Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2014, Article ID 480372, 17 pages
http://dx.doi.org/10.1155/2014/480372

Research Article

A QoS-Based Wireless Multimedia Sensor Cluster Protocol

Juan R. Diaz,[1] Jaime Lloret,[1] Jose M. Jimenez,[1] and Joel J. P. C. Rodrigues[2]

[1] *Universidad Politécnica de Valencia, Camino Vera s/n, 46022 Valencia, Spain*
[2] *Instituto de Telecomunicações, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal*

Correspondence should be addressed to Jaime Lloret; jlloret@dcom.upv.es

Received 15 January 2014; Accepted 15 March 2014; Published 18 May 2014

Academic Editor: Sana Ullah

Wireless Sensor Networks (WSNs) provide a wireless network infrastructure for sensed data transport in environments where wired or satellite technologies cannot be used. Because the embedded hardware of the sensor nodes has been improved very much in the last years and the number of real deployments is increasing considerably, they have become a reliable option for the transmission of any type of sensed data, from few sensed measures to multimedia data. This paper proposes a new protocol that uses an ad hoc cluster based architecture which is able to adapt the logical sensor network topology to the delivered multimedia stream features, guaranteeing the quality of the communications. The proposed protocol uses the quality of service (QoS) parameters, such as bandwidth, delay, jitter, and packet loss, of each type of multimedia stream as a basis for the sensor clusters creation and organization inside the WSN, providing end-to-end QoS for each multimedia stream. We present real experiments that show the performance of the protocol for several video and audio cases when it is running.

1. Introduction

The number of Wireless Sensor Network (WSN) real deployments is increasing considerably in the last years [1, 2], mainly because of their huge benefits [3]. New wireless technology standards, recent advances in energy-efficient hardware and video coding algorithms are allowing multimedia delivery over ad hoc networks.

Nowadays, the features of the sensor nodes and smart devices are very similar to the regular personal computer features. Last generation of sensor nodes can include advanced models of CPUs, with several cores, 1 or 2 GB of RAM, and storage capacities up to 64 GB. Moreover, they can include multiple wireless interfaces such as Bluetooth, Wi-Fi, 3G, and 4G,

The amounts of multimedia services that can be offered through the network are very large [4, 5]: VoIP, IPTV, radio, teaching, multimedia streaming, games, and so forth. There are a lot of multimedia platforms and protocols used in different fields [6], from the entertainment to the training in the business environment.

Ad hoc network is a self-organizing multihop system of wireless nodes which can communicate with each other without a preexisting infrastructure [7]. Multimedia ad hoc networks can be ideal to allow a distributed multimedia service in commercial and social environments that require high visibility to the offered products.

With the widespread use of wireless technology, the ability of mobile wireless ad hoc networks to support multimedia services with quality of service (QoS) has become a challenging research subject as described by Khoukhi and Cherkaoui in [8]. According to Barenco Abbas et al. the main goal of QoS is to achieve a more deterministic network behavior [9]. Chen et al. suggested in [10] that when we need to provide an acceptable QoS in the network, we should define the values of QoS metrics in order to establish the necessary requirements. These requirements are different if it is a real-time service or an on-demand service.

Due to the severe limitations of the ad hoc networks (in terms of energy, processing power, memory, bandwidth, etc.), it is necessary to carefully design the multimedia ad hoc network protocol. Some works are focused on proposing multichannel cross-layer architectures [11], while others are focused on providing fast rerouting algorithms [12], but in this case we focus our research on providing the best topological structure based on the type of multimedia streams.

There have been many studies proposing different topological structures for ad hoc networks that can be summarized into two main types: planar and hierarchical topologies [13]. Planar topologies in ad hoc networks may be of great complexity, mainly in mobile ad hoc networks, because any node displacement may change the entire network topology. For this reason most of researchers have proposed the use of a hierarchical structure for performing an ad hoc network topology [14]. In many cases this hierarchical structure split nodes into different groups called clusters [15, 16].

In this paper we show the design and performance test of a new multimedia protocol which takes into account the QoS in WSNs. The protocol uses the QoS parameters to structure the network topology. Then, a node decides where to join based on its QoS needs. The protocol is based on the architecture proposed by us for wireless ad hoc networks in [17]. While in [17] we only propose architecture for ad hoc networks, in this paper we have particularized the architecture to Wireless Sensor Networks and we have focused this work to the design and deployment of the network protocol.

This paper is organized as follows. Section 2 presents the research papers related with our work. Proposed protocol and architecture are described in detail in Section 3. The system operation is explained in Section 4. Section 5 shows the obtained results and our discussion of the performance study. Finally, in Section 6, conclusion and future work are shown.

2. Related Work

We have structured the related work section in 2 parts. The first part shows several cluster formation algorithms, while the second part discusses published cluster-based multimedia ad hoc networks.

There are several surveys that review existing works on cluster formation algorithms. On one hand, according to Wei and Anthony Chan [18] cluster topologies can be classified into four categories: single-hop or multihop, stationary or mobile, synchronous or asynchronous, and location-based or non-location-based. On the other hand, Yu and Chong [19] made a categorization of clustering schemes in stationary and mobile ad hoc networks and sensor. They classified 14 proposed clustering schemes into six categories based on their main objectives. Moreover, they discussed each clustering scheme in terms of objective, mechanism, performance, and application scenario and discussed the similarities and differences between schemes of the same clustering category. We have also found [20], authored by Agarwal and Motwani. They reviewed several clustering algorithms which help organize mobile ad hoc networks in a hierarchical manner and presented their main characteristics. With this survey we see that a cluster-based MANET has many important issues to examine, such as the cluster structure stability, the control overhead of cluster construction and maintenance, the energy consumption of mobile nodes with different cluster-related status, the traffic load distribution in clusters, and the fairness of serving as cluster heads for a mobile node.

We have also found two papers written by Abbasi and Younis [21] and Boyinbode et al. [22], which present a synthesis of existing clustering algorithms in WSNs and highlight the challenges in clustering. They survey different clustering algorithms for WSNs, emphasizing their objectives, features, complexity, and so forth. They also compare their metrics such as convergence rate, cluster stability, cluster overlapping, location awareness, and support for node mobility.

Despite this review, we would like to mention 4 clustering algorithms not included in these surveys because of their importance.

Ramachandran et al. [23] proposed two new distributed clustering algorithms for wireless ad hoc networks. They presented a 2-stage O(N) randomized algorithm for a N node complete network, which finds the minimum number of star-shaped clusters, all at their maximum size. They also proved the correctness of this algorithm. They then presented a completely deterministic O(N) algorithm in which cluster heads are elected autonomously by the nodes. They compared their performance using simulations on top of Bluetooth's device discovery procedures. Results show that the randomized algorithm performs better with respect to both cluster and network formation times.

Chatterjee et al. [24] proposed a weight based distributed clustering algorithm (WCA) which can dynamically adapt itself with the ever changing topology of ad hoc networks. Their approach restricts the number of nodes to be catered by a cluster head so that it does not degrade the MAC functioning.

Lehsaini et al. [15] showed the development of an architecture that creates clusters and establishes connections between sensors of the same type by building different sensor networks. In their proposal the cluster heads manage the network since they have connections with other cluster heads and these connections allow connecting cluster members from different clusters when they are of the same type, forming a specialized network. One of the main goals is that if all cluster heads switch off at the same time, the system is able to continue working, although there will not be new connections between clusters through the cluster heads.

Kavitha and Karthikeyan [25] proposed an energy enhanced version of the M-SPIN (EEM-SPIN) protocol using WCA for WSNs. It has the flexibility of assigning different weights and takes into account combined metrics to form clusters automatically. Limiting the number of nodes inside a cluster allows restricting the number of nodes catered by a cluster head so it does not degrade the MAC functioning. For a fixed cluster head election scheme, a cluster head with constrained energy may drain its battery quickly due to heavy utilization. In order to spread the energy usage over the network and achieve a better load balancing among cluster heads, reelection of the cluster heads may be a useful strategy.

Next, we review how some of the main cluster-based multimedia ad hoc networks are created.

Huang et al. [26] have presented a cluster-based model to support multimedia service. The proposed model transmits multimedia streaming stably in ad hoc networks, while mobile users who consume multimedia streams tend towards

group-based behavior. An on-demand connection prediction to measure the likelihood of connectivity of cluster-based routes in a future time is applied to the cluster-based transmission of multimedia streaming. They proposed a routing method called PLCBRP (cluster-based routing with the prediction of connection probability), which combines the cluster-based routing protocol with the prediction scheme. PLCBRP discovers an optimal loosely cluster-based route for transmitting long multimedia streams. Simulation results indicate that PLCBRP delivers more data packets and provides more quality on the transmission of multimedia streaming than other flat on-demand routing protocols do.

Tang and Li [27] developed a QoS supporting scheme for dynamic traffic conditions by controlling data generating rates at individual clusters. Besides, they have investigated an explicit solution on the energy distribution at different clusters in the WSN, based on an optimal energy allocation criterion. The obtained network energy distribution formula is particularly convenient for node deployment design in WSNs. The proposed algorithm is presented and validated by numerical simulations. Some situations are also discussed and presented by experimental examples.

Rosário et al. proposed MEVI in [28] a smart multihop hierarchical routing protocol for efficient video communication over Wireless Multimedia Sensor Networks. It combines a cluster formation scheme with low signaling overhead in order to ensure reliable multihop communication between cluster heads and base stations. For route selection, a cross-layer solution selects routes based on network conditions and energy issues and a smart scheme to trigger multimedia transmission according to sensed physical environmental conditions. The cluster approach aims to minimize the energy consumption. MEVI allows the transmission of multimedia content with QoS/QoE support by introducing a hierarchical routing protocol. Simulation experiments show the benefits of MEVI in disseminating video content for large and small field size, compared with low-energy adaptive clustering hierarchy (LEACH) and power efficient multimedia routing (PEMuR) in terms of network lifetime and video quality level.

