Garcia-Sanchez et al. EURASIP Journal on Wireless Communications and Networking 2012, 2012:198
http://jis.eurasipjournals.com/content/2012/1/198

EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

On the synchronization of IEEE 802.15.5 wireless mesh sensor networks: Shortcomings and improvements

Antonio-Javier Garcia-Sanchez*, Felipe Garcia-Sanchez, David Rodenas-Herraiz and Joan Garcia-Haro

Abstract

As part of the recent IEEE 802.15.5 wireless mesh sensor networks (WMSN) standard, Synchronous Energy Saving (SES) is planned to provide energy savings to scheduled communications with strict temporal requirements that, a priori, facilitate the development of delay-sensitive applications. It is accomplished by means of different mechanisms, among which we highlight a straightforward synchronization process. However, the SES synchronization scheme introduces variable delays in the dissemination of information and reduces the lifetime of the nodes and the entire network significantly, thus limiting the full exploitation of SES. This article presents a new synchronization approach, that we call *High-Performance Synchronization Algorithm for wireless mesh sensor networks* (HIPESYN), which is adapted to the IEEE 802.15.5 standard for synchronous communications. HIPESYN supports intensive bandwidth applications in a much better way than with the original design. The proposed algorithm is also thoroughly evaluated and its results carefully discussed.

Keywords: Wireless mesh sensor networks, Synchronization, Energy saving

1. Introduction

Wireless Sensor Networks (WSNs) are created by the interconnection of low-cost communication devices (nodes) that report information acquired from different sensors (temperature, humidity, pressure, etc.) at low bitrates. In general, nodes are resource constrained regarding memory, computing capacity, and especially energy. In a WSN, the power consumption is a crucial issue since it determines the time period during which the network operates in accordance with the requirements defined by the user. This time period is denoted as network lifetime. In addition, the nodes are usually placed at isolated or difficult access locations, where supply of external energy is often not possible. These facts imply the use of low-consumption electronic components to save as much energy as possible. However, this solution is insufficient for applications where it is necessary to deploy hundreds or thousands of nodes to monitor a large area or to transmit sensor data to end-users separated by several kilometers. In this context, to obtain an arbitrarily long WSN lifetime, the collaboration among nodes is essential. Therefore, one of the main objectives of this technology is the ability of all network nodes to operate in coordination, allowing an efficient power consumption of each device while carrying out the required tasks successfully.

Wireless Mesh Sensor Networks (WMSNs) are specialized in providing mesh capabilities through collaborative sensor nodes that not only sense the environment and forward monitoring data, but are also able to perform the necessary routing tasks to deliver the acquired information to the final destination. This means that all nodes belonging to the WMSN must fulfill the typical functionalities of a mesh topology such as scalability, multi-hop, selection of the best path to destination, robustness to changes, low power consumption, reliability, asynchronous/synchronous communications, etc. (Lee et al. 2006), simultaneously coping with their particular limitation in resources. In this framework, many studies such as (Kominami et al. 2010; Hussey et al. March 2010; Liu et al. 2010) or even commercial mesh implementations (Zigbee (ZigBeeTM Alliance), WirelessHart (WirelessHart® Technology), IETF

* Correspondence: antoniojavier.garcia@upct.es
Department of Information and Communication Technologies, Universidad Politécnica de Cartagena (UPCT), Campus Muralla del Mar, E-30202, Cartagena, Spain

6LoWPAN (Hui & Thubert), etc.) can be found. Unfortunately, none of them has efficiently and reliably solved many of the aforementioned functionalities of WMSN.

To this end, the IEEE has recently released the IEEE 802.15.5 standard (IEEE Computer Society) in order to provide, in a single recommendation, all the distinctive features of the WMSNs, in particular, an efficient multi-hop scheme and an effective energy-saving procedure. The low-rate version of this standard is fully compatible with IEEE 802.15.4 (IEEE Computer Society) (dominant standard in the WSN market). IEEE 802.15.4 consists of a very small stack composed of the physical and Medium Access Control (MAC) layers, a simple design regarding interoperability, a long battery life scheme, and a low rate (250 kbps at 2.4 Ghz) transmission. In fact, this standard is also denoted by the acronym LR-WPAN (Low Rate-Wireless Personal Area Network). Unfortunately, IEEE 802.15.4 does not specify how to support multi-hop routing abilities, delegating the design and development of them to the upper layers. The objective of IEEE 802.15.5 is therefore to solve the limitations of IEEE 802.15.4, developing basic mesh networking functions and primitives. With this aim, IEEE 802.15.5 provides features such as node discovery, multicast, reliable broadcast, synchronized and unsynchronized operations, power saving (ON/OFF scheduling strategy), and route tracing, thus taking into account the strict constraints of the WSN devices. The result is a recommendation, known as the LR-WPAN mesh standard, which enables a straightforward migration from IEEE 802.15.4 to mesh networks, which in addition facilitates the development of applications on top, following topologies with a high number of nodes without incurring extra-complexity and cost.

