CHAPTER 2

AGENTS IN WSN

2.1 Wireless networks

Wireless networks deliver a lower bandwidth than wired networks; the mobility of the hosts, which causes topological changes in the underlying network increases the network information. The limitation of power also leads the user to disconnect the mobile unit frequently to save power (Sklar 2001).

Transport layer routing, which uses the Transmission Control Protocol (TCP), and transport connections in wireless networks are plagued by problems such as high bit rates, frequent route changes and partitions (Black 1999). While running the wireless protocol over such networks, the throughput of the connection is viewed to be poor, because this protocol takes the lost or delayed acknowledgements as congestion. The wireless transmission protocol needs more effective mitigation strategies.

The effect of channel conditions on the wireless protocols performance should be overcome, and there is a need to provide a robust, flexible protocol, that consistently gives high performance in a variety of network environments. In this work, the performance of the wireless protocol is improved when multi agents are implemented along with the existing wireless protocol.

The wireless communication revolution has brought fundamental changes in data networking, and telecommunication, and is making integrated
networks a reality. A wireless network consists of mobile or stationary nodes, which can communicate with each other over the wireless links without the aid of any established infrastructure or centralized administration (Sklar 2001).

Each node has the capability to communicate with another node in its vicinity, which forwards the data packets to the designated node. These types of networks are useful in any situation where a temporary network connection is needed, as in the case of disaster relief or on the battlefield. These systems depend on the completeness and timeliness of flooding to identify the available resources, discover efficient paths and coordinate in-network computation.

The wireless link losses have a drastic adverse impact on the wireless protocols performance, due to the difficulty in distinguishing congestion losses from wireless link losses over wireless networks. In WSNs, the nodes are mounted on vehicles for vehicular applications or carried by human beings. These networks concentrated on improving the MAC layer protocols and routing protocols.

WSNs are multihop wireless networks, which consist of a finite number of radio equipped nodes that are autonomous. Each node has a transmission radius, and is capable of transmitting a message to all its neighbours that are within the transmission range. Flooding applications include paging a particular host or sending an alarm signal, which is also used for route discovery in a source initiated on-demand routing. In a one-to-all network, the transmission of packets by a source node will reach all the nodes that are within the predefined radius or range, and in one-to-one network, each transmission is directed only to one neighbour in that network (Prasad et al 2000).
2.2 Wireless Protocol Issues

In order to efficiently use the WSN for real-time applications, the issues related to the wireless protocols are reduced. The following issues have to be solved when the wireless protocol is used in WSNs.

i. Bit Error Rate
ii. Route Recomputation
iii. Network Partitions
iv. Multipath Routing
v. Latency
vi. Window Translation Rate

2.2.1 Bit Error Rate

This is a measure of how many packets get corrupted, resulting in lost wireless protocol data segments or acknowledgements. Repeated errors make the congestion window at the sender remain small, resulting in low throughput.

2.2.2 Route Recomputation

When an old route is no longer available or a route is failed due to link failure, the network layer at the sender attempts to find a new route to the destination. When a packet is not received by the destination, it has to be retransmitted. The wireless protocol sends time out, retransmits a packet and invokes congestion control. The route recomputation will increase the reliability of the network.
2.2.3  *Network Partitions*

If the sender and receiver of a wireless protocol connection lie in different partitions, all the sender’s packets get dropped by the network, resulting in the sender invoking congestion control.

2.2.4  *Multipath Routing*

This minimizes the frequency of route re-computation, and results in a significant number of out-of-sequence packets arriving at the receiver.

2.2.5  *Latency*

The high rate links lead to a large bandwidth delay product, which may disrupt the dynamic response to variations in the link quality. The feedback mechanisms in the wireless protocol do not respond well to a large bandwidth delay product.

2.2.6  *Window Translation Rate*

In a wireless protocol session through a bottleneck link, the congestion window is given by,

\[ W = R \cdot \Delta + \beta \]  \hspace{1cm} (2.1)

where,

- R - fair share of the bottleneck rate
- \( \Delta \) - round trip propagation delay for the session
- \( \beta \) - target buffer backlog for the session
If the fair rate for the session is time varying \( R(t) \), \( \Delta(t) \) an estimate at the transmitter of \( \Delta \) at time \( t \) and \( R(t-\Delta) \) is the available rate as known to the transmitter at time \( t \); then the native rate adapted window is,

\[
W(t) = R(t-\Delta) \cdot \Delta(t) + \beta
\]  

(2.2)

### 2.3 Intelligent Agent

An agent is used more and more in Information Technology (IT) as well as in communications. An agent is defined as a piece of software that can achieve a specific task in an autonomous way (Jens et al. 1998). All the agents have the possibility of coordinating, communicating and cooperating with the system or with other agents.

An intelligent agent is an information processing system, located in some environment, which is able to carry out flexible and autonomous actions in order to meet its aims of design (Jennings et al. 1998). An intelligent agent is a software entity, which carries out operations for a user or for another program, with some degree of freedom and autonomy and which exploits the knowledge or representations of the desires and objectives of the user (IBM 1995).

An intelligent agent is an autonomous entity which observes and acts upon an environment, and directs its activity towards achieving goals. Intelligent agents may also learn or use knowledge to achieve their goals. This may be very simple or very complex: a reflex machine, such as a thermostat, is an intelligent agent, as is a human being, or a community of human beings working together towards a goal. Intelligent software agents have a collection of properties that make them very adequate to provide services to users.
2.3.1 Features of Intelligent Agents

The following are some of the general features of intelligent agents used for various applications.

i. **Autonomy** - The agents can perform tasks without direct and continuous guidance from the user.

ii. **Learning** - The agents can be applied with machine learning techniques to construct automatically a user profile and adapt the actions to the user’s preferences.

iii. **Proactiveness** - A personal agent can anticipate the needs of the user and perform tasks that may be beneficial for the user, without an explicit request from the user.

iv. **Social ability** - A personal agent can get in touch with other agents that offer information about any domain in which the user may be interested, and provide information that the user needs in a friendly and personalized way.

2.3.2 Types of Agent Models

Agent-based computing has led to many attempts at defining the term agent. A system situated within an environment, and a part of it that senses that environment and acts on it, over time, in pursuit of its own agenda, so as to effect what it senses in the future, can be broadly called as an agent. An agent is capable of flexible autonomous action, which implements the autonomous, communicating functionality of an application.

The concepts of a distinct, discrete entity (e.g., a program), a computational environment (i.e., a computer platform), an agent interaction environment, and agent independence/autonomy, the specification of tasks and/or goals and communication are included in the agent. However, an exact
definition of an “agent” is sometimes contentious, and the most stringent definitions rule out much of what are generally considered as agent-based applications. An agent has deliberative capacities that rule out the entire class of purely reactive agents from attaining agent hood.

An agent in a node can be either an active or a reactive component during network implementation. Links and nodes are reactive components that react to incoming packets and apply their behaviors to the packet. Compared to them, agents can be active components. For instance, when transport agents TCP and User Datagram Protocol (UDP) react to external orders for sending data, they may generate the necessary packets for data transmission or generate connection control packets. In the case of transport agents, they require external components to generate the data to be sent (Roshan et al 2004).

Agents and Multi-Agent Systems (MASs) are more successfully deployed, even if many theoretical issues underlying them remain unexplored. The lack of rigorous formal definitions and characterizations of agent hood, and the ambiguity or even disagreement about what constitutes MAS, make the gap between theory and practice quite evident. The system classification is meant to be theoretically comprehensive, while providing an agent designer with a practical means to choose a system that is appropriate for the task at hand.

The analysis of agent systems focuses on aspects such as distribution, concurrency and security. In shifting from a comprehensive viewpoint to a narrower focus, the idea is to highlight a niche in the “agent system space” that is not currently filled by existing systems. The system classification used to restrict the possible criteria while establishing a list of features is fairly simple – differentiating between systems that facilitate the
design and implementation of individual agents, and those that take a more system-oriented view (i.e., concentrate on systems composed of many agents).

The distinction between the Single Agent System (SAS) and MAS system implementation often displays some functional overlap; an agent system may be geared towards providing a framework in which agents interact, while also providing agent architecture or templates.

2.4 Intelligent System Models

Intelligent models can be classified into the following two types.

i. Single-Agent Models
ii. Multi-Agent Models

2.4.1 Single-Agent Models

SAS is mostly concerned with the design of individual agents. There are two aspects of supporting the individual agent design: concerns regarding the internal structure of an agent (i.e., the agent’s architecture) and concerns regarding the mechanisms the agent uses to interact with its external environment. An agent’s “architecture” refers to the structure of its internal processes, which dictate how data from the sensors is processed, and what resultant actions are produced by the effectors. The term control system will be used instead; the two terms are meant to be synonymous.

A typical single-agent learning model contains the following four elements.
i. Environment
ii. Learning agent
iii. A set of actions
iv. Environmental response

The learning agent selects an action for the system, which leads the system’s evolution along a unique path, till it encounters another decision-making state. After a state transition, the learning agent gathers sensor inputs from the environment, and derives information about the new state, immediate reward and the time spent during the most recent state-transition. Using this information and the algorithm, the agent updates its knowledge base and selects the next action, as shown in figure 2.1.

![Reinforcement Learning Scheme](image)

Figure 2.1 Reinforcement Learning Scheme

This completes the iteration process, as this process repeats; the learning agent continues to improve its performance. A simulation model of the system provides the environment components of the model. The varieties of problems considered to make a dynamic update are given below.
i. Maintaining application consistency - The states of the components must not be affected by changes in the application architecture.

ii. Preserving the bindings of the components - Bindings have to be preserved by redirecting the calls to new components and managing transient states.

iii. Initializing new components - New components must be initialized with adequate internal state according to the former component.

iv. Preserving communication channels by avoiding message loss, duplication or excessive delays.

**2.4.2 Multi-Agent Model**

A multi-agent model is a dynamic federation of agents connected by the shared environments, goals or plans, and which cooperate and coordinate their actions (Huhns 1999). The capacity to communicate, to coordinate and cooperate makes the use of multi-agents in communication environments interesting.

An agent is “a computer system and is situated in some environment that is capable of autonomous action in this environment in order to meet its design objectives”. An agent system is shown in figure 2.2. An agent will not have complete control over its environment, in most domains of reasonable complexity. It has partial control, in that it can influence its environment. This means that from an agent’s point of view the same actions performed twice in apparently identical circumstances, might appear to have entirely different effects.
Intelligent agents are not only able to perceive their environment, but are also able to respond in a timely fashion to changes, and are able to exhibit goal-directed behavior by taking the initiative in order to satisfy their design objectives. Intelligent agents are capable of interacting with other agents (possibly human) in order to satisfy their design objectives.

MASs are typically distributed systems, in which several distinct components, each of which is an independent problem solving agent, come together to form a coherent whole. A MAS is a system that contains,

i. Two or more agents.
ii. At least one autonomous agent.
iii. At least one relationship between two agents where one satisfies the goal of the other.
The typical structure of MAS consists of a number of agents that interact with one another through communication as illustrated in figure 2.3. The agents have the ability to act in an environment and have different “spheres of influence”. This means the control over or ability to influence different parts of the environment. In some cases these “spheres of influence” may overlap or coincide. This gives rise to dependency relationships between the agents. Agents will also be linked with one another by other relationships.
2.5 Heterogeneous Agents

MASs are primarily concerned with the interaction of multiple, often heterogeneous agents in a single framework. A heterogeneous agent-based system can be divided into the following two sub-categories.

i. Agent-Based Modeling (ABM) systems
ii. Communication infrastructures

2.5.1 Agent-Based Modeling

ABM systems provide a set of tools geared towards agent simulation. Individual agent design and implementation is done entirely within the constraints and conventions imposed by the constructs of the system. A major concern of the ABM systems, is providing a comprehensive set of statistical analysis tools.

2.5.2 Communication Infrastructure

A communication infrastructure agent designer is generally expected to create a model of the agents, their capabilities, the environment in which they reside, and the methodology for evaluating the system from scratch.

The desirable features of multi-agent systems are presented in a variety of example applications. They include,

i. Distributed system
ii. Network routing protocol based on ants
iii. E-commerce auctions
iv. Team of box-pushing robots or a robotic soccer team
The last agent application gives a single virtual agent, such as a web-spider or other information gathering agent. Another application would likely include heterogeneous agents, while an auction necessitates a large degree of agent autonomy.

2.6 Types of Heterogeneity

A distributed system requires a robust communication infrastructure, and is based on the client/server model. Individual agents can be relatively simple and homogeneous; their purpose lies in making use of many host computers rather than performing complex sub-tasks themselves. To develop such a massively distributed system, the agent designer should be supplied with a simple method of connecting agents over a network. In addition, failure recovery is of great concern. When an agent fails, the integrity of the data set has to be preserved. The server must have a way of reallocating a sub-task to another agent.

A network routing protocol that has an ant colony metaphor displays distribution. However, it is likely that agents will be heterogeneous to some extent, with different roles to play in the system, whereas a distributed system, an ant-like network protocol, has the possibility of supporting many agents on a single host, which implies a large degree of concurrency. In addition, a system monitor that supplies complete information about the system is desirable. The following are the two types of heterogeneity.

i. Autonomous Heterogeneity
ii. MAS Heterogeneity
2.6.1 *Autonomous Heterogeneity*

Heterogeneity can be expected from agents in the application. An agent designer can expect agents to have to interact with what are commonly referred to as middle or broker agents. Middle agents are responsible for what is often called yellow pages service, which means that the agents take requests from one agent, locate another agent that can fulfill the request, and put the two in contact. A complete treatment of security includes a way of making sure that an agent is actually the agent it is identified as, providing some data encryption policy, so that unwanted agents cannot eavesdrop or intercept data being passed from one agent to another, and making sure that only agents that have permission are allowed access to the system.

As information technology becomes accessible to more people, and as commercial and government organizations are challenged to scale their services to larger market shares and wider user communities, while minimizing or reducing their costs in doing so, there is an increased demand for software applications to provide the following three features to their human end-users.

i. Rich application end-to-end functionalities.

ii. Reduction of human involvement in the process by reducing information overload, reducing configuration management and system maintenance overheads, and enabling the rapid specification of new, often context-aware tasks.

iii. The use of existing software applications and systems is done in a novel or adaptive way.

Distributed MAS (DMAS) technologies and web services show their inherent modularity and ease with which they can be recombined to form
new applications. When designing new distributed software systems, however, the broad requirements and their translations into specific implementations are typically addressed by partial, complementary and overlapping technologies.

The goal of a multi-agent system is to have a positive impact in any combination of the following three areas.

i. To interact a human end-user with information-based perceptual capabilities, by reducing information overload and providing context-relevant information.

ii. To qualitatively and quantitatively improve the range of actions and activities in which the end user can engage.

iii. To enhance the means, typically through the context aware use of devices, as is done in pervasive and ubiquitous computing, by which humans may perceive the world or by which humans may affect their decisions within it.

The MAS is based on the assumption that operates in an open world. The networked environment in which an agent operates is open, or without bounds; it is dynamic in nature from the perspective of network topologies, agent capabilities and agent locations; and the networked environment is uncertain, that is, the same agent that provided an answer to an earlier request may not be available when called upon again.

In the MAS there is also an assumption, that often there will be some degree of service or functional replication so that should one agent fail, one or many other agents and service providers can be found to substitute for the failed agent. The MAS infrastructure has to be domain independent, and a reusable substratum on which MAS systems, services, and components live,
communicate, interact and interoperate; the infrastructure should support agents and facilitate their social interactions with each other, rather than impose itself.

2.6.2 **MAS Heterogeneity**

The heterogeneities that have been identified and striven to accommodate within the RETSINA system is a MAS heterogeneity. This includes the following.

i. **Communication Heterogeneity**: This includes different communication interfaces that the agent should use for effecting its communication with its peers (e.g. radio, wire, etc.), and the available underlying network protocols used.

ii. **Coordination Heterogeneity**: There are multiple coordination techniques such as capability-based coordination, team-oriented coordination, the Contract Net Protocol (CNP), auction-based coordination schemes, and others, which depend primarily on the task that needs to be performed, and the coordinating attitude of an agent.

iii. **Environmental Heterogeneity**: The operating environment can range from the network operating environment, in which network protocols are performing (e.g. throughput, transport reliability, network connection permanence, etc.), to the computational environment in which software capabilities change, to physical and terrain environments of agent-augmented hardware and robots.

iv. **Security Heterogeneity**: In the RETSINA, security is viewed as being parameterized by the application, in addition to the encryption of communication and authentication of the component
identities. The security models are evaluations of trust per individual agent vs. trust of agents running on a trusted platform, and digital rights management for MAS-aided information fusion, aggregation and sharing.

v. **Semantic Heterogeneity:** The main focus of any multi-agent system, this heterogeneity expresses the issue that any two interoperating agents must be certain when using a vocabulary of terms, or translations thereof, that they are using the same concepts with the same relevant inferences of relations as the other communicating agents.

vi. **Systems Heterogeneity:** This heterogeneity is derived from the differences of devices and hardware, operating systems, implementation language and execution environment. Despite these challenges, non-trivial Learning in Multi-Agent Systems (LMASs) can be implemented and executed reliably, by recognizing some important characterizations of Agent Business Systems Engineering (ABSE).

### 2.7 Multi-Agent System

The key point in the widespread acceptance of MAS by the research community is their deployment in wireless devices. If personal agents have to be able to communicate with other service providing agents, they must follow a set of standard norms concerning the agent communication language to be used, and the communication protocols to be followed. Multi agent is a software development framework aimed at developing agent systems and applications conforming to the Foundation for Physical Intelligent Agents (FPIA), standards for intelligent agents.
2.7.1 **MAS Features**

A preliminary list of MAS features to be used by an agent designer shall include the following.

i. A communication infrastructure
ii. Mechanisms for distributed computing
iii. A service that enables location independence
iv. The means of performing failure recovery
v. Support for heterogeneous agents
vi. Concurrent agent operation
vii. A system monitor
viii. A yellow pages service
ix. Explicit security mechanisms

2.7.2 **MAS Link Interfaces**

A wireless protocol has the ability to adapt to situations that differ from the ones for which it was originally designed. The wireless TCP is prepared to work in MAS with reasonably low delays and low link error rates. In such cases, data is seldom lost or corrupted, due to link errors and the main cause of packet loss is data being discarded in congested routers. For that reason, the wireless TCP always considers a loss indication as a sign of congestion, and takes action accordingly. However, there are a number of situations where this assumption is no longer valid.

The wireless TCP is unable to distinguish a loss due to congestion, where decreasing the sending rate is necessary to alleviate the congested link, from a random loss, where reducing the rate is not only useless, but counterproductive as well. Such a situation causes the wireless TCP sender to
cut repeatedly the sending rate by half, leading to a serious degradation of performance. The lack of a mechanism to inform the wireless TCP sender that the destination is reachable again, introduces extra delays that increase the response time of the connection (Rappaport 2002).

2.7.3 **MAS Architecture**

The MAS mechanism is an agent-based solution that is intended to overcome most of the problems that wireless TCP faces over networks. Most of its functionality resides in an agent that is situated in the network backbone, and acts as an intermediary between the mobile terminals and the content providers. The agent aims to reduce latency and slow start time, improve the wireless link utilization, and reduce the buffer needs in the intermediate elements of the network. It is, in principle, designed for asymmetric downlink traffic, such as web browsing and File Transfer Protocol (FTP) file transfer, and could easily be extended to deal with person-to-person symmetric communication.

The agent implements a modified version of the wireless TCP, in order to adapt its behavior to the special characteristics of the MAS environment, something that current wireless TCP versions have failed in achieving. It splits the TCP connection into two, in order to hide the wireless link from the wired servers, and adapts the connection parameters in a way that is completely transparent to the endpoints. However, the introduction of an agent allows solving the wireless problem locally, and without the need to involve the endpoints. A careful design and implementation is carried out so as to make it scalable, as an acceptable tradeoff maintaining end-to-end semantics, is difficult.
Figure 2.4 shows only the main elements that are relevant from the Radio Network Controller (RNC) point of view. Some agent network elements as well as their interconnections are omitted, on the assumption that the core network is properly dimensioned, and that the main bottleneck is the wireless link. Thus, only the elements that play an important role in the wireless TCP connection and adaptation are represented; from figure 2.4 it is inferred that,

i. All the connections are initiated by the mobile terminals, which attempt to send requests to the remote servers.

ii. The adaptation mechanism has important information about the available bandwidth assigned to every user. This information is sent to the agent, both periodically and upon every bandwidth change, in order to update the connection TCP parameters. It also forwards all the packets sent from and towards the terminals.

iii. The agent, which is located between the core network elements and the external network, forwards the requests from the terminals and adapts the TCP parameters of the connections in order to optimize the data transfer.
iv. Remote servers receive requests from the agent and send the requested files back.

2.7.4 MAS Challenges

The MAS operates in an open, unbounded world. Therefore, MASs should have their importance and their requirements must be context-aware. The requirements must also vary, based on the environmental conditions. The environmental condition interacts with the agents that can engage. The environment is also dynamic and changing, which challenges any software engineering paradigms that require the explicit enumeration of objects and relationships among objects in the environment.

In a distributed computing system, where there is no centralized point of control, the failure of any one computation system or communication link or network node can affect the whole distributed execution state of the MAS. The distributed execution state of the MAS application is inconsistent, and the resulting inconsistency can be difficult to identify. Because of this it is also difficult to remedy. In an agent-based wireless network, the heterogeneities that have been identified and tried to be accommodated within the multi agent RETSINA system are as follows.

i. Communication Heterogeneity
ii. Coordination Heterogeneity
iii. Environmental Heterogeneity
iv. Security Heterogeneity
v. Semantic Heterogeneity
vi. Systems Heterogeneity
2.8 Agent Applications in Wireless Networks

Recently, research is going on in the field of multi-agent systems in wireless networks. The study of mobile agent communications in wireless networks can present some solutions like, the use of intelligent agents for the management of heterogeneous networks, the proposal of a multi-agent architecture to control the M-commerce (Gregory et al 2001), the definition of a reference scenario using the fixed and mobile agents to guarantee QoS between a server and a mobile terminal (Anastasi et al 2000), and the implementation of an-agent based home banking service for a mobile network (Hartmann et al 1999). These studies identified the following five fields for the application of agent technology in the wireless network.

i. Mobile Terminal Localization - Wireless networks will provide to multimedia applications a large bandwidth with the necessary quality of service. So, a method will consist in using the localization of the terminal and making forecasts to allocate the resources necessary to the terminals, and also for the re-routing of the traffic. The evaluation of the localization of the mobile terminal becomes an integral part of the wireless network management system. A simple method to locate the mobile terminal is to employ measurements to determine the radio path loss. The localization of the mobile terminal is based on the measurement of the propagation of the path loss between the mobile terminal and the base stations, as also on other measurements (McGuire et al 2003).

ii. Improvement of a Mobility Protocol - Agent technology can be used to improve the effectiveness of a mobility protocol, such as Mobile-IP. This protocol was adopted by Internet Engineering Task Force (IETF) to ensure mobility in an Internet Protocol (IP) network. This protocol does not maintain the performances of a
connection in the case of a user’s mobility. The time needed to update the databases indicating the new position of the user can degrade the connection quality. Before the arrival of the localization request, several information packets can be lost after their dispatch to the previous destination of the mobile agent.

iii. **Adaptation of the Handover to the Users Need** - Agents can be used in the wireless network to support the vertical handover. An agent, having carried out and deployed the software of an air interface for a radio technology will autonomously request the software for another technology before the vertical handover.

iv. **Signalization Control** - Mobile agents are useful to control signalization. The agents can also be used to dynamically negotiate the needs of quality of service, security, and mobility for the user in the wireless network.

v. **Reduction of the Wireless Access** - The mobile agents can distribute the code on the wireless network equipment. The number of exchanges necessary to provide a personalized service can be greatly reduced. This will improve the performance of the wireless network by decreasing the consumption of bandwidth and reducing latency.

### 2.8.1 **MAS Application Problems in Wireless Networks**

The problems concerning the application of the agents in a wireless environment are related to the security, congestion, cost, compatibility, and implementation. The access to internal network resources must be well secured. Security has to play a major role in the design of the software environment supporting the agent deployment. Congestion will reduce the packet delivery ratio and increase the latency in wireless networks. The cost, in terms of agent migration and local processing also needs to be taken into
consideration. The compatibility of the agent platforms in terms of the code and interfaces needs to be assured. Implementing mobile agent technology in a wireless environment, is also a heavy task.

### 2.8.2 Agent Applications in Wireless Sensor Networks

Agent-based wireless sensor networks can provide better reliability, robustness, and reduce the costs of data retransmission and communication bottlenecks due to the slowness of the communication networks. By employing the agent-based technologies in wireless sensor networks, the systems can make accurate decisions quickly, and reduce the data rate and data redundancy problems (Li et al 2009).

### 2.8.3 Mobile Agents in Wireless Sensor Networks

Mobile agents are software codes which can hop from one computing node to the next node in a wireless sensor network. The advantage of mobile agent technology in a wireless sensor network is that it reduces the transmission of raw data from the source to the destination in the network. The transmission power requirements of the mobile agents are generally much greater than the wireless node controller’s processing power requirements. Therefore, it is beneficial to efficiently transmit a mobile agent program to process raw data in the local network node, rather than transmitting raw data from different wireless sensor nodes to a sink node for processing (Li et al 2009).

Agilla is a mobile agent middleware package designed for TinyOS WSN systems (Fok et al 2005). This agent is aware that WSN systems can generate lots of raw observation data. The mobile agent is a perfect technology to address the bandwidth and power consumption problems in
WSN systems. In order to further reduce energy consumptions, many WSN systems have power management schemes to operate a wireless node in an active mode and a sleep mode periodically.

2.9 RETSINA Agent

REUsable Task-based System of Intelligent Network Agent (RETSINA) is a cooperative multi-agent system that consists of the following three classes of agents.

i. Interface Agents
ii. Task Agents
iii. Information Agents

In RETSINA, the agents are distributed and run in different machines. They may dynamically enter and leave the system. In order to solve the problem of communication among agents, the RETSINA system introduces a matchmaking mechanism. During the operation of the system, agents who enter the system register their information to an agent and when they leave, they tell the agent to delete their registered information.

RETSINA is an open MAS, that supports heterogeneous agents and has been implemented on the premise, that agents in a system should form a community of peers that engage in peer to peer interactions. Any coordination structure in the community of agents should emerge from the relations between agents, rather than as a result of the imposed constraints of the infrastructure itself. In accordance with this premise, RETSINA does not employ centralized control within the MAS; rather, it implements distributed infrastructural services that facilitate the interactions between agents.
RETSINA provides a domain-independent, componentized and reusable substratum to,

i. Allow heterogeneous agents to coordinate in a variety of ways, and
ii. Enable a single agent to be part of a multi-agent infrastructure.

RETSINA provides facilities for reuse and a combination of different existing low-level infrastructure components. It also defines and implements higher level agent services and components that are reconfigurable and reusable. The modifications to the wireless TCP can be classified in two ways.

i. Take into consideration how and where the adaptation is performed: end-to-end, link layer and split connection.
ii. Classify according to which wireless TCP problem they address end-to-end solutions and assume that it is not possible to rely on the network in order to improve the performance of the transport protocol.

In such cases, the endpoints are responsible for performing the necessary changes to ensure a good adaptation, and they must be aware of the problems of the wireless transport protocol. The main advantage is that these solutions can be used in any situation, as they do not depend on the underlying layers. However, the code of either the sender or the receiver or even both must be modified, which might be a shortcoming in many cases. Conventional link layer solutions make the wireless TCP appear as a higher quality link, but, with reduced effective bandwidth.

Some of these solutions are not able to overcome the problems of disconnections caused by the wireless TCP. These solutions are said to be
more efficient than the previous ones and the endpoints do not need to be aware of the adaptation. However, there is a need to translate from one protocol to the other at the base station, with the resulting overhead. It implements distributed infrastructural services that facilitate the interactions between agents as opposed to managing them. RETSINA provides a domain-independent, componentized and reusable substratum to,

i. Allow heterogeneous agents to coordinate in a variety of ways, and
ii. Enable a single agent to be part of a multi-agent infrastructure.

RETSINA provides facilities for reuse and a combination of different existing low-level infrastructure components, and also defines and implements higher level agent services and components that are reconfigurable and reusable. They provide the means by which the agent communicates with the networked world. The communicator provides a level of abstraction that insulates the components from the following.

i. Issues of Agent Communication Language (ACL)
ii. Communication Session Management (CSM)
iii. Location of Agent Services (LAS)
iv. Logging and visualization of agent messages and state information
v. Communication transport being used (e.g. infrared, telephone, base band, etc.)

The three RETSINA architectures provide abstract architectural descriptions that, together address many of the heterogeneities. An individual agent is the functional architecture of a society of RETSINA agents, and the infrastructure architecture provides most systems the integration of agent components and technologies. The schematic diagram of the RETSINA agent
architecture is shown in figure 2.5. In the architecture, the black line represents the control flow and the red line depicts the data flow.

![Diagram of RETSINA Agent Architecture](image)

Figure 2.5 RETSINA Agent Architecture

### 2.9.1 RETSINA Individual Agent Architecture

The RETSINA individual agent architecture implements Hierarchical Task Network (HTN) planning, scheduling and execution monitoring in three parallel execution threads, and provides the means by which the agent communicates with the networked world. This enables actions in a priority queue, and work with the Execution Monitor, which actually executes the enabled actions, monitors the execution, and handles failures.
The coordination among the three planning modules is done in such a way that high-priority actions can interrupt those being executed by the Execution monitor, if those being executed are of a lower priority.

2.9.2 RETSINA Functional Architecture

The RETSINA multi-agent system is a collection of heterogeneous software entities that collaborate with each other to provide a result or a service to other software entities or to an end user, as shown in figure 2.6.

![Figure 2.6 RETSINA Functional Architecture](image)

The RETSINA functional architecture categorizes agents as belonging to any of the following four agent types.
i. **Interface agents** present agent results to the user, or solicit input from the user. In addition, they could learn from user actions. Interface agents typically represent specific modes of input or output, such as a voice recognition agent or a speech generation agent, or can interact with device agents to determine the proper way to display information given a device display characteristics, or to retrieve input from a user. Interface agent behaviors can also be associated with task agents.

ii. **Task agents** encapsulates task-specific knowledge and uses that knowledge for requesting or performing services for other agents or humans. In this respect, they are the typical agent coordinators of a multi-agent system.

iii. **Middle agents** provide infrastructure for other agents. A typical instance of a middle agent is the matchmaker or yellow pages agent. Requesting agents submit a capability request to the Matchmaker, which will then locate the appropriate service-providing agents, based upon their published capability descriptions, known as advertisements.

iv. **Information agents** model the information world to the agent society, and can monitor any data- or event-producing source for user-supplied conditions. Information agents may be a single source if they only model one information source, or may be multi-source if one information agent represents multiple information sources. Information agents can also update external data stores, such as databases, if appropriate.

RETSINA agents also support other forms of coordination techniques, such as team-oriented coordination, auction-based coordination, and CNP.
### 2.9.3 RETSINA Infrastructure Architecture

<table>
<thead>
<tr>
<th>RETSINA MAS INFRASTRUCTURE</th>
<th>INDIVIDUAL AGENT INFRASTRUCTURE IN RETSINA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAPABILITY TO AGENT MAPPING</strong></td>
<td><strong>CAPABILITY TO AGENT MAPPING</strong></td>
</tr>
<tr>
<td>Matchmaker Module</td>
<td>Matchmaker Module</td>
</tr>
<tr>
<td><strong>NAME TO LOCATION MAPPING</strong></td>
<td><strong>NAME TO LOCATION MAPPING</strong></td>
</tr>
<tr>
<td>ANS Module</td>
<td>ANS Module</td>
</tr>
<tr>
<td><strong>SECURITY</strong></td>
<td><strong>SECURITY</strong></td>
</tr>
<tr>
<td>Certificate Authority</td>
<td>Security Module</td>
</tr>
<tr>
<td>Cryptography Services</td>
<td>Private &amp; Public Keys</td>
</tr>
<tr>
<td><strong>PERFORMANCE SERVICES</strong></td>
<td><strong>PERFORMANCE SERVICES</strong></td>
</tr>
<tr>
<td>Failure Monitoring</td>
<td>Self Monitoring Cloning</td>
</tr>
<tr>
<td><strong>MAS MANAGEMENT SERVICES</strong></td>
<td><strong>MANAGEMENT SERVICES</strong></td>
</tr>
<tr>
<td>Logger Activity Visualizer Launcher</td>
<td>Logger Module</td>
</tr>
<tr>
<td><strong>ACL INFRASTRUCTURE</strong></td>
<td><strong>ACL INFRASTRUCTURE</strong></td>
</tr>
<tr>
<td>Public Ontology Protocols Servers</td>
<td>ACL Parser Private Ontology Protocol Engine</td>
</tr>
<tr>
<td><strong>COMMUNICATION INFRASTRUCTURE</strong></td>
<td><strong>COMMUNICATION MODULES</strong></td>
</tr>
<tr>
<td>Discovery Message Transfer</td>
<td>Discovery Module RETSINA Communicator</td>
</tr>
</tbody>
</table>

**OPERATING ENVIRONMENT**

- Machines, OS Network Multicast Transport Layer TCP/IP Wireless Infrared SSL

---

Figure 2.7 RETSINA MAS and Individual Agent Infrastructures
Agents in MAS are expected to coordinate by exchanging services and information, to be able to follow complex negotiation protocols, to construct models of each other and shared models of their tasks and, to agree on commitments, and to perform other socially complex operations. This is shown in figure 2.7.

In addition, agents need conventions, such as ACL, conversational policies and ontologies that define the meaning of the terms that can be used for providing the basis for achieving semantic interoperability and agreement with each other. The architecture of RETSINA represents the abstract dependencies of the upper layers of infrastructure modules on the lower layers. The figure 2.7 is organized to show the internal modules of an individual agent, so that it may interact with the infrastructure components. In the left column, the organization of the MAS, and in the right column, how the overall architecture of the MAS is described.