In [29], Diaz et al. propose a new multimedia-oriented application layer protocol, which takes into account the multimedia services offered by the nodes in the wireless ad hoc network in order to select the best multimedia service provider node and to provide the best QoE and QoS to the nodes participating in the ad hoc network. Authors show the designed protocol and decision algorithms in order to provide the best multimedia service to the end users. Video streaming is more challenging problem than audio streaming. It requires a considerable bandwidth to provide enough QoS. The system takes into account the delay, jitter, lost packets, and bandwidth parameters in order to select the best service provider node. Moreover, the system takes into account the estimated QoE parameter (based on a previously studied formula) and the closest node which implies less RTT and thus lower zapping times, in order to have the best QoE. The authors validate their proposal through an implemented study case.

The protocol proposed in this paper is based on the architecture proposed by us for wireless ad hoc networks in [17]. While our previous work was based on the architecture definition and deployment, this paper is focused on the protocol including the restriction given by the sensor networks. Moreover, we have included the case where a request can be performed from outside the WSN (like in a regular WSN) and the tests to perform the real experiments are completely different.

3. Protocol Description

In this section we are going to describe the proposed protocol. First, we describe the architecture features, the elements of the framework, and their relationship. Then, we explain the characteristics of the protocol, the structure of the protocol header, and the protocol fields. Finally, we show the messages designed for the proper operation of our proposal.

The main objective of the protocol is to let the sensor nodes communicate taking into account multimedia flow characteristics. It uses a cluster-based ad hoc architecture that will control the QoS parameters for each multimedia communication, by establishing the appropriated values and guaranteeing the service along the time. The protocol allows the sensor to communicate and exchange information about their state and properties. Moreover, sensor nodes use this information to determine the most appropriate neighbors. The protocol dynamically manages the creation of the cluster as a function of the network features, the number of devices, the sensor capacity, and the multimedia flows.

3.1. System Architecture. The starting point of our system is a set of wireless sensor nodes located in a delimited place which form a WSN. Each wireless sensor node has different power, processing, memory, and transmission capacities. They are able to select other wireless sensor nodes as ad hoc neighbors if they are under their radio coverage area. Wireless sensor nodes are responsible for retransmitting the multimedia flows, which may be audio or video, and can use a wide range of codecs.

Figure 1 shows the elements of a cluster. Some sensor nodes are able to provide sensed data to the WSN as audio IP or video IP services. There are three types of communications as a function of the source or destination of the communication: (1) communication started from outside the WSN to a node placed inside the WSN, (2) communication started from a node placed inside the WSN to a destination outside the WSN, and (3) communication started from a node placed inside the WSN to a node of the WSN. A node from an external network can provide multimedia contents and audio and video real-time communication services.

Wireless sensor nodes can sense multimedia data or act as a data forwarding nodes inside the WSN. They can communicate with other nodes under their coverage area. New nodes will select the better reachable cluster based on their features and the type of multimedia traffic that is going to be transmitted. We can distinguish in Figure 1 two types of nodes, sensor nodes that do not have any connection with nodes from an external network (they can only establish connections with nodes from their cluster) and sensor nodes

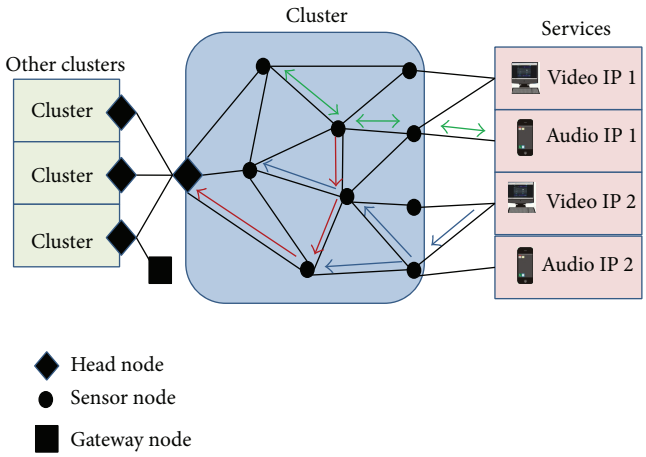

Figure 1: Cluster elements and possible communications.

that have connections with other clusters (cluster heads) or with an external network (gateway nodes). Gateway nodes have two interfaces at least in order to connect with the WSN and with the external network.

The network is organized in clusters. Every cluster of the architecture is dedicated to a specific multimedia flow, which will be identified by predefined multimedia profiles. We have created a Multimedia Init Profile (MIP) in order to manage the configuration of the sensor nodes [26]. MIP defines which type of multimedia flow can be delivered by the sensor node. MIP groups in a single logical component all required information to guarantee the adequate QoS to the multimedia traffic. MIP gathers the restrictions that will be applied to QoS parameters for the multimedia flows (Bandwidth, Delay, Jitter, and Lost Packets) and the cluster properties (maximum number of hops and the number of connections with external networks). There is only one MIP associated with each cluster in the WSN, but there could be several clusters using the same MIP. When a MIP is assigned to a sensor node, the following information is assigned: type of multimedia traffic (audio or video), range of codecs that can be used by the multimedia flows inside the cluster, maximum bandwidth available for retransmissions, and the maximum admissible Delay, Jitter, and Lost Packets.

Figure 2 shows the elements of the proposed architecture and their relationship. The architecture defines three operation levels: *Hardware Infrastructure*, *Logic Management,* and *Admin Interface*.

Hardware Infrastructure level is formed by the elements in charge of building the physical and logical network topology. The physical topology is made of wireless sensor nodes. Each node can be head node, gateway node, or sensor node (each node can only have one role). When a sensor node starts for its first time, it searches other sensor nodes in its coverage area. This process lets the node exchange the required information to group the nodes in clusters by means of the developed protocol. Then, the logical topology is created. Sensor nodes can only belong to a single cluster and have neighborhoods with the nodes of that cluster. The head node belongs to a single cluster but can have neighborhoods with other clusters' head nodes. The wireless connections between head clusters create a higher hierarchical level that allows exchanging information between clusters. The criteria used to determine which cluster will be the sensor node joined to are based on the MIP associated with the sensor node, and thus on the type of multimedia traffic it is disposed to retransmit. The sensor node will only establish neighborhoods for multimedia traffic delivery with other sensor nodes in the WSN that are using the same MIP, so all sensor nodes in the same cluster will have the same MIP. Head nodes exchange information and control messages with other head nodes. They maintain a database with existing head nodes and clusters and the MIPs associated with them. They will deliver multimedia data to other head nodes only if the destination cluster head node has the same MIP and will retransmit the multimedia flow to the cluster nodes if the multimedia flow belongs to the same MIP. A regular node can become a cluster head when the cluster head leaves the network or fails down.

The *Logic Management* level defines the protocol elements to manage the elements of the *Hardware Infrastructure*, by using the information received from the *Admin Interface* level. MIP is the logic element that gathers the information about the multimedia streams permitted in the cluster. It is the central element of the *Logic Management* level. In this level, the logical processes (discovery process, adjacency process, and forwarding process) that act over the sensor nodes as a function of their current state are also defined. When a sensor starts, it received the configured MIP from the *Admin Interface* level; then the discovery process is started and the node tries to find other nodes with the same MIP inside its coverage area. When it discovers other nodes, the adjacency process is started in order to create a neighborhood between both sensor nodes. These steps are followed by all new nodes in order to build the cluster. When a cluster is formed, it has the capacity to retransmit multimedia flows according to the ones defined in its MIP. Forwarding process is started when a sensor node creates a multimedia flow request or when a multimedia flow request is received from outside of the cluster (from other cluster or from outside the WSN through the gateway). It establishes the path to follow through the cluster and reserves the resources in every node belonging to the path. It makes possible the multimedia delivery and is responsible for guaranteeing the required QoS by the MIP during the communication.

Admin Interface level allows the interaction between the user and the sensor device. There is a graphic user interface (GUI) that lets the user modify the sensor init configuration, including the IP addressing and MIP selection. *Admin Interface* Level allows controlling manually the init process and disconnect process. The application also lets the user connect or disconnect the node to the WSN. The user can only make changes before the init process starts, so if a change is required, the sensor node must be stopped by using the disconnect process. Then, it should be initiated using the init process.

The number of available MIPs that can be selected by a sensor node, as well as the properties of each MIP, must be

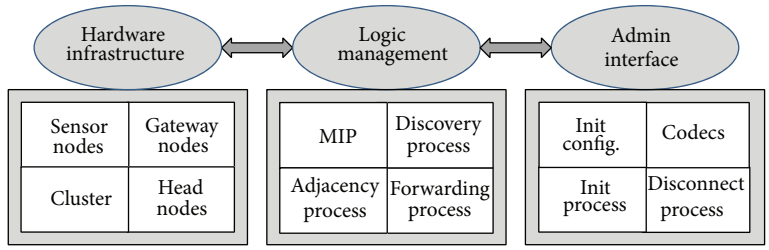

Figure 2: Elements of the architecture.

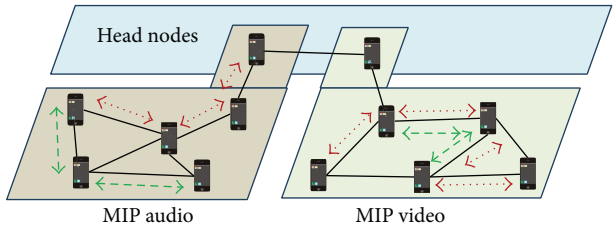

Figure 3: WSN structure based on MIP clusters.

Table 1: Protocol header.