In this sense, the studies in (Lee et al. 2010a; Lee et al. 2010b) give technical details about the design of IEEE 802.15.5 and present a methodical testbed for multicasting and power saving in asynchronous communications, denoted by the standard as Asynchronous Energy Saving (AES). It tackles the communications in a mesh topology by using a contention-based algorithm, where each station transmits data only when the physical medium is idle. As a result, the information may reach its destination but suffering high delay variability and achieving low transmission rates. This behavior is usually valid for sensor monitoring (temperature, pressure, etc.) applications. In addition, neighbor nodes may transmit simultaneously provoking message collisions and later retransmissions, further increasing delay and power consumption, and consequently decreasing the nodes and network lifetime. However, when the dissemination of information requires stricter timing, bitrate and low-power requirements than those provided by the AES solution (e.g., delay-sensitive applications involving video or audio transmission (Koenen)), an alternative method must be regarded. The synchronous communication mechanism called Synchronous Energy Saving (SES) is the part of the IEEE 802.15.5 standard which, a priori, is designed for this type of applications. However, as far as the authors know, it is not evaluated in the open literature.

In this respect, the *first contribution* of this research is aimed at studying and evaluating the performance of SES by means of computer simulation. To satisfy this goal, we select input design parameters such as the number of regions into which the mesh topology is divided, the number of hops per region, the duty/sleep cycle periods, and the data rate. Using them we will show the real potential of SES and its impact on several important metrics such as the throughput, latency, jitter, message delivery ratio, and energy consumed. Moreover, as an added-value to this study, we will discuss and offer the most appropriate ranges for the input parameters to achieve the best performance results in accordance with the requirements of the end-users.

Our evaluation study follows the specifications described by the IEEE 802.15.5 standard. This study reveals an important shortcoming of the SES process, in particular, regarding its synchronization method. In a mesh scenario, where the information flows are addressed from any given network node to another (both usually out of direct coverage, that is, in different network regions), an appropriate synchronization is achieved when it is not interfered by the data forwarding task. In this regard, we show that the network performance of IEEE 802.15.5 deteriorates noticeably because, in a same region, the synchronization process prevails over the reception of data (from a neighbor region). The IEEE 802.15.5 standard addresses this situation through border nodes placed between two adjacent regions. Border nodes are in charge of retransmitting the sensor data to the sink. To this purpose, they must wait for the finalization of the synchronization process what leads to introduce delays in the data transmission and to not assuring information delivery in a timely manner. Furthermore, these delays imply a less number of messages disseminated and an extension of the ON period on each node involved in the communication, thus increasing the energy consumed. All these drawbacks clearly jeopardize the implementation of different WMSN applications, in particular, the delay-sensitive ones.

To relieve this effect, and after an intensive study of WSN synchronization protocols found in the open literature, our *second contribution* refers to the proposal of a new synchronization algorithm. As it will be shown, this solution adds to the synchronization method very valuable properties, namely energy efficiency and low-computation. We denote this synchronization mechanism as

***Hi**gh-**Pe**rformance **Syn**chronization Algorithm for wireless mesh sensor networks* (HIPESYN). In HIPESYN, nodes do not synchronize in periodic time intervals. Quite the opposite, synchronization is adjusted to the real operation of each node fulfilling the synchronization process, for instance, when a node has to send information. We show how to integrate the proposed algorithm in the IEEE 802.15.5 standard, thus assuring an accurate synchronization between regions and improving the communications performance in a mesh topology. Finally, an analytical and simulation study shows how our solution provides the system with higher flexibility than the SES mode, as well as with a noticeable improvement of the network lifetime. Unlike SES, the scheme proposed would enable the development of WMSN multimedia services (Akyildiz et al. 2007) such as those incorporating video/audio, paving the way for a plethora of new WSN applications.

The entire system is described in detail in the remaining part of this article, which is organized as follows. Section 2 summarizes the related work found in the open literature about synchronization protocols suited for the requirements of the WMSN topologies. Section 3 outlines the IEEE 802.15.5 standard. Then, in Section 4, SES is explained and its performance meticulously evaluated, discussing the results obtained. Section 5 details the HIPESYN synchronization mechanism proposed with a special focus on performance improvement (analysis and simulation) in comparison with the standard SES. Finally, Section 6 concludes and points to future directions of this research study.

2. Related study

In a WSN, each node has its own notion of time based on its internal clock. Individual clocks may tick at slightly different rates, thus resulting in independent clock's drift and a global (network) lack of synchronization. The consequence may be a drift of seconds per day, accumulating significant errors over time. Obviously, this may produce serious inconveniences to applications that depend on a strict synchronized global notion of time.

This issue is addressed in the scientific literature, where multiple synchronization protocols, designed specifically for WSN, can be found. Given the scope of this study, focused on LR-WPAN mesh, we look at the most relevant synchronization protocols which also provide mesh features such as *high-density network nodes, scalability, multi-hop, fault-tolerance, and low-computational complexity*. It should be noted that well-known WSN algorithms as (Elson et al. 2002; Palchaudhuri et al. 2003; Römer 2001; Ping 2003) exhibit severe drawbacks when they run in mesh network topologies, because they do not offer some of the aforementioned features.

In order to accomplish all these features, other mechanisms are more suitable for WMSN. The study in (Sichitiu & Veerarittiphan 2003) describes a low power consumption scheme; thanks to the simplicity of the synchronization algorithm proposed. The goal is to include within the data messages information about synchronization. Given the time variability associated with the data messages delivery, a high delay may be suffered in the global synchronization, penalizing it due to precision inaccuracies among logical clock readings of the wireless network nodes. To avoid this, other protocols such as (Mock 2000; Su & Akyildiz 2005) use synchronization messages. In this case, a master node, defined by (Mock 2000), manages the network; thanks to a greater processing capacity and memory than the remaining nodes. This node starts the synchronization process, broadcasting a pair of messages to several slave nodes (usually nodes with low computing and memory capacity) in each synchronization round. This procedure entails the control of the synchronization precision, but these synchronization messages increase the energy consumed by node. Following this line, master nodes in (Su & Akyildiz 2005) broadcast the timing information to their neighbors, which use this value as time reference. The neighbor nodes become leader nodes, that is, nodes with a greater level of responsibility to further broadcast the timing information to their respective neighbors. Master nodes repeat this process in periods exclusively devoted for this purpose. This technique reduces the number of synchronization messages in comparison with (Mock 2000). However, the study in (Su & Akyildiz 2005) executes several complex processes requiring a large amount of memory and computational capacity. The investigations in (Zhang & Deng 2005; Gelyan et al. 2007; Jabbarifar et al. 2010) present techniques which complement the forwarding of synchronization messages, and the slave nodes maintain synchronization and calculate the time drift with their master in a probabilistic manner. This method reduces power consumption in comparison with the previous algorithms, because it decreases the number of network messages required. However, it increases the computational complexity and introduces precision variability because of the probabilistic nature of this mechanism. Other recent proposals such as (Nieminen et al. 2011) exploit the idea of using multiple frequency channels for carrying out the synchronization tasks. This avoids the precision inaccuracies caused by interfering the data transmission with the synchronization process. Conversely, it penalizes complexity and latency due to the association process that a node has to follow for the channel switching.

In the same context, other synchronization algorithms as (Koubâa et al. 2008; Ganeriwal et al. 2003) have been

tested and validated in a hierarchical/cluster-tree topology. In this configuration, source and destination nodes can be separated by long distances; thanks to a dedicated path of sensor nodes forming parent–child links. This solution may result in communication bottlenecks when a sensor node fails. Koubâa et al. (Koubâa et al. 2008) describe how the Task Group 15.4b specifies two different synchronization algorithms for IEEE 802.15.4 networks: (i) the time division approach and the (ii) beacon-only period approach. In the first case, each parent of the cluster (denoted as coordinator) schedules the transmission of its synchronization message to avoid collisions during its ON state with the one of any neighbor or neighbor's parent in its area of coverage. In the second scheme, a specific interval is reserved only for the synchronization tