SPIN-RL) [68].

Directed Diffusion [39] is a highly energy efficient protocol. It uses the naming scheme to diffuse the data. This naming scheme defines attribute-value pairs for the data and queries the sensors when demanded by using those pairs only. In order to create a query, an interest is defined using a list of attribute-value pairs such as name of objects, interval, duration, geographical area, etc. It works in four phases namely interest propagation phase, gradient setup phase, reinforcement phase and data delivery phase to construct routes between the sink and the sensors of interest to the sink’s request.
Figure 2.3: Taxonomy of Routing Protocols
Rumour Routing [29] is a variation of directed diffusion which mainly works for contexts in which geographic routing criteria are not applicable. It stands between event flooding and query flooding and proceeds by routing the queries to the nodes that have observed a particular event rather than flooding the entire network to retrieve information about the occurring events. Simulation results have shown that this form of routing results in substantial savings of energy over event flooding and is also capable of handling node failure.

**Hierarchical Routing Protocols**

The need of hierarchical routing emerges from the fact that single-gateway networks are not capable of long-haul communication. These networks cannot support additional loads and cannot cover a larger area of interest [35]. Hierarchical or cluster based routing methods, originally proposed in wire line networks, operate by assigning special tasks to high energy nodes. These higher-energy nodes are used to process and send the information, while low-energy nodes are used to perform only the task of sensing in the proximity of the target. This task of clustering can greatly contribute to overall system scalability, lifetime, and energy efficiency by performing the tasks of data aggregation and fusion in order to decrease the number of transmitted messages to the sink node[68]. LEACH [86,135], PEGASIS[113], TEEN[2] and MECN[30] are some of the protocols following this strategy.

LEACH stands for Low Energy Adaptive Clustering Hierarchy Protocol[96,135]. It minimises energy consumption by cluster based operation. It dynamically selects sensor nodes as cluster heads based on incoming signal strength and forms clusters in the network. Cluster heads then directly communicate with the sink to relay the collected information from each cluster and saving the energy of rest of its cluster members which would otherwise have been used for communication. Cluster heads change randomly over time in order to balance the energy consumption of nodes. Selection of CH nodes is made by choosing a random number between 0 and 1. LEACH achieves as much as 70% reduction in energy dissipation as compared to direct communication and about 40%–80% as compared to the minimum
transmission energy routing protocol. Alike SPIN, many variations of this protocol such as[58,61] have also been proposed and implemented by researchers.

Another protocol, called Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [113], is a near optimal chain-based protocol. It is basically an enhancement of LEACH protocol and was proposed to improve it by addressing the overhead caused by cluster formation in LEACH. It works by constructing chains of nodes instead of clusters according to a greedy algorithm, where nodes select their closest neighbors as next hops in the chain. It is assumed that the nodes have a global knowledge of the network and the chain construction starts from the nodes that are farthest from the sink. As a result of this chain operation, instead of maintaining cluster formation and membership, each node only keeps track of its previous and next neighbour in the chain. PEGASIS achieves over a factor of 3 reductions in energy consumption in comparison to LEACH for different sizes and topologies. On the other hand, it introduces excessive delay for distant node on the chain and bottleneck because of single leader. A variation of PEGASIS, called Hierarchical PEGASIS[113] has also been proposed to reduce the delay incurred for packets during transmission to the base station.

Threshold Sensitive Energy Efficient Sensor Network (TEEN) [2] is another hierarchical protocol designed for event driven applications such as forest fire detection. As the name implies, multi-hop routes are generated according to a threshold related to sensory data, which is set by the application. This protocol works by organizing the sensor nodes into multiple levels of hierarchy including sensor nodes and cluster heads. In order to evenly distribute the energy consumption, the cluster heads are periodically changed within the cluster.

- **Location Based Routing Protocols**

In location based routing, sensor nodes are addressed depending on their locations. The distance between neighbouring nodes can be estimated on the basis of incoming signal strengths. Relative coordinates of neighboring nodes is obtained either by exchanging information between neighbor nodes or by directly communicating with a Global Positioning System (GPS) [146]. To save energy,
some location-based schemes demand that nodes should go to sleep if there is no activity. Protocols belonging to this family include GAF (Geographic Adaptive Fidelity), GEAR (Geographic Energy Aware Routing Protocol) and Span [54, 61, 68].

GAF was initially designed for MANETs but works equally well for WSNs [68]. It divides the given area into zones called grids. Sensors associate themselves with a point in the virtual grid with the help of GPS and collaborate with other sensors in that zone. GAF works in three states. The discovery state determines the neighbors in the grid; the active state reflects the participation in routing; and the sleep state describes when the radio is to be turned off. In order to handle mobility, each node in the grid estimates its time of leaving the grid and sends this to its neighbors. The sleeping neighbors adjust their sleeping time accordingly in order to keep routing fidelity. Before the leaving time of the active node expires, sleeping nodes wake up and one of them becomes active. GAF is implemented both for non-mobile (GAF-basic) and mobile (GAF-mobility adaptation) nodes.

Geographic Energy Aware Routing (GEAR) [47] is an enhancement of the traditional directed diffusion protocol as already described. It uses energy-aware and geographically informed neighbor selection heuristics to route a packet toward the destination region. The protocol works by determining the user’s interest as in directed diffusion but restricts them to only a certain region rather than to the whole network resulting into conserving more energy as compared to Directed Diffusion.

Span is a position based algorithm which works by selecting some sensor nodes to act as coordinators based on their positions. A particular sensor node can become a coordinator if two neighbors of this non-coordinator node are unreachable either through a direct link or via one or two coordinators (three-hop reachability). These coordinators then form a network backbone used to forward messages. New and existing coordinators are not necessarily neighbors in [30], which in effect makes the design less energy-efficient because of the need to maintain the positions of two or three-hop neighbors in the complicated Span algorithm.
• **Negotiation Based Routing Protocols**

This category of protocols makes use of high level descriptors in order to eliminate redundant data transmissions through negotiation. The prime advantage of using negotiation-based routing in WSNs is to restrain duplicate information and prevent redundant data from being sent to the next sensor or to the base station by conducting a series of negotiation messages before the real data transmission begins. The SPIN family of protocols (SPIN-P, SPIN-BC, SPIN-EC etc.) [58] are few negotiation based protocols.

• **Multi-path Based Routing Protocols**

This type of routing uses multiple paths rather than a single path in order to enhance network performance. Multiple path leads to increased fault tolerance, improved network reliability between the source and destination at the expense of increased energy consumption, traffic generation and cost of maintaining alternate paths[3,80]. Alternate paths are kept alive by sending periodic messages. This type of routing is able to meet three objectives: path discovery, path distribution and path maintainence. Directed Diffusion [30] is an example of Multipath routing Protocol. Few other multipath routing protocols being described below:

Braided Path Routing [3] works by initially computing a primary path for the packets to be routed. Then, for each sensor node on this primary path, the best path from a source sensor to the sink that does not include that node is computed. The best alternate paths may not be disjoint from the primary path and are called idealized braided multipaths. Moreover, the links of each of the alternate paths lie either on or geographically close to the primary path. Therefore, the energy consumption on the primary and alternate paths is likely to be comparable as opposed to the scenario of mutually ternate and primary paths. The braided multipath can also be constructed in a localized manner in which case the sink sends out a primary-path reinforcement to its first preferred neighbor and alternate-path reinforcement to its second preferred neighbor.
Reliable Energy Aware Routing (REAR) [119] considers residual energy capacity of each sensor node in establishing routing paths and supporting multi-path routing protocol for reliable data transmission. In addition, REAR allows each sensor node to confirm the success of data transmission to other sensor nodes by supporting the DATA-ACK oriented packet transmission [30].

- **Query Based Routing Protocols**

  As the name suggests, in these protocols, destination node(s) forwards a query for data to a sensing node through the network [56,64]. The node which has the data matching the query then sends the data back to the node that initiated the query. Usually these queries are described in natural language or high-level query languages.

  Directed Diffusion and Rumour Routing protocols described above fall under the category of Query based Protocols.

- **QoS Based Routing Protocols**

  This category of routing protocols ensure that while delivering data to the base station, the network satisfies certain QoS metrics like delay, energy, bandwidth etc. Some of the protocols which fall under this category are described in subsequent paragraphs.

  Sequential Access Routing (SAR) [30] is one of the first routing protocol which introduced the concept of QoS into routing decisions. SAR works by calculating a weighted QoS metric as the product of the additive QoS metric and a weight coefficient associated with the priority level of the packet. SAR minimizes the average weighted QoS metric throughout the lifetime of the network. To meet the problem of changing topology, base station pre computes the triggered path periodically.
SPEED protocol [127] is particularly used for real time communication. It requires each node to maintain information about its neighbours and uses geographic forwarding to find the paths. In addition, SPEED strives to ensure a certain speed for each packet in the network so that each application can estimate the end-to-end delay for the packets by dividing the distance to the sink by the speed of the packet before making the admission decision.

- **Coherent Based Routing Protocols**

  This category of routing protocols takes into consideration the node processing ability of the sensor units. In coherent routing, the sensor nodes which sense the data only timestamp it and forward the raw data to the aggregators. On the other hand in case of non-coherent routing[43], the data is first processed locally and are forwarded later.

However due to both wireless communication effects and the peculiarities of sensor networks, the process of routing has become a complicated issue. Further, routing protocols in WSNs also differ depending on the application and network architecture. Next section presents the issues and prominent challenges that are still prevailing in WSN especially in routing domain.

### 2.2.5 Challenges in Routing in WSN

Despite the fact that researchers have been putting efforts to improve the routing protocols, still study of literature revealed that there are many unfolded challenges dominating in routing in WSN. Few important ones are discussed below.

- **Deployment of Nodes**

  The topological deployment of sensor nodes is application specific which in turn substantially affects the performance of routing protocol. In case of deterministic deployment, the motes are manually placed and data is routed through pre-determined paths. However; in non-deterministic deployment, nodes are scattered
randomly creating an adhoc infrastructure [54,61]. In such an infrastructure, the position of the sink or the cluster-head is also crucial in terms of energy efficiency as inter sensor communication is between short transmission ranges. When the distribution of nodes is not uniform, optimal clustering becomes a vital issue to enable energy efficient network operation. Due to bandwidth limitations a multiple hop route is preferred.

- **Energy Preservation**

One of the major limitations of sensor nodes is the limited energy supply. The major portion of residual energy of these resource constraint miniature devices is consumed in discovering the neighboring mote and performing the tasks of both computations and transmitting information in a wireless environment. Further, as the transmission power of a wireless radio is proportional to distance squared or even higher order in the presence of obstacles, multi-hop routing consumes less energy than direct communication. However, multi-hop routing introduces significant overhead pertaining to topology management and medium access control. As such, formulating energy-conserving forms of communication and computation are essential especially in a multi hop environment [16,46].

- **Node/Link Heterogeneity**

Heterogeneous nodes such as temperature sensors, pressure sensors, and oxidant sensors are often deployed in WSN to meet the application’s needs. This heterogeneous set of sensors can generate readings and report data at different rates, lead to diverse quality of service constraints and even follow multiple data reporting models. Dealing with such a diverse mix of sensors makes the process of routing more complex and challenging [51,63].

- **Fault Tolerance**

The routing protocol designed for WSN should be robust enough to handle the worst case functioning of the network. The sensor units can fail due to various
reasons such as malfunctioning hardware, software glitches, dislocation or environmental hazards, for example, fire or flood. In such a case, medium access control (MAC) and routing protocols together must accommodate formation of new links and route to the data collection BSs. This may require rerouting packets through regions of the network where more energy is available or actively adjusting transmitting powers and signaling rates on the existing links to reduce energy consumption. Therefore, multiple levels of redundancy may be needed in a fault-tolerant sensor network [118].

- **Scalability**

  Usually, in a non-deterministic environment, sensor motes are required to monitor the whole environment. For example, for habitat monitoring or forest fire detection, order of hundreds or thousands or more number of sensors are deployed. The routing protocols should be scalable enough to support the growing size of network [56].

- **Network Dynamics**

  Network dynamics refers to the study of topological changes in the network. For few applications, static sensor nodes and hence simple routing may suffice however; for applications that require mobile sinks or cluster-heads, routing messages amongst the moving nodes is challenging in addition to other factors such as energy, bandwidth etc. Monitoring static events allows the network to work in a reactive mode i.e. simply generating traffic when reporting while dynamic events require periodic reporting and consequently generate significant traffic to be routed to the sink [46].

- **Connectivity**

  Connectivity implies ensuring that all the nodes within the network are connected in such a manner that the data can traverse from source to destination. Now since a WSN is usually a dense sensor network therefore it is apparent that most of the
nodes stay connected. These connections should further allow dynamic network topology and should pose no shrinking affect due to node failures. Further a routing protocol is expected to ensure maximum connectivity even when distribution of nodes is random [24].

- **Coverage**

Because of the hardware constraints, a typical sensor is limited in its sensing range and can thus cover only a limited physical area of the given environment. This property of sensor nodes makes optimal coverage a design parameter for any of the routing protocol[106].

- **Quality of Service**

In certain applications, timely delivery of the sensed data is as crucial as its accurate delivery. Data which is not delivered in a stipulated period of time from the moment it is sensed becomes useless. Therefore, bounded latency for data delivery is another parameter for time-constrained applications. On the other hand, in many applications, conservation of energy, which is directly related to network lifetime, is considered relatively more important than the quality of data sent. As energy is depleted, the network may be required to reduce the quality of results in order to lengthen the total network lifetime. Hence, energy-aware routing protocols are required to ensure quality of service [31].

An in-depth analysis of the routing protocols and their applicability in various application scenarios reveal the fact that though extensive research has been done in the field of deterministic WSN, the researchers have almost remained silent in the case of non-deterministic environment. Further, researchers have been spending time in exploring and exploiting an energy efficient alternative to the conventional client server paradigm used for routing but to the best of our knowledge, none of the above listed protocols is found to be best suitable for all applications associated with non-deterministic environment. *The deficiencies of this traditional paradigm like excessive*
consumption of bandwidth, traffic on the network channel, packet delays etc. are of utmost concern especially for mission critical applications.

A special class of software agents called mobile agents seems to be a promising solution to the above stated problems because of their properties like autonomy, pro-activity, reactivity, intelligence etc. and their adaptability to a distributed environment.

The subsequent section presents an in-depth study of software agents in general and mobile agents in particular.

### 2.3 SOFTWARE AGENTS

The term agent comes from greek ‘agein’, which means to drive or to lead [40]. Though the term has a very broad scope, it has been explored by computer science community to describe current trends in computer science and develop programming techniques and software that enable a more active role of the computer. In the domain of computer science, it is thus termed as software agent. Defining it more precisely, “Software Agent is an object of the environment that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators” [74]. These special software entities carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user's goals or desires [104].

![Figure 2.4: Agent Interaction Environment](image-url)
Though the concept of Software Agent looks much like the conventional objects of OOP, it differs from them in terms of the degree of autonomy or the *self-initiating-property*.

### 2.3.1 Evolution and Behaviour

The concept of software agents owe its roots in the Distributed Artificial Intelligence (DAI) research which was conducted about 3 decades ago. Carl Hewitt proposed an Actor system [74] where each Actor had an explicit internal state and had the capability to respond to the messages of various other Actors. The subsequent years focused on the more theoretical aspect of bringing intelligence to software agents. The last two decades have seen a huge expansion of systems to solve practical problems drawing on changes in distributed processing, object-oriented programming, the Internet, the Web and the increased digitization of information and services. Study in this domain reveals that the agent paradigm is an extension rather than a replacement of conventional systems that are either object-oriented or component-based. The software Group at MIT compares and contrasts software agents to conventional software and highlights the differences to be as "*Software agents differ from conventional software in that they are long-lived, semi-autonomous, proactive, and adaptive*"[104]. The realm of software agents has fascinated the researchers to a great extent because of its vast set of primary and secondary properties which make it stand ahead of its counterparts. This section describes in detail the various features of agents.

- **Social Ability**

  It is the ability of software agents to communicate with other agents that constitute a part of its environment [74]. The communicating agents may work towards a single global goal or separate individual goals.
• **Autonomy**

Autonomy refers to self-governance[1]. According to this property, an agent can function on its own without the need of human guidance or any external elements. They have control on their own actions and internal states.

• **Reactivity**

Agents can distinguish their environment and respond in a timely fashion to the changes that occur. Pure reactive agents do not have any internal symbolic models of their environment and they act using a response type of behavior by responding to the present state of the environment in which they are embedded. The agents have no goals and even no internal states.

• **Adaptability**

Agents can adapt to changing environment and can set up their own goals based on their implicit purpose[26]. They attain and process situation information, both spatially and temporally.

• **Intelligence**

Intelligence is the property when a software agent is able to incorporate knowledge-based technology and act proactively on perceiving the dynamic state of its environment [74]. It is this ability of software agents which make them unique and an energy efficient solution in a given distributed environment.

• **Learning**

The property of intelligence helps software agents to learn by their experience and adjust their future action sequences and behavior so as to avoid future mistakes [26]. Hysteretic agents act based on perceptions and also past experience.
• **Pro-activity**

Pro-activeness or self-starting capability refers to the ability of agents to take the initiative rather than acting simply in response to their environment.

• **Goal-oriented**

Agents should exhibit goal driven behavior that their action will cause beneficial changes to the environment.

• **Mobility**

A mobile agent can migrate between various machines to perform assigned tasks. Mobility is neither a necessary nor a sufficient condition for agent-hood.

The above listed properties of software agents distinguish them from regular objects and expert systems. Table 2.2 gives a tabular comparison of agents, objects and expert systems.

**Table 2.2 : Comparison of Agents, Objects And Expert Systems**

<table>
<thead>
<tr>
<th>Entities Properties</th>
<th>Agent</th>
<th>Object</th>
<th>Expert system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Autonomous</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Reactive</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Adaptable</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intelligent</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Goal Oriented</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mobile</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learning Ability</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Pro-Activity</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

However, an agent may or may not possess all the above-mentioned characteristics for acting in an agency. They can in fact, function efficiently even by possessing some
of the features. The taxonomy of agents, depending upon the properties they possess is given in the next section.

2.3.2 Taxonomy of Software Agents

This section describes the taxonomy of software agents on the basis of the properties which they possess. The various types of agents are shown in Figure 2.5

- **Collaborative Agents**

These agents are autonomous, reactive, social and pro-active in nature. They can rationally act in a given open and time-constrained multi-agent environments in an autonomous manner. They are static and large coarse-grained agents which can together create a system that interconnects separately developed collaborative agents, thus permitting the group to function beyond the capabilities of any of its member.

![Figure 2.5: Taxonomy of Software Agents](image)

- **Interface Agents**

These agents are autonomous and good learners. They also have communication ability and communicate with the user instead of other agents. Their cooperation with other agents, if any, is limited typically to asking for advice. They learn to
better assist their user by observing and imitating the user, receiving feedback and explicit instructions from user and asking other agents for advice. They try to perform some direct manipulation tasks in order to accommodate novice users. The major motivation behind the design of such agents is to eliminate the tedium of humans in performing several manual sub-operations.

- **Mobile Agents**

Mobile agents are capable of executing on different machines in a dynamic networked environment, and sense and (re)act autonomously and proactively in this environment to realize a set of goals or tasks. They are thus autonomous, social and adaptable mobile entities. A Mobile agent approach trades server computation and cost for savings in network bandwidth and client computation. This approach is advantageous when the server's CPU is not a bottleneck. These types of agents provide a natural development environment for implementing free market trading services. The flexible distributed computing architecture and mobile agents provides a radical and attractive rethinking of the design process[65].

- **Information Agents**

Also known as Internet agents or Internet Softbots, these agents act as tool to manage information explosion. They collect, manage, and manipulate information from various distributed sources. These agents have the primary properties of being autonomous and social. In addition, they also have varying characteristics: they may be static or mobile, non-cooperative or social, may or may not learn.

- **Reactive Agents**

These agents react in a stimulus-response manner to the present state of the environment in which they are embedded. A reactive agent is viewed as a collection of modules, which operate autonomously and are responsible for specific tasks. Communication between the modules is minimized when using
these agents. They tend to operate on representation, which are closed to raw sensor data. They are autonomous and social but unintelligent.

- **Proactive Agents**

  Unlike reactive agents who just react to situation as directed, proactive agents are able to reason on the changes in environment, hence can change their intentions and beliefs, can change plan of action and execute the actions.

- **Hybrid Agents**

  Hybrid agents refer to agents which consist of two or more agent philosophies. Hybridism usually translates to ad hoc or unprincipled designs. Many hybrid architectures tend to be very application-specific.

- **Smart Agents**

  As the name suggests, these types of agents possess almost all the features of agents including autonomy, social ability, proactivity, adaptability, intelligence, ability to learn, goal oriented and ability to travel from one environment to the other. However, no agent today belongs to this class of agents. Agents of such form offer many open research issues such as standardization of agent oriented technology, infrastructure and technology needs to be established before real smart agents can be developed and deployed.

- **Competitive Agents**

  Competitive agents are autonomous, social, reactive and proactive. They are quite similar to collaborative agents except that they may compete with other agents in order to perform tasks for their owners. They may not perform task for other agent request if it may be detrimental to its objectives.
Considering the above significant types of agents, it is evident that over the years the big community of information have been restructured into smaller elements called agents that are now possessed to have many abilities well suited to handle complex applications. In other words, a group of software agents can be treated as a dynamic collection of simple agents with self-describing interfaces who while progressing through their life span continue to collect the information in a “knowledge soup,” and same is presented to the user according to the application in which these agents operate [108]. In such an environment, individual agents do not manage the entire system to achieve the target where as they would interact and collaborate actively forming a multiagent system (MAS) [74]. In such a system agents integrate to provide the needed information as discussed in detail in the next section.

2.3.3 MultiAgent Systems (MAS)

A system in which a community of software agents works in collaboration and coordination with each other to achieve a common goal is called a Multi-Agent System. It has evolved as a result of sociological relationships among agents [1]. In MAS, at least one agent has an ascribed set of goals and other agents in the system may adopt the goal. The goal describing agent(s) is referred as the Coordinator or parent agent and the goal adopting agents are called as Sociological Agents as shown in Figure 2.6.

Once an agent adopts the goal of goal describing agent, a relationship is created between the two. If the goal cannot be achieved at this level it may again be forwarded to different homogeneous or heterogeneous adjunct agents. More
precisely, MAS can be defined as a loosely coupled network of problem solvers that work together to solve problems that are beyond the individual capabilities of knowledge of each problem solver.

A typical MAS inherits most of the advantages of distributed intelligence over centralized and sequential processing. Table 2.3 delineates the distinguishing characteristics of MAS.

A MAS comprises of variety of homogeneous as well as heterogeneous agents having different behavioral aspects. In a MAS, agents are usually classified on the basis of their behavior and locomotive ability [105]. Since the behavior of software agents is observed within MAS, the agent community refers the former group of agents as internal agents.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic</td>
<td>Agents in MAS form dynamic groups to solve specific problems, pool together resources and disband after the problems are solved releasing resources to local usage.</td>
</tr>
<tr>
<td>Robust</td>
<td>MAS is more reliable and fault-tolerant as compared to individual agents serving a goal.</td>
</tr>
<tr>
<td>Concurrent</td>
<td>A MAS can make member agents reason and perform system tasks in parallel and asynchronously, resulting in faster and flexible execution of tasks.</td>
</tr>
<tr>
<td>Adaptive</td>
<td>Agents in MAS can re-configure themselves to suit system changes such as noise, resource allocation and faults without disturbing the entire system.</td>
</tr>
<tr>
<td>Scalable</td>
<td>Agents can be added or deleted without greatly disrupting the system.</td>
</tr>
</tbody>
</table>

In the same manner, since the locomotive ability is noticed when the agents communicate outside the system, later group of agents is referred to as external agents. The communication between internal and external agents is made possible through mediator agents. The categories of agents listed above are described in short as follows:
A) Internal Agents

Internal Agents are local to a particular architecture and are classified according to the role being played within the environment i.e. how they behave inside the system and with each other. Figure 2.7 illustrates the classification of internal agents in a multiagent system. Within a MAS, agents can be cooperative (collaborate with each other and share some common goals), self-interested (with distinct goals), competitive (possess mutually exclusive goals), destructive (intentionally provide wrong information), interface (takes the input and ultimately deliver the output by delegating the task to other agents), task-oriented Agents (the lowest level of agents), reactive (reacts as directed) and proactive (may decide on its own to improve the probability of success).

![Figure 2.7: Classification of Internal Agents](image)

B) External agents

External Agents are also referred to as dynamic agents as these possess the ability to change their residing locations. These agents move out of a system to perform a task and these may or may not return to the initiating point. External agents fall into two categories namely mobile agents and ants. The agents which move out to different locations to gather the desired information for carrying out a task and then returning to originating node are termed as mobile agents [104]. Mobile agents spread intelligence across networks. Ongoing research in the domain of mobile agents reveals the fact that these agents have the potential to provide an energy efficient
solution in a networked environment. In contrast to mobile agents ants are special external agents which originate at one node, keep on changing their locations and may die on any other node [74]. Ants follow the principle of “STIGMERGY”. It is a form of indirect communication through an environment. The insect ants when move in search of food stimulate a hormone named as pheromones, which attracts other surrounding ants. In the same fashion, the routing of an ant-based agent is pheromone distribution dependent where pheromone distribution depends upon the environment in which the agent’s properties are utilized. Swarm intelligence stems from the work of ants in which unintelligent internal and external agents possibly belonging to heterogeneous platforms, work independently or with relatively small amount of collaboration to achieve a greater goal that requires intelligence.

C) Mediator Agents

As the name suggest, mediator agents provide an interface between internal agents and external agents. These agents come into picture when one category of agents needs support from the other category of agents. In other words, mediator agents are also called as matchmaking agents as these facilitate the communication between service requester and provider.

Current research work more dominantly employs the mobile agents to improve the efficiency of WSN. Therefore the next section provides an overview of mobile agents in detail.

2.3.4 Mobile Agents

Mobile agents are a distributed computing paradigm. As already described in section above, a special class of software agent which is mobile in nature i.e. which can migrate and execute on different machines in a dynamic networked environment is known as Mobile Agent[43]. Like any other agent, it senses and (re)acts autonomously and proactively in a given environment to realize a set of goals or tasks. A typical mobile agent can migrate from one machine to another under its own control and can suspend execution any time.
The inception of mobile agent paradigm for communication is a result of limitations/weaknesses in its predecessor paradigms. One of the first paradigm proposed in this light was the message passing paradigm [74] which lets processes communicate by explicitly sending and receiving messages. Both asynchronous and synchronous message passing is being used for communication. However, literature survey reveals that developing distributed applications based on message passing primitives is a very complex and error-prone task, and programs are hard to analyse and debug.

Next paradigm that came into existence was remote procedure call (RPC). Through this concept, the research community successfully introduced the concept of mobility of data in a given network. In this form of communication, all processes call remote procedures rather than explicitly sending and receiving messages. An RPC supports client/server-style of interactions in which clients issue requests to servers, which execute the requested procedure and then return the results. Like message passing paradigm, many RPC mechanisms support both asynchronous and synchronous calls. The success of RPC inspired the researchers to move code along with the data. A piece of program was sent to another machine and executed there. This is called remote evaluation (RE) if the sender starts the action, while it is called code on demand (COD) if the receiver does it.

While remote evaluation only allows for ‘code mobility’, the concept of a mobile agent moved a step ahead and provided a support for ‘process mobility’, i.e., program executions may migrate from node to node of a computer network[99]. Obviously, for migrating agents not only code but also the state information of the agent has to be transferred to the destination.

An agent’s state is further subdivided into data state and execution state. While the first includes the agent’s global variables and instance variables, the later comprises the local variables and the active threads. These two types of migration of the mobile agent are termed as weak migration and strong migration respectively and are shown in the Figure 2.8.
In case of strong migration, the underlying system captures the entire agent state (consisting of data and execution state) and transfers it together with the code to the subsequent location. Once the agent is received at its new location, its state is automatically restored. Though this type of migration captures, transfers and restorates the complete agent state completely transparently by the underlying system, providing this degree of transparency in heterogeneous environments at least requires a global model of agent state as well as transfer syntax for this information. Moreover, the given agent system must support functions to externalize and internalize agent state. Only few languages such as Facile and Tycoon [74] allow externalizing state at such a high level. It was also realized that transferring the complete agent state can be cumbersome, particularly for multi-threaded agents and thus strong migration might be a very time-consuming and expensive operation. These difficulties thus led to the development of the so-called weak migration scheme, where only data state information is transferred. The size of the transferred state information can be limited even more by letting the programmer select the variables making up the agent state. As a consequence, the programmer is responsible for encoding the agent’s relevant execution states in the program variables. While this method may substantially reduce the amount of state to be communicated, it puts additional burden on the programmer and makes agent programs more complex. Table 2.4 compares and contrasts the various entities described above on the basis of their mobile units.
Table 2.4: Mobile Units in Different Paradigms

<table>
<thead>
<tr>
<th>Message Passing</th>
<th>RPC</th>
<th>Remote Evaluation</th>
<th>Code on Demand</th>
<th>Mobile Agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Code</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Execution State</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Examples</td>
<td>Smalltalk</td>
<td>Java</td>
<td>Java servlets</td>
<td>Java applets</td>
</tr>
</tbody>
</table>

2.3.5 Benefits of Mobile Agents

The use of mobile agents has simplified the implementation of many applications in a networking environment. The various advantages incurred by using them are as follows:

- **Reduction of Communication**

  The use of mobile agents reduces communication with respect to latency, bandwidth and connection time at the expense of minimal overhead for sending agent code and execution state across the network. Communication latency is reduced by sending an agent with a sequence of service requests across the network rather than issuing each service request by a separate remote procedure call. Similarly, communication bandwidth is controlled by migrating the agent across the network in order to deliver instructions for the generation of data on a remote host. It also gets reduced by moving the agent across the network to the source of data in order to reduce the data before transmission. An example for the reduction of communication by mobile code is the NeWS window system[144] where clients communicate with the display server by sending brief PostScript programs instead of drawing a grid by sending several thousand messages for individual points.
• **Asynchronous tasks**

With mobile agent technology, the client part of the application can be transferred from the mobile device to stationary servers in the network. From an end user’s perspective, not only individual requests but the entire task is moved to the network, where it is performed asynchronously.

• **Dynamic Protocols and Intelligent Data**

Mobile agents permit dynamic protocols, i.e. new protocols to be installed automatically and only as needed for a particular interaction. To receive an agent initially, the client and server must share some standard protocol. Once the agent is running, though, it can use a specialized protocol for communication back to its home server. Further, an executing agent can communicate repeatedly with the server without intervention from the user, allowing the construction of dynamic services. For example a news service could transmit news updates to agents on distributed clients by using a special multicast protocol. A recent example of intelligent data is the MPEG4 compression standard for video, where the decompression algorithm is bundled with the data. This approach makes the standard highly flexible and allows the upgrade to use improved compression techniques.

• **Software Deployment**

Mobile agents can contribute to automate the software installation and updating process. These mobile entities are capable gathering information about the environment, query the user for installation preferences, configure the system, create directories and uncompress and compile the software. However, this approach to software deployment has its limitations since it might not be possible to capture every special case and error condition of the installation process and the programming of suitable deployment agents might become very difficult. A better approach to software deployment would be to
use the agent language itself, since the agent language is in particular designed to prevent such damage.

- **Temporary Applications**

  In addition to deploying software packages, the agent could be the application itself. In many of the cases, an application-agent might be self-contained and has no communication or migratory needs at all. It is much smaller than a stand-alone application since it could exploit the infrastructure provided by the mobile agent system. Examples of such temporary applications can be travel guides and route planners downloaded on a mobile computer for a particular trip and discarded afterwards. Upon arrival at a new location, the user might temporarily download services that are specific to the new environment. Java based applets are also the examples of applications of mobile agents.

- **Distributed and Heterogeneous Computing**

  Mobile agents can also serve as the basis for general-purpose distributed and heterogeneous computing. They provide the necessary infrastructure for communication between the tasks in a heterogeneous environment. The agent system furthermore supports the independent compilation and initiation of agents so that further agents can be assigned to a task at runtime. Prospective applications for agent-based distributed computing are parallel algorithms with a reasonable low communication overhead compared to its computation requirement and particle or object based simulations.

2.3.6 **Application Areas of Mobile Agents**

The benefits of using mobile agents in any networked environment have attracted researchers to implement them in various application domains. The section discusses the key application areas of mobile agents such as information retrieval, E-commerce, network management, load management, just to list a few.
• **Information Retrieval**

In information retrieval applications, mobile agents typically visit several nodes of the network in search for given information. The number of sites to be visited, often called itinerary, can be either statically defined at the agent creation or dynamically built from the information the agent collects during its travel. For example, when one agent visits several Web servers following the found links, its itinerary is dynamically generated by an agent-based solution which provides one or more agents that visit WWW servers searching for interesting pages [104].

• **Electronic Commerce**

One of the most attractive applications of the mobile agent technology is electronic commerce. In such cases, the network nodes model virtual marketplaces and mobile agents well suit to model buyers and sellers that roam through a network to carry out exchanges of goods, services and money. A buyer (mobile)agent can travel from site to site to act in behalf of a user who wants to buy goods. The “intelligence” of the agent can be used to compare determine the cheapest service available in the marketplace and also to coordinate different services when required.

• **Network Management**

The network today is a complex set of resources and services available for the applications. The use of mobile agents has provides a more scalable and reliable management model, because the execution of an agent occurs autonomously and locally to the devices/resources it is devoted to manage, without being affected by the network latency and by possibly intermittently network connections. Also, mobile agents suit both low-level and high-level network management issues [79]. These mobile entities can be executed onto specific devices to monitor, control and program and are also enriched with the necessary “intelligence” to deal with high-level
application issues required by today’s networks, i.e., the management of services and data available to applications.