1 byte		1 byte	0–255 bytes
2 bits	6 bits	8 bits	0–2040 bits
Version	Type	Length	Value

defined before the system is started. Each MIP represents a different type of multimedia traffic, so the MIP should be created taking into account the network characteristics, such as the nodes density, their location, node distribution, and radio coverage, jointly with the characteristics of the multimedia flow: type of traffic (audio or video), used codec, and QoS requirements. MIP definition is adapted to each particular case. For example, in a network topology with low nodes density and mixed video and audio flows, only two MIPS can be defined, one to create a cluster for audio delivery and another cluster for video delivery. But, if there is a network topology with high sensor nodes density, only dedicated to video delivery, but using a great variety of codec, several MIPs will be defined to split the multimedia flows that use video codecs in different clusters. The MIP assigned to the sensor includes the following information: maximum bandwidth (MaxBW) dedicated by the sensor node for retransmitting multimedia flows, minimum bandwidth (MinBW) required by a single multimedia flow to be processed, maximum delay (MaxDelay) permitted for the multimedia flow from the source to the destination, maximum jitter (MaxJitter) for a single multimedia flow and maximum hops (MaxHops) for a message in the WSN. Each MIP is identified by one-byte hexadecimal code, called HCode, and an alphanumeric code, called ACode.

Figure 3 shows the WSN MIP-based cluster structure. We have defined two MIPs: first one for audio flow delivery and the other for video flow delivery. Inside each cluster there could be simultaneous flow delivery with similar characteristics because they use the same MIP.

3.2. Protocol Fields. The developed protocol is included in the application layer of the TCP/IP stack protocols. UDP is chosen as encapsulation protocol at the transport layer in order to reduce the processing load of the sensor node, the bandwidth consumption, and the delay of the packets.

We wanted a simple protocol, with few fields, although it should be versatile. Protocol modifications should be easily made without big changes in the packet structure. Thus, we used the TLV (type-length-value) coding technique for the protocol implementation. TLV allows us to create new types of messages quickly and easily.

In Table 1, the protocol header fields are shown. We have included Version, Type, Length, and Value. The Type field allows us to interpret a received message. The information included in each type of message is variable and depends on the message objective, transmitter sensor node role, and receiver sensor node role. Generally, the size of the message is variable, so we have defined the Length field. It provides the length of the information carried at the Value field. Using TLV coding techniques increases flexibility and scalability of the protocol and these types of messages can be extended or be redefined in future revisions of the protocol very easily.

The protocol header fields are described below in greater detail.

(i) Version. This field provides the version of the protocol. Each version matches a specific and well-defined messages list. All devices in the WSN must use the same protocol version to communicate properly. The size of Version field is set to two bits in order to keep reduced to the size of the protocol message. The default value of the Version field is "00," which matches the protocol version 1.

(ii) Type. It is a numeric code used to identify the message type. Each message Type is defined in the specific protocol version. There is a message table which includes information about message length and how the message information carried at the Value field has to be interpreted on the reception

side. The size of the Type field is 6 bits, allowing a maximum of 64 message types.

(iii) Length. This field indicates the length of the Value field. The numeric value is given in bytes. The Value field is variable and its size depends on the type of message. The size of the Length field is one byte; the values range goes from 0 to 255. When the value of Length field is 0 it shows that the Value field does not exist; that is, it does not need to transmit any additional information.

(iv) Value. This field holds the information to be exchanged between the sensor nodes. The size of the field can take values between 0 and 255 bytes that matches the values of the Length field.

3.3. Data Structure. In order to carry out the required processes performed by the proposed protocol, the wireless sensor nodes have to exchange information. We have developed different types of messages with the purpose of performing next functions: exploring the network looking for devices with similar multimedia streaming purpose, creating sensor nodes adjacencies in order to build the cluster topology, sharing information about the sensor nodes status, their tables, and other network parameters, to start and run the multimedia flows through the cluster and notify to the neighbor nodes any event. Each defined message establishes the additional information to be included in the Value field. The following variables and structures were defined to facilitate the management of information.

(i) NODE_ID. It is the node identifier. This identifier must be unique across the whole network. The NODE_ID parameter size is 2 bytes and its value should be set before the init process starts, at the initialization process. There are 3 different mechanisms to generate a sensor NODE_ID: (1) static configuration, the identifier is manually defined by the sensor administrator; (2) automatic configuration, the last two bytes of the IP address are used as NODE_ID; and (3) dynamic configuration, where a network service uses the Multicast IP address 239.100.100.255. This last configuration option requires the previous configuration of one or more nodes as servers with preconfigured tables in order to assign the NODE_ID. This option has been only designed for testing and to facilitate the research work, but it is discouraged to use it in real environments because it introduces the need of servers.

(ii) NODE_RESOURCES. This variable contains information about the available bandwidth of the sensor node for multimedia delivery. The size of this variable is two bytes. The bandwidth is measured in Kbps. The initial value of the variable is set to the MaxBW value of the assigned MIP. When a new resource reservation for multimedia delivery is made, the NODE_RESOURCES value is decremented until the resource reservation is canceled or the delivery ends. When the NODE_RESOURCES value is below the MinBW parameter then the node changes its value to zero and no new multimedia delivery is allowed.

(iii) NODE_ADJ. It is a data structure representing the connectivity state of a node into the cluster. The size of this parameter is variable and ranges between 1 and 511 bytes. First byte shows the number of adjacencies of the node in that moment. Then, the data structure is built by concatenating the NODE_ID of the neighbor node who has established a successful adjacency with. When the node starts and it has not still been established any adjacency, the initial value of NODE_ADJ is set to 0x00 and is 1 byte in size. When the first adjacency is created the first byte is changed to 0x01 and the neighbor NODE_ID value is joined. From this point, every time a new adjacency is created, the first byte will be incremented and the new NODE_ID value will be concatenated to the NODE_ADJ structure. Because of data structure limitations, the maximum number of adjacencies by a node is limited to 255 adjacencies.

(iv) NODE_NCON. It indicates the total local number of properly established and active adjacencies. The size of the variable is 1 byte. Its initial value is set to 0x00. This variable matches the value of the first byte on the NODE_ADJ data structure. NCON value is incremented or decremented each time an adjacency is created or destroyed.

(v) NODE_NSEQ. This variable represents the version number of the state table of the sensor device. The parameter size is 2 bytes. When the sensor node starts, the initialization process set its value to 0x0000. When a state change occurs, for example, when an adjacency with other node of the WSN is created or destroyed, the NODE_NSEQ value is increased or decreased. Then, the system sends a cluster state update (CSU) message to all nodes with successful adjacencies to update their state table. When a CSU message is received, the node compares the NODE_ID and NODE_NSEQ values on the received message with the information stored in its state table. If the value of NODE_NSEQ for this NODE_ID in the state table is below the received value, the state table is updated with the information included in the CSU message. Then, the message is forwarded to all local adjacencies except to the neighbor that sent the original CSU message. If NODE_NSEQ of the CSU message is equal to or lower than the values of the state table, the CSU message is discarded.

(vi) NODE_STATE. It is a data structure created by concatenating other local variables and structures: NODE_NSEQ, NODE_ID, NODE_RESOURCES, and NODE_ADJ variables. The data structure size is calculated as a function of the number of adjacencies established by NODE_NCON value, and it ranges between 7 bytes, when there is not created any adjacency, and 517 bytes, when the maximum value of adjacencies has been reached.

(vii) CSU_NSEQ. This variable is a sequence number value used in CSU messages in order to allow message fragmentation. When the size of the information in the state table cannot be fit into a single message, the CSU sequence number

allows fragmenting the information into multiple messages sequentially numbered. The field size is 1 byte and the default value is set to 0x80 when no fragmentation is needed. When fragmentation is used, the packets are consecutively numbered starting from 0x01. Possible values range between 0x01 and 0xEF. When the last fragment is sent, the sequence number is increased from the previous message and, then, the first bit is changed to "1" indicating that this is the last fragment of the sequence.

(viii) CLUSTER_MIP. The CLUSTER_MIP value matches the HCode value of the assigned MIP. This is a 1-byte variable. This parameter is exchanged between neighbor nodes in the adjacency process. The MIP table is defined for the whole WSN. The number and characteristics of the defined MIP depend on the traffic pattern and multimedia flows of the network.

(ix) CLUSTER_N. This parameter is used to distinguish two different clusters using the same MIP. The size of the variable is 1 byte. When the first cluster node creates the cluster and it is not aware of other clusters with the same MIP, then it selects the CLUSTER_N value equal to 0x00. Next, the first cluster node sends a request to existing cluster heads in order to know their CLUSTER_N. After this step it becomes cluster head and adds next available value from the received replies to its CLUSTER_N parameter.

(x) CLUSTER_ID. It is the cluster identifier. This value must be unique for each cluster within the same WSN. Two independent clusters into the WSN can share the same MIP but they must always have different CLUSTER_ID value. The size of the variable is two bytes. Its value is established by the first node in the cluster. The first node is defined as the node that receives the discovery message ACK to establish the first cluster adjacency. The CLUSTER_ID value is built by concatenating two variables, CLUSTER_MIP and CLUSTER_N. In case of CLUSTER_ID duplications in the same WSN (because of lost messages or formed cluster joining), the oldest cluster keeps its CLUSTER_ID, and the youngest cluster changes its value to the next free value. An update message is sent to all nodes into the cluster to notify and update the new CLUSTER_ID.

(xi) CLUSTER_DIAMETER. This variable shows the current cluster diameter. The cluster diameter is defined by the highest value of the lowest distance between any two nodes in the cluster. Distance between two sensor nodes is calculated by the routing algorithm. It is measured in number of hops. The size of the variable is 1 byte. When a sensor node starts, it has not established any adjacency yet, and then the CLUSTER_DIAMETER value is set to 0. Later, when the first adjacency in the cluster is created, the value is changed to 1 on both nodes. Each time a new sensor node is added to the cluster topology, the CLUSTER_DIAMETER is recalculated using the routing protocol in order to guarantee that it does not overcome the Maxhops value established in the MIP cluster. If the new adjacency exceeds Maxhops, then adjacency fails to be established.

(xii) MEDIA_RESOURCES. This parameter identifies the bandwidth resources needed for a single multimedia communication. This variable is used when the sensor node creates and processes a new delivery request. Possible values can vary from MinBW to MaxBW of the assigned MIP. It depends on the characteristics of the codec used for multimedia delivery. Its value represents the bandwidth measured in Kbps and it is 2 bytes long.

(xiii) MEDIA_SOURCE. When a request for resource reservation takes place, the NODE_ID value of the source node (SN) is copied in this variable. SN is the sensor node where the multimedia delivery was originated inside the cluster. Like NODE_ID variable, the MEDIA_SOURCE size is 2 bytes. The origin of the multimedia delivery can be located outside the WSN; in this case the SN is defined as the gateway node used to enter the WSN.

(xiv) MEDIA_TARGET. This variable carries the NODE_ID value of the target node (TN). In a similar way as the SN was defined, the TN is the sensor node where the multimedia transmission ends inside the cluster. Its size is also 2 bytes. As in the previous case, the multimedia communication may finish outside the WSN, through a gateway sensor node connected to an external network. In this case, the MEDIA_TARGET is defined as the NODE_ID of the gateway node.

(xv) MEDIA_ROUTE. This structure contains the full route for a multimedia packet flowing from the MEDIA_SOURCE to the MEDIA_TARGET. It is built by adding every sensor NODE_ID on the route. The route is calculated by the routing algorithm. Its size can vary from 4 bytes, when SN and TN have established a valid adjacency, to 32 bytes, when there are 16 hops on the route, the maximum number of allowed hops for any cluster. The first NODE_ID used to build the structure is the MEDIA_SOURCE and the last matches the MEDIA_TARGET.

(xvi) MEDIA_NHOP. This variable has the number of hops between MEDIA_TARGET MEDIA_SOURCE as it is calculated from the routing algorithm in the MEDIA_SOURCE sensor node. The size of this parameter is 1 byte. The maximum number of hops allowed by the protocol implementation inside a single cluster of the WSN is set to 16 hops. However, the number of hops between any two nodes on a specific cluster can never be greater than the CLUSTER_DIAMETER parameter (as it is defined in the assigned MIP).

(xvii) MEDIA_NSEQ. This is the sequence number assigned to a multimedia delivery for the source node. The size of the variable is 2 bytes. The initial value is set to the hexadecimal value 0x0000. Each time a new request for multimedia delivery is originated in a sensor node the MEDIA_NSEQ value is incremented by one. This variable allows the protocol to differentiate between several multimedia flows being delivered simultaneously from the same source node.

(xviii) MEDIA_INFO. This is a data structure that contains the whole information for a single multimedia delivery that is needed and used by the remaining cluster sensor nodes. It is built on the SN when a new multimedia request is originated. The following parameters and structures are added in order to build the MEDIA_INFO structure: MEDIA_RESOURCES + MEDIA_NSEQ + MEDIA_NHOP + MEDIA_ROUTE. The size of the data structure depends on the number of hops on the route indicated by the MEDIA_NHOP. The size can vary between 11 and 39 bytes.

3.4. Message Table. The messages used by the protocol are described in this section. Here we define the version 1 of the protocol. The TLV coding used by the protocol encapsulation allows us to change the list of messages in the following versions. For a better understanding, the whole list of messages has been organized considering the system process they belong to. UDP protocol is selected at the transport layer. Despite this, relevant messages need to be confirmed. For example "ACK Discovery" is a confirmation message for the "Discovery" message and "Confirm Join" message confirms the "Request Join."

Table 2 shows the protocol messages used at the adjacency process. Messages belonging to the adjacency process are shown on Table 3. Table 4 describes the forwarding process messages and the disconnect process messages are listed on Table 5. System processes are detailed in the next section.

4. System Operation

This section details the protocol operation. There are four main processes: discovery, adjacency, forwarding, and disconnect.

4.1. Discovery Process. Figure 4 shows the messages exchanged in the discovery process. A sensor node starts the discovery process when the sensor node initialization process has finished. This sensor node is named new node (NN). The NN changes to the discovering state and it begins sending messages looking for other sensor nodes. A "Discovery" message is sent every 60 seconds. If there are not answers after three messages, the sensor node stops sending "Discovery" messages. The Value field at the "Discovery" message has the NODE_ID, to identify the CN, and the CLUSTER_MIP to inform the selected MIP. Messages are sent to the Multicast IP address "239.100.100.CLUSTER_MIP," where the last byte matches the CLUSTER_MIP parameter. Thus only sensor nodes with the same MIP and listening to the Multicast IP address, will receive the messages. The receiver sensor nodes are called border cluster node (BCN). BCN replies by sending the "ACK Discovery" message to the NN. The "ACK Discovery" messages are sent to the unicast IP address of the new sensor node and the multicast address is not used anymore. The "ACK Discovery" message has the following information: The BCN NODE_ID, the NODE_NCON that shows the amount of established adjacencies, the CLUSTER_ID to identify the cluster, and the CLUSTER_DIAMETER. When the NN receives the "ACK Discovery" message, it compares the CLUSTER_DIAMETER with the MAX_HOPS parameter; if both values are equal, then the adjacency process finishes here. If two of more clusters are available, the NODE_NCON information is used by the new sensor node in order to select the most appropriate cluster to connect with. The lowest value is preferred.

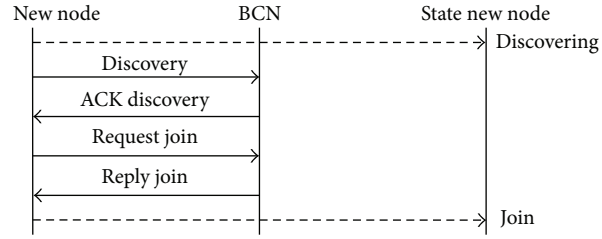

Figure 4: Discovery process.

The new sensor node keeps waiting at least for 60 seconds after sending the first "Discovery" message and before selecting the target cluster; thus it allows arriving on time other possible "ACK Discovery" messages from different BCNs. When a valid cluster is discovered, and it is selected, the candidate sensor node sends a "Request Join" message to the selected sensor node (or nodes if they belong to the same selected cluster). This message contains the BCN NODE_ID that it is looking to build the adjacency, the CLUSTER_ID it wants to join and the available resources in the candidate sensor node through the NODE_RESOURCES parameter. The BCN uses the information about the NN resources to update its own state table and to notify the other sensor nodes in the cluster topology. Then, it replies the NN by sending a "Reply Join" message; it changes to the Join state and the discovery process ends. If two or more BCNs from the same cluster are discovered the sensor node will have adjacencies with all of them.

4.2. Adjacency Process. The state table holds the information about all sensor nodes belonging to the same cluster. All sensor nodes in the same cluster share the same state table. There is a table entry for each sensor node in the cluster; thus when a new adjacency appears, the full state table is exchanged. Each table entry is stored in a single NODE_STATE structure. This structure keeps the following information about a single sensor node: NODE_ID, available resources, number of adjacencies, neighbors NODE_ID, and NODE_NSEQ. If it is the first adjacency of the new sensor node, then there is only one entry on its state table and this is about its own link-status information.

The Adjacency process begins when the new sensor node makes a transition to the Join state. The exchange of messages in the adjacency process is displayed in Figure 5. This image represents the specific case when the adjacency between two sensor nodes, a NN and a BCN, is successfully completed. The inside cluster node (ICN) is defined as any other sensor node inside the cluster that is not going to build a direct adjacency with the new sensor node. A "Cluster State Update (CSU)" message is sent from the NN to

Table 2: Discovery process messages.

Discovery process				
Message	Type	Length	Value	Description
Discovery	0x01	3 Bytes	NODE_ID CLUSTER_MIP	Message looking for neighbors to join or to create a cluster
ACK discovery	0x02	6 Bytes	NODE_ID NODE_NCON CLUSTER_ID CLUSTER_DIAMETER	Confirms the CLUSTER_ID available to join
Request join	0x03	6 Bytes	NODE_ID NODE_RESOURCES CLUSTER_ID	Sensor node sends a request to build a new adjacency
Confirm join	0x04	4 Bytes	NODE_ID CLUSTER_ID	Sensor node confirms the request to join

Table 3: Adjacency process messages.

Adjacency process				
Message	Type	Length	Value	Description
Cluster state update (CSU)	0x11	10–255 bytes	CSU_NSEQ NODE_ID NODE_STATE NODE_STATE ... NODE_STATE	To exchange the state table information between adjacency nodes. When state table information exceeds the 255-byte limit; it is fragmented in two or more messages.
ACK cluster state update (ACK CSU)	0x12	3 bytes	CSU_NSEQ NODE_ID	Confirmation message of a CSU message
Node state update (NSU)	0x13	8–255 bytes	CSU_NSEQ NODE_STATE	Information about the state of a single node. If the NODE_STATE variable exceeds 255 bytes information is fragmented in two or more messages.
ACK node state update (ACK NSU)	0x14	3 bytes	CSU_NSEQ NODE_ID	Confirmation message of a NSU message
Cluster join	0x15	4 bytes	NODE_ID CLUSTER_ID	The node has filled its state table with the cluster information and it requests to join the cluster
ACK cluster join	0x16	4 bytes	NODE_ID CLUSTER_ID	Confirmation message of a cluster join message

Table 4: Forwarding process messages.

Forwarding process				
Message	Type	Length	Value	Description
Request forwarding	0x21	13–41 bytes	NODE_ID MEDIA_INFO	A new multimedia flow requests a resource reservation. This message is sent from the source node to the target node
Reserve resources	0x22	13–41 bytes	NODE_ID MEDIA_INFO	Target node sends a resource confirmation to the source node
Confirm reserve resources	0x23	13–41 bytes	NODE_ID MEDIA_INFO	Reservation confirmation from the source node to the target node
Queue reserve	0x24	13–41 bytes	NODE_ID MEDIA_INFO	Notification message when multimedia flow is placed in queue
Reject reserve	0x25	13–41 bytes	NODE_ID MEDIA_INFO	Reservation cancelation
End transmission	0x26	13–41 bytes	NODE_ID MEDIA_INFO	Multimedia delivery is finished and resources have to be released

Table 5: Disconnect process messages.

Disconnect process				
Message	Type	Length	Value	Description
Disconnect	0x31	2 Bytes	NODE_ID	A node is breaking an adjacency
ACK disconnect	0x32	2 Bytes	NODE_ID	Confirmation of the disconnect message

the BCN. The message is built with the NN NODE_ID and the CSU_NSEQ. The sequence number is used to fragment the "CSU" message information when necessary. Moreover, full information on the NN state table is included in the message; the NODE_STATE structure is used here. The "CSU" message needs always to be acknowledged by an "ACK CSU" message from the BCN; if any "ACK CSU" message is not received in 10 seconds, after the "CSU" message was sent, it is sent it again. If a sensor node sends the same "CSU" message three times, and it does not receive any answer, the adjacency process finishes unsuccessfully. After the "ACK CSU" message, the BCN sends its own state table information to the NN by sending one or more "CSU" messages. The state table is encoded in NODE_STATE structures as table entries. If the message size exceeds the limit of 255 bytes, then the information is fragmented to be sent on several "CSU" messages. The CSU_NSEQ value is used to allow fragmentation. Each "CSU" message needs to be acknowledged by an individual "CSU ACK" message in order to avoid losing information; neighbor sensor nodes need to keep the same state table; otherwise routing algorithm will not be able to calculate the most appropriated route between sensor nodes in the same cluster. After the state table has been fully exchanged between the NN and the BCN, the NN makes a transition to the associated state. This is a transitional state, both sensor nodes are sharing the whole cluster link-state information but they have not completed their adjacency yet. BCN has not updated any other ICN yet.

When the NN changes to the associated state it sends a "Cluster Join" message to the BCN. "Cluster Join" message has the CLUSTER_ID it is trying to join. The BCN updates its state table with the STATE_NODE information of the NN and it increases by one its NODE_NSEQ value. Then, the BCN sends a "Cluster Join ACK" message to the NN in order to accept the new adjacency. At this moment, the NN makes a transition to the established state, which means that it has joined the cluster. The adjacency process with the NN is completed. Finally, the NN information is flooded to the rest of the sensor nodes in the cluster by sending two "NSU" messages to all sensor nodes in the cluster. The main difference between "CSU" and "NSU" messages is that "CSU" message carries the full state table information, but the "NSU" message only carries an individual table entry for a single sensor node. In this case, two "NSU" messages should to be sent: one for the NN information and the other for the BCN updated information. Every ICN checks the NODE_NSEQ for each message; then, it updates its state table and, finally, forwards the "NSU" message to all its neighbors, except to the one it has received the message from. If the NODE_NSEQ in the received STATE_NODE structure is equal or greater than the NODE_NSEQ in the ICN state table "NSU" message is ignored. Each "NSU" message needs to be acknowledged by an "ACK NSU" message, even when the "NSU" message is ignored.

4.3. Forwarding Process. The forwarding process starts when there is a request for multimedia delivery in the cluster. Figure 6 shows the message flow diagram for the forwarding process. The example detailed in the figure explains how a new multimedia delivery is requesting a resource reservation. The request is queued by a starved node without enough resources and finally it is processed when resources are released at the queued sensor node. Source node (SN) is defined as the first sensor node in the cluster where the multimedia request takes places. SN can be a gateway node, if the request is generated outside the WSN, or it can be any other sensor node if the request is generated inside the WSN. Target node (TN) is the destination multimedia flow inside the WSN; it can be a gateway node if the IP address destination is outside the WSN. In the diagram, the first hop node (FHN) has been defined as the first cluster node on the path to the target node. FHN is calculated by the routing algorithm starting from the SN neighbors. In this case, ICN will be those nodes on the path between the SN and the TN.

When the forwarding process starts, the SN is in the established state or in the forwarding state and it receives a new multimedia flow request. First, it checks if there are enough local resources to process it. If the SN has enough available resources, the full path to the TN is calculated. The routing algorithm is used only once and only at the SN; the SN state table information contains the whole information about the cluster needed to establish each hop on the path; thus the path cannot be modified.

The message exchange starts when a SN sends a "Request Forwarding" message to the first hop node (FHN), which is the first NODE_ID on the calculated path. The message holds the SN NODE_ID and the MEDIA_INFO data structure. The MEDIA_INFO structure provides complete information about the multimedia request: MEDIA_RESOURCES, MEDIA_NSEQ, MEDIA_NHOP, and MEDIA_ROUTE. The ROUTE_MEDIA structure contains the NODE_ID from all hops on the path, from the SN to the TN. MEDIA_RESOURCES show the bandwidth resources required to enable to process the multimedia communication. MEDIA_NHOP matches the amount of hops on the path. MEDIA_NSEQ is the sequence number assigned by the SN to identify this particular multimedia flow.

The FHN receives the "Request Forwarding" message and checks if its NODE_ID is included in the MEDIA_ROUTE structure. If not, the message is discarded. If it is inside, the FHN checks it resources availability and the value is compared to the MEDIA_RESOURCES value. If there are enough local resources, the FHN reads the next NODE_ID on the hop list and the "Request Forwarding" message is forwarded to

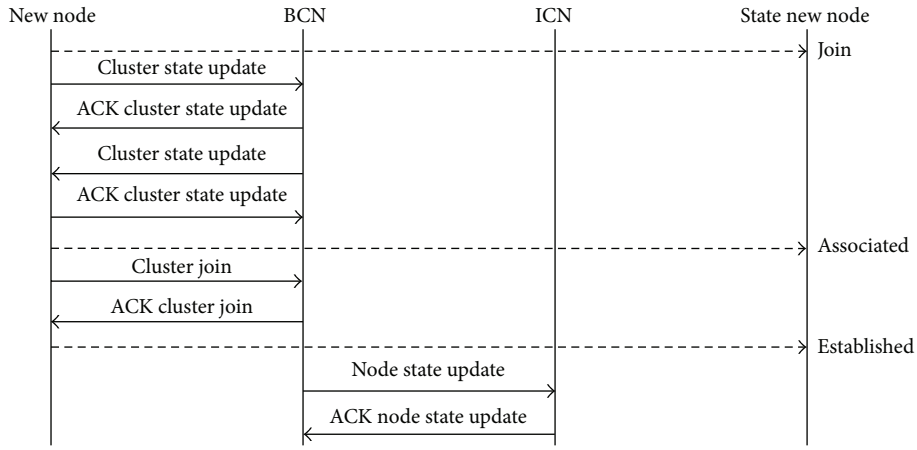

Figure 5: Adjacency process.

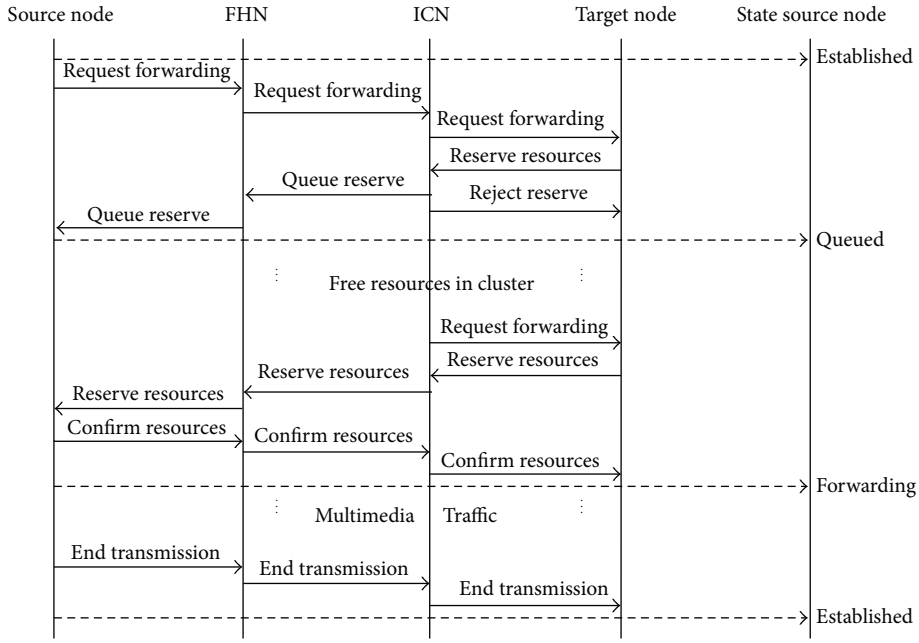

Figure 6: Forwarding process.

it. Resource reservation at the FHN is not set yet and can be used by current traffic, but the bandwidth resources of this request will not be used by other request reservation until the reservation is confirmed or rejected. All hops on the path perform the same process, hop by hop, in order to reach the TN. Finally, the TN receives the "Forwarding Request" message. The TN NODE_ID is compared with the last hop on the list provided by the MEDIA_ROUTE structure in order to check that the TN is included in this multimedia request. Available resources are checked as the other sensor nodes on the path. If there are enough resources, a reservation is made. The NODE_RESOURCES variable is decremented in the amount indicated by the MEDIA_RESOURCES parameter. This is a temporary reservation and it needs to be confirmed by the SN. Thus, the TN sends a "Reserve Resources" message back to the SN. This message also carries the MEDIA_INFO structure and it should follow the same path of the "Forwarding Request" message, but in the opposite direction. Each sensor node on the path performs a temporary reservation and follows the message back to reach the SN.

If a sensor node on the path cannot make the reservation because there is not enough bandwidth available to guarantee the multimedia communication, the designed protocol can put the request in queue for this sensor node. This process is shown in Figure 6, where the ICN decreases its bandwidth when it receives the "Reserve Resources" message. When an ICN is congested it can perform three different actions: it stores the request in a waiting queue, then it sends a "Reject Resources" message to the TN and finally, and it sends a "Queue Reserve" message to the SN. Both messages use

the MEDIA_ROUTE information in order to repeat the same path and to inform all sensor nodes in the path. The temporary resources reservation made in the sensor nodes between the ICN and the TN are cancelled by the "Reject Resources" message. On the other side, the "Queue Reserve" message releases the prereservation made at the sensor nodes between the SN and the ICN. The SN puts the multimedia request in a request queue and it waits to receive a notification from the congested sensor node when requested resources will be available. Both, "Reject Resources" and "Queue Reserve" messages have the NODE_ID field with the NODE_ID of the congested sensor node, ICN; thus all sensor nodes on the path can locate the congestion problems. Routing algorithm will not include congested nodes in the path. There are two options when the SN receives the "Queued Reserve" message: (1) it can wait for released resources in congested nodes or (2) it can calculate again the path to the TN but avoiding the congested sensor node. In this second case, the SN sends a "Reject Reserve" message to the congested sensor node; the waiting queue in the congested sensor node will be deleted. If the SN keeps the request in queue, then a timer is started in order to prevent a blocked multimedia communication. When the timer expires, a "Reject Reserve" message is sent to the congested sensor node.

If the congested sensor node keeps the request queued, it waits till other multimedia communications ends in order to have enough bandwidth resources. The forwarding process is started again from this point. Figure 6 shows the exchanged messages. Congested sensor node sends a "Request Forwarding" message to the TN. The original MEDIA_INFO is used. Then, a "Reserve Resources" message is sent back to the SN again from the TN. Finally, since all sensor nodes in this example have enough available resources to make the reservation, the "Reserve Resources" message reaches the SN. SN knows that all sensor nodes on the path to the TN have enough resources and they have made a temporary reservation to process the request. Then, SN sends a "Confirm Resources" message to the TN through the MEDIA_ROUTE and temporary reservations are confirmed. The SN changes to the forwarding state and the multimedia delivery begins.

When the multimedia delivery ends, SN sends an "End Transmission" message. This message carries the MEDIA_INFO structure, which is sent to the TN to inform each sensor node that the delivery has finished and the allocated resources can be released.

4.4. Disconnect Process. Figure 7 shows the exchanged messages in the disconnect process. Disconnect process is started by the sensor node to shut down or reboot. The sensor node sends a "Disconnect" message to each neighbor. Then, a 10-second timer is activated waiting the "ACK Disconnect" message. If no neighbor sends the "ACK Disconnect" message in the timer interval, then the "Disconnect" message is sent again until 3 times. After it, the sensor node leaves the cluster.

Next, neighbor sensor nodes update their status table. All information about the disconnected sensor node is removed. Then, they send a "NSU" message to their neighbors. The "NSU" message is flooded across the cluster, in order to let all sensor nodes update their state table. Every "NSU" message is acknowledged by the "ACK NSU" message.

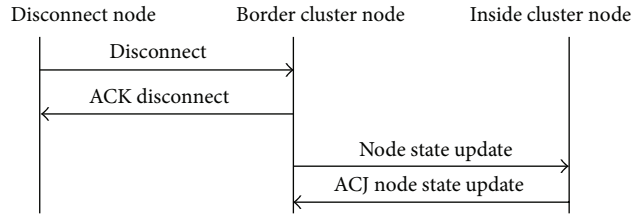

Figure 7: Disconnect process.

5. Performance Study

In order to validate the proposed algorithm we have designed and built a test bench. Our protocol organizes sensor nodes in four clusters: two audio clusters and two video clusters. Each cluster has assigned a different MIP. When a NN starts in the wireless network it knows the MIP that it belongs to. Then it tries to discover other sensor nodes with the same MIP and finally it joins the cluster. If it is the first sensor node in the network with this particular MIP the sensor node keeps waiting for new sensor nodes with the same MIP.

Several topologies arrangements have been studied in order to know how the quality of service parameters change when the diameter of the topology increases. The QoS parameters, delay, jitter, and packet loss have been measured for each MIP cluster in three experimental conditions: cluster diameter of one hop, two hops, and three hops.

Before the wireless sensors start, they have been configured with static IP address and wireless ad hoc network configuration, wireless channel, and interface speed. IEEE 802.11g standard has been selected as the wireless technology for the wireless sensor nodes.

The four MIPs are simultaneously working in the same WSN. Two audio MIPs have been selected: AUDIO_64K and AUDIO_192K. First, the AUDIO_64K matches the regular audio communications and audio IP calls performed through the PCM codification standard and the G.711 codec, the most compatible and widely used at all kind of audio applications and protocols. These deliveries offer a sound quality similar to the quality of a phone line. The AUDIO_192K MIP matches codecs used at high quality audio communications. With this kind of codecs it is possible to deliver music and human voice with nearly perfect quality.

For video deliveries we have chosen two MIPs: VIDEO_1500K and VIDEO_3500K. The first MIP, VIDEO_1500K, has been chosen because it represents the quality for a video delivery performed in high definition TV (HDTV) with 720p format. In the same way, the VIDEO_3500K is included because it is a typical standard delivery for 1080p format in HDTV.

5.1. MIP Comparison. The first test bench was set to find out if there are differences between multimedia deliveries belonging to different MIPs when they take place over similar

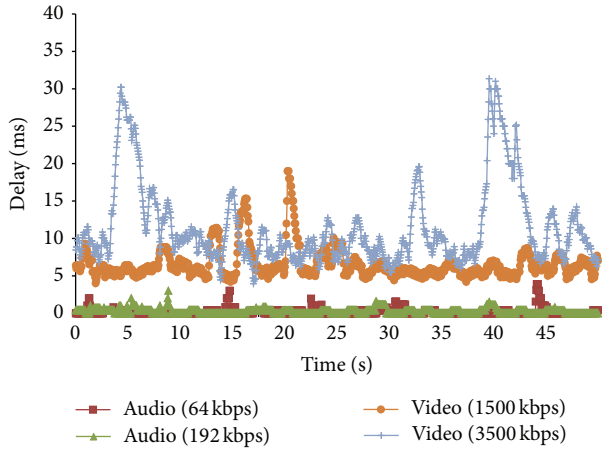

Figure 8: Delay as a function of the time for each MIP.

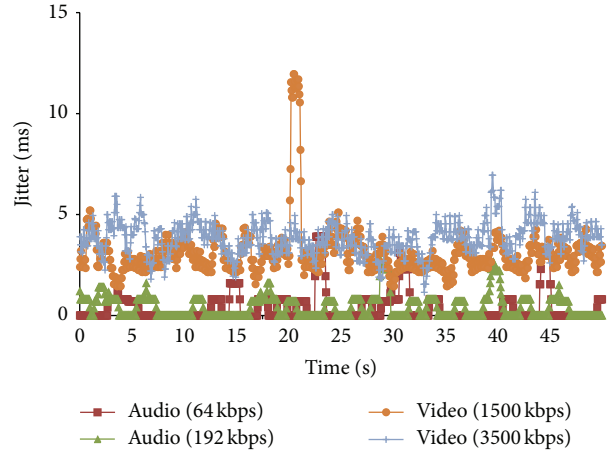

Figure 9: Jitter as a function of time for each MIP studied.

cluster topologies. In order to study the cluster behaviour in terms of QoS parameters for each MIP, the WSN topology was designed with the aim of building four clusters, one cluster for each MIP. In this experimental design the maximum number of hops for every cluster was established at two hops. In order to be able to compare the obtained results for each cluster, several environment variables and experimental conditions have been controlled: there are the same number of sensor nodes at each cluster, the same average distance between sensor nodes in each cluster, only one multimedia delivery is in progress at a time, and the noise level at the 2.4 GHz microwave band is measured and controlled. Figure 8 shows the delay measured for four different MIPs through clusters with identical characteristics but with different MIP settings. The figure represents the average delay of the last 20 samples received at any time. In order to estimate the average delay, we applied (1) on the obtained measurements:

$$Y_i = \frac{\sum_{j=i}^{i+20} X_j}{20}. \quad (1)$$

Both studied audio codecs have obtained similar delay results. Figure 8 shows that the average delay remains below 5 milliseconds when audio is delivered. These results indicate that the quality of audio transmission can be performed over this cluster without any loss of quality, even high quality audio with 192 Kbps. Results for video codecs seem not to be as good as for audio codecs. However, the average delay for video delivery is always below 30 milliseconds (there are two peaks of about 30 milliseconds at the 5th second and at the 40th second) and these values are enough to guarantee an excellent quality on video regular communications.

For the same number of hops, we observe that the delay is rising when the bandwidth spent for multimedia communication through the cluster grows. The behaviour of audio codecs compared to video codecs is clearly different. In order to determine if there is a significant difference between both audio codecs and between both video codecs, we need to make the statistical analysis of the experimental data. Table 6 shows the analysis results. The 99% confidence interval was calculated for an average delay of each experimental condition (α = 0,01). In order to establish relationship between each series, three null hypothesis were assumed: there are not differences between audio and video measures, there are not differences between two audio measures with different bandwidth consumption, and there are not differences between two audio measures with different bandwidth consumption

As expected, mean delay values are significantly different when any audio codec is compared with any video codec, so we can completely reject the null hypothesis and accept the alternative hypothesis: difference between delay of audio and video MIPs has statistical significance. In the same way, when the mean delays for both video MIPS are compared a statistical significant difference can be concluded. However, when audio MIPs with different bandwidth consumption, 64 Kbps and 192 Kbps, are compared, it is not possible to deduce any significant difference because the mean delay of one audio MIP is inside the confidence interval of the other MIP. In the last case, it is not possible to reject the null hypothesis at least with $P = 0.01$.

Figure 9 shows the jitter obtained in the experimental tests. As it happens with the delay results, we can see that the jitter for audio cluster is significantly lower than the jitter for video delivery. The second important result is that all multimedia delivery has jitter values below 15 milliseconds. Only few samples are over the 10 milliseconds. The quality of a multimedia communication can be affected by jitter values when they are as low as 20 or 30 milliseconds, but it is possible to easily manage a jitter value of 15 milliseconds building a buffer in the receiver side to eliminate its harmful effect.

Data was analyzed to know if there are some significant differences between two similar MIPs, that is, two audio MIPs or two video MIPs. Table 7 shows the statistical parameters for each data series with $\alpha = 0.01$. Statistical inference has been conducted as the previous delay analysis. Mean jitter values for AUDIO_64K and AUDIO_192K are very similar. Even the AUDIO_192K shows a mean jitter a bit bigger than AUDIO_64K. However, mean value of the first data series is included at the confidence interval of the second and vice versa. Null hypothesis cannot be rejected and it is not possible

Table 6: Statistical values and confidence interval for delay as a function of the MIP.

MIP	Parameters						
	N	μ (ms)	σ (ms)	Min (ms)	Max (ms)	Confidence interval (ms)	
AUDIO_64K	1000	0.276	0.696	1	4	0.204	0.348
AUDIO_192K	1000	0.205	0.282	1	4	0.175	0.234
VIDEO_1500K	1000	6.588	2.089	4	19	6.370	6.805
VIDEO_3500K	1000	11.461	5.509	5	31	10.887	12.034

to deduce any difference between both audio data series. By contrast, differences between mean jitter for both video MIP can be accepted. Null hypothesis is rejected in this case. Moreover, the null hypothesis is rejected between the mean jitter values of audio and video MIPs.

Based on these results, we can conclude that, in this experimental setup, delay and jitter parameters obtained using different video MIPs are different. Moreover, obtained QoS parameters of video MIPs are different than the QoS parameters of audio MIPs. These results confirm the benefits to divide the whole WSN into several clusters based on MIP configuration. Clusters with multimedia video traffic have a different QoS behaviour as a function of the features of the video delivery and they are also different than the audio delivery. Keeping separate multimedia flows through the MIP architecture allows the network to improve the delay and jitter parameters for multimedia delivery with low requirements.

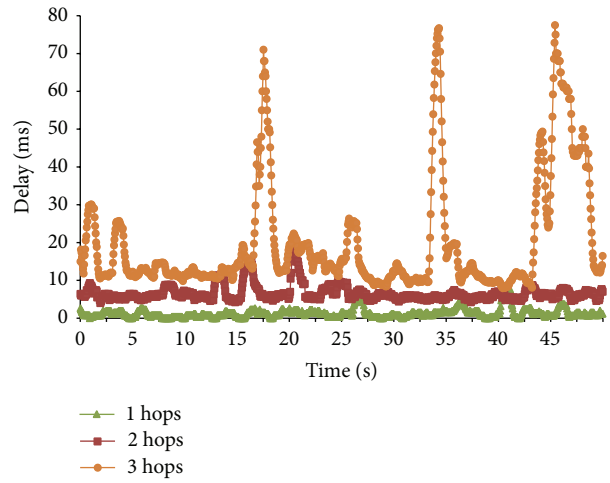

Figure 10: Delay as a function of time for different number of hops.

5.2. Cluster Comparison. In this second experiment we have studied the number of hops in the cluster. In order to perform this study only one MIP was selected VIDEO_1500K. The WSN topology was modified to achieve three different cluster diameters. Multimedia delivery was always performed through the maximum number of hops allowed in each cluster topology. The number of hops selected for the three experiments were: one, two, and three hops.

Figure 10 shows the results obtained for the delay as a function of the time in the three cases. The main result was that the delay is worse when the number of hops increases, but three-hop case has very high peaks, which might be taken into account. Delay for 1 hop delivery is minimal; values were only few milliseconds above zero, and there was not any big value in the whole series; all values were below 100 milliseconds. Delay for 2-hop condition moves between 5 and 10 milliseconds; there were some peaks on the graph; however they are of small size. Finally, delay for 3-hop transmission was the biggest with values between 10 and 20 milliseconds; there are also many peaks with mean values above 70 milliseconds. Multimedia delivery can be optimal with values of up to 150 milliseconds; above this limit quality of service would be decreased. However, it should be noted that the measured delay is only the delay introduced by the sensor nodes transmission on the cluster, but in a real case there are other processes and transmissions out of the cluster that need to be considered to calculate the final delay.

In order to corroborate the correct interpretation of these results, a statistical analysis was performed. Table 8 shows the statistical analysis. In the inference analysis, $\alpha = 0.01$ is assumed and the 99% confident interval was calculated for each experimental condition. Two null hypotheses have been stated: there are no differences between the mean delays in the cluster with one hop and the cluster with two hops, and there are no differences between mean delay in the cluster with one hop and the cluster with two hops.

Mean delay of each series is outside the confidence interval of the remaining cases. Both null hypotheses can be fully rejected with 99% probability. It is possible to affirm that the mean delay value through a two-hop cluster is bigger than through a one-hop cluster, and the mean delay value through a three-hop cluster is bigger than through a two-hop cluster.

Jitter values are shown in Figure 11. We can see that the values for data series of 1 and 2 hops are very similar, with an average below 5 milliseconds. Otherwise, the 3-hop series shows higher values, around 10 milliseconds, but it is always below 15 milliseconds. These jitter results have been obtained through the control of some experimental conditions: there was only one delivery, reduced noise level, and so on. But in a real environment there are a lot of variables that can affect the multimedia delivery. Thus, a 10 milliseconds level of jitter obtained in these ideal conditions should be interpreted with caution.

Jitter measures for one and two hops are very similar and we need to make the statistical inference analysis to know the relationship between these data series. The analysis is conducted following the same criteria than the previous delay analysis. It is shown in Table 9.

Table 7: Statistical values and confidence interval for Jitter as a function of the MIP.

MIP	Parameters						
	N	μ (ms)	σ (ms)	Min (ms)	Max (ms)	Confidence interval (ms)	
AUDIO_64K	1000	0.412	0.784	0	4	0.348	0.475
AUDIO_192K	1000	0.410	0.556	0	3	0.365	0.455
VIDEO_1500K	1000	3.104	1.315	1	12	2.996	3.211
VIDEO_3500K	1000	3.824	0.786	1	7	3.760	3.888

Table 8: Statistical values and confidence interval for delay as a function of the number of hops.

Hops	Parameters						
	N	μ (ms)	σ (ms)	Min (ms)	Max (ms)	Confidence interval (ms)	
1	1000	1.374	1.269	0	8	1.271	1.477
2	1000	6.589	2.089	4	19	6.419	6.759
3	1000	19.797	15.119	8	78	18.565	21.028

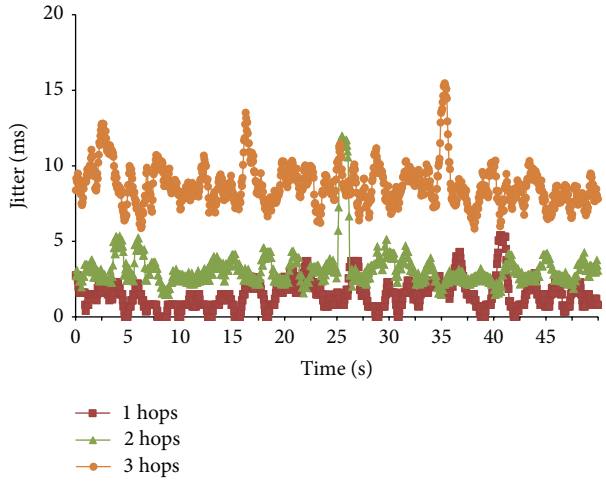

Figure 11: Jitter as a function of time for different number of hops.

As it happens in the case of delay, null hypothesis can be rejected and the alternative hypothesis is accepted. Mean jitter values for 2 hops series data is significantly bigger than for 1 hop, and mean jitter value for 3 hops is bigger than for 2 hops.

From these results we can conclude that, in this experimental environment, the cluster diameter may negatively affect the multimedia traffic QoS parameters. The proposal architecture and the developed network protocol can improve QoS parameters by minimizing the maximum number of hops into the cluster.

5.3. Packet Loss Study. Table 10 shows packet loss percentages for each experimental case. We can see that there is very few percentage of lost packets when there is only one hop in the WSN. When the topology becomes more complex, and packets have to make two hops through the WSN, packet loss starts to take relevant values, although that only happens in clusters with assigned video MIP. Video MIP spends an amount of bandwidth between 10 and 50 times thee Audio MIP, so the probability of collision on the wireless network grows. When the number hops rises to three, significantly packet loss takes place even for audio MIPS with 64 Kbps and 192 Kbps. We can see that video delivery through three sensor nodes and 3500 Kbps bandwidth, equivalent to HDTV at 1080p, produces over 1% of packet loss. As a function of the codec used for video delivery, this percentage of packet loss can decrease drastically the quality of experience (QoE) of the end user.

The conclusion that we can extract from these results is that loss packet parameter can become a decisive QoS parameter that must be considered when the number of hops is equal to or bigger than three hops and the spent multimedia delivery bandwidth is high. MIP based cluster architecture can help by two ways: limiting the number of hops into a specific cluster and isolating heavy multimedia traffic into a separate cluster in order to improve QoS parameters of the other clusters.