- **Load Management**

Mobile Agent technology provides a cost effective solution to the problem of distributing the load in a set of computational entities. In such cases, multi-agents are used to decentralize the distribution of the computational load [75]. In fact, a complex application can be divided into autonomous parts, each of which delegated to a mobile agent. Each mobile agent is in charge of searching for the most convenient node of the network, where to execute its own part of code. During the execution, agents can move to other nodes where more computational resources are available, in order to better distribute the load[144]. The case of load balancing applications is however quite different from other applications because it requires that the application is restored exactly as it was before the movement of agents, because it must be transparent to the application itself. It thus makes use of strong mobility mechanism for the same, which grants that also the execution state is transferred and resumed at the destination node.

Considering the above listed applications of mobile agents in various domains, a survey was carried to explore the feasibility of employing mobile agents in WSN so that current research work could proceed. Next section hence explores the same.

### 2.3.7 Exploring the Feasibility of Mobile Agents in WSN

The applicability of mobile agents in multiple domains attracted researchers working in WSN. One of the initial works was that given by Chen et al.[75,78] in which mobile agents were used for dissemination. The authors also gave various design issues for using mobile agents in WSN[77]. Thereafter the research community exploited these distributed autonomous mobile entities for aggregation in addition to data dissemination[76]. Over the years, mobile agents have been used for energy balancing, extending the lifetime of the network etc. The use of mobile agents for one
or the other aforementioned tasks in WSN also depends upon the application. With its exclusive set of features, mobile agent technology was thus found to be a promising solution to the resource constrained network of wireless sensors, especially when deployed in non-deterministic environment. The main factor which encourages the integration of mobile agents with wireless sensors in a given network is that they reduce communication cost. Mobile agents have increased flexibility provided by mobility and the agent itself can be send to the server for direct computation. Large amount of raw information transferred in order to determine their relevance can be very time consuming and clog of the networks. Mobile agent approach trades server computation and cost for savings in network bandwidth and client computation. Extensive simulation-based comparisons between Mobile agent based WSN (MAWSN) [75] and client/server based WSN (CSWSN) has revealed that, depending on the parameters, MAWSN considerably reduces the energy consumption while conditionally improving the end-to-end delay. Majority of the work exploited the ability of mobile agents (MA) to carry processing codes that allow the computation and communication resources at the sensor nodes to be efficiently harnessed in an undefined area. These intelligent units are capable of adjusting their behaviors depending on quality of service needs (e.g. data delivery latency) and the network characteristics to increase network lifetime while still meeting those quality of service needs. The basic advantage(s) of mobile agent technology in comparison to the client/server technology are given below:

- **Scalability**

  The use of mobile agents supports scalability of sensor nodes in a network. Agent architectures that support adaptive network load balancing could do much of a redesign automatically [79,80].

- **Reliability**

  Mobile agents can be sent when the network connection is alive and return results when the connection is reestablished. This asynchronous working of mobile-agent-based computing model is not affected much by the reliability of the network [110].
- **Extensibility and Task Adaptivity**
  Mobile agents can be programmed to carry different task-specific integration processes which extend the functionality of the network.

- **Energy Awareness**
  The itinerary of the mobile agent is dynamically determined based on both the information gain and energy constraints. It is tightly integrated into the application and is energy efficient.

- **Progressive accuracy**
  A mobile agent always carries a partially integrated result generated by nodes it already visited. As the mobile agent migrates from node to node, the accuracy of the integrated result is constantly improved assuming the agent follows the path.

The survey presented above unveiled the fact that there are numerous software agents which contribute to achieve energy efficient data dissemination in sensor networks. The current body of research focuses on mobile agents to achieve a secure and efficient communication. Mobile agents are mainly characterized by autonomy, adaptability and mobility. The main factor which encourages the development of mobile agents is that they reduce communication cost. Mobile agents have increased flexibility provided by mobility and the agent itself can be send to the server for direct computation. Large amount of raw information transferred in order to determine their relevance can be very time consuming and clog the networks. Mobile agent approach trades server computation and cost for savings in network bandwidth and client computation. The approach is advantageous when the server's CPU is not a bottleneck. It gives performance optimization for distributed operations that involve heavy network delays and/or weak connectivity; extended autonomy in terms of existing support for asynchronous execution and disconnected operations. It provides a natural development environment for implementing free market trading services. The flexible distributed computing architecture and mobile agents provides a radical and attractive rethinking of the design process.
2.4 CONCLUSION

The chapter provided the motivation behind the research work being carried out in this thesis. It initially detailed the basic concepts of wireless sensor networks and thereafter focussed on presenting the properties of software agents and mobile agents, in particular that make them suitable to be employed in WSN. Next chapter presents the literature survey by exploring the contribution of eminent researchers and the drawbacks of fundamentals protocols deployed in WSN laying the foundation for current research work.