6. Conclusion

Recently, the interest on WSNs has been increasing considerably, mainly because the nodes capacity to deliver huge amount of data efficiently in isolated geographic zones in harsh environments. The augmentation of the bandwidth in the new wireless technologies makes possible the use of multimedia sensors with new WSN usages. One of the main requisites in real time audio IP and video IP delivery is to meet the QoS requirements, but this is a difficult task in such types of networks. The way the WSN is organized and how sensor nodes communicate and create neighborhoods will be decisive to guarantee QoS.

In this work we propose and develop a new communication protocol that creates ad hoc clusters based on the multimedia flow features that are delivered inside the WSN. In order to achieve this goal, we have defined the MIP as a logical scheme that lets us manage the QoS requirements and the features of the sensor nodes building the cluster. The protocol allows the creation of clusters with a maximum diameter, which is adequate for each type of multimedia

Table 9: Statistical values and confidence interval for jitter as a function of the number of hops.

Hops	Parameters						
	N	μ (ms)	σ (ms)	Min (ms)	Max (ms)	Confidence interval (ms)	
1	1000	1.485	0.974	0	5	1.406	1.564
2	1000	3.118	1.336	1	12	3.009	3.227
3	1000	8.709	1.387	6	15	8.596	8.822

Table 10: Packet loss percentage.

Packet loss	AUDIO_64K	AUDIO_192K	VIDEO_1500K	VIDEO_3500K
1 Hop	0.00%	0.00%	0.00%	0.00%
2 Hops	0.00%	0.00%	0.04%	0.10%
3 Hops	0.19%	0.26%	0.51%	1.30%

flow and selects the most appropriate nodes, with enough resources, to be in the path of the multimedia delivery. We have detailed the protocol features, the designed messages, and the used variables. Moreover, we have explained the processes of the architecture, detailing how neighbour discovery, neighborhood creation, and multimedia delivery are taken place. Finally we have measured several cases in a test bench with real devices. We have proved that the protocol is able to achieve the adequate values of QoS parameters for different MIPs.

Our future research is focused on adding new MIP parameters such as sensor node mobility, energy consumption (like it has been added in [30]), and network stability [31]. Moreover, we are going to include the distribution capacity to the routing algorithm and add security mechanisms to guarantee the authenticity and integrity of the delivered multimedia data.

Conflict of Interests

The author(s) declare(s) that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work has been partially supported by the "Ministerio de Ciencia e Innovación," through the "Plan Nacional de I+D+i 2008–2011" in the "Subprograma de Proyectos de Investigación Fundamental," Project TEC2011-27516. This work has also been partially supported by the Instituto de Telecomunicações, Next Generation Networks and Applications Group (NetGNA), Portugal, by the Government of Russian Federation, Grant 074-U01, and by National Funding from the Fundação para a Ciência e a Tecnologia (FCT) through the PEst-OE/EEI/LA0008/2013 Project.

References

[1] D. Bri, M. Garcia, J. Lloret, and P. Dini, "Real deployments of wireless sensor networks," in *Proceedings of the 3rd International Conference on Sensor Technologies and Applications (SENSORCOMM '09)*, pp. 415–423, Athens, Greece, June 2009.

[2] M. Garcia, D. Bri, S. Sendra, and J. Lloret, "Practical deployments of wireless sensor networks: a survey," *International Journal on Advances in Networks and Services*, vol. 3, no. 1-2, pp. 170–185, 2010.

[3] L. Karim, A. Anpalagan, N. Nasser, and J. Almhana, "Sensor-based M2M agriculture monitoring systems for developing countries: state and challenges," *Network Protocols and Algorithms*, vol. 5, no. 3, pp. 68–86, 2013.

[4] J. Lloret, L. Shu, R. Lacuesta, and M. Chen, "User-oriented and service-oriented spontaneous ad hoc and sensor wireless networks," *Ad-Hoc and Sensor Wireless Networks*, vol. 14, no. 1-2, pp. 1–8, 2012.

[5] M. Edo, A. Canovas, M. Garcia, and J. Lloret, "Providing VoIP and IPTV Services in WLANs," in *Handbook of Research on Mobility and Computing: Evolving Technologies and Ubiquitous Impacts*, pp. 426–444, IGI Global, 2011.

[6] J. Mueller, T. Magedanz, and J. Fiedler, "NNodeTree: a scalable peer-to-peer live streaming overlay architecture for next-generation-networks," *Network Protocols and Algorithms*, vol. 1, no. 2, pp. 61–84, 2009.

[7] R. Diab, G. Chalhoub, and M. Misson, "Overview on multichannel communications in wireless sensor networks," *Network Protocols and Algorithms*, vol. 5, no. 3, pp. 112–135, 2013.

[8] L. Khoukhi and S. Cherkaoui, "Intelligent QoS management for multimedia services support in wireless mobile ad hoc networks," *Computer Networks*, vol. 54, no. 10, pp. 1692–1706, 2010.

[9] C. J. Barenco Abbas, L. J. García Villalba, and A. L. Sandoval Orozco, "A distributed QoS mechanism for ad hoc network," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 11, no. 1, pp. 25–30, 2012.

[10] Y. Chen, T. Farley, and N. Ye, "QoS requirements of network applications on the internet," *Information Knowledge Systems Management*, vol. 4, no. 1, pp. 55–76, 2004.

[11] T. Çevik and A. H. Zaim, "A multichannel cross-layer architecture for multimedia sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 457045, 11 pages, 2013.

[12] Z. Li, J. Bi, and S. Chen, "Traffic prediction-based fast rerouting algorithm for wireless multimedia sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 176293, 11 pages, 2013.

[13] J. Lloret, C. Palau, F. Boronat, and J. Tomas, "Improving networks using group-based topologies," *Computer Communications*, vol. 31, no. 14, pp. 3438–3450, 2008.

[14] J. Lloret, M. Garcia, J. Tomás, and F. Boronat, "GBP-WAHSN: a group-based protocol for large wireless ad hoc and sensor networks," *Journal of Computer Science and Technology*, vol. 23, no. 3, pp. 461–480, 2008.

[15] M. Lehsaini, H. Guyennet, and M. Feham, "An efficient cluster-based self-organisation algorithm for wireless sensor networks," *International Journal of Sensor Networks*, vol. 7, no. 1-2, pp. 85–94, 2010.

[16] J. Lloret, M. Garcia, D. Bri, and J. R. Diaz, "A cluster-based architecture to structure the topology of parallel wireless sensor networks," *Sensors*, vol. 9, no. 12, pp. 10513–10544, 2009.

[17] J. R. Diaz, J. Lloret, J. M. Jimenez, and S. Sendra, "MWAHCA: a multimedia wireless Ad Hoc cluster architecture," *The Scientific World Journal*, vol. 2014, Article ID 913046, 14 pages, 2014.

[18] D. Wei and H. Anthony Chan, "Clustering ad hoc networks: Schemes and classifications," in *Proceedings of the 3rd Annual IEEE Communications Society on Sensor and Ad hoc Communications and Networks (SECON '06)*, pp. 920–926, Reston, Va, USA, September 2006.

[19] J. Yu and P. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Communication Surveys*, vol. 7, no. 1, pp. 32–48, 2005.

[20] R. Agarwal and M. Motwani, "Survey of clustering algorithms for MANET," *International Journal on Computer Science and Engineering*, vol. 1, no. 2, pp. 98–104, 2009.

[21] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*, vol. 30, no. 14-15, pp. 2826–2841, 2007.

[22] O. Boyinbode, H. Le, and M. Takizawa, "A survey on clustering algorithms for wireless sensor networks," *International Journal of Space-Based and Situated Computing*, vol. 1, no. 2-3, pp. 130–136, 2011.

[23] L. Ramachandran, M. Kapoor, A. Sarkar, and A. Aggarwal, "Clustering algorithms for wireless ad hoc networks," in *Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM '00)*, pp. 54–63, August 2000.

[24] M. Chatterjee, S. K. Das, and D. Turgut, "WCA: a weighted clustering algorithm for mobile Ad Hoc networks," *Cluster Computing*, vol. 5, no. 2, pp. 193–204, 2002.

[25] M. Kavitha and K. Karthikeyan, "Comparison of data centric protocols for WSN and energy enhanced M-Spin (EEM-SPIN)," *International Journal of Engineering Research & Technology*, vol. 1, no. 9, 2012.

[26] Y.-M. Huang, M.-Y. Hsieh, and M.-S. Wang, "Reliable transmission of multimedia streaming using a connection prediction scheme in cluster-based ad hoc networks," *Computer Communications*, vol. 30, no. 2, pp. 440–452, 2007.

[27] S. Tang and W. Li, "QoS supporting and optimal energy allocation for a cluster based wireless sensor network," *Computer Communications*, vol. 29, no. 13-14, pp. 2569–2577, 2006.

[28] D. Rosário, R. Costa, H. Paraense et al., "A hierarchical multihop multimedia routing protocol for wireless multimedia sensor networks," *Network Protocols and Algorithms*, vol. 4, no. 4, pp. 44–64, 2012.

[29] J. R. Diaz, J. Lloret, J. M. Jiménez, and M. Hammoumi, "A new multimedia-oriented architecture and protocol for wireless Ad Hoc networks," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 16, no. 1, 2014.

[30] P. Spachos, P. Chatzimisios, and D. Hatzinakos, "Energy aware opportunistic routing in wireless sensor networks," in *Proceedings of the Global Communications Conference (GLOBECOM '12)*, pp. 405–409, December 2012.

[31] N. Meghanathan and P. Mumford, "Centralized and distributed algorithms for stability-based data gathering in mobile sensor networks," *Network Protocols and Algorithms*, vol. 5, no. 4, pp. 84–116, 2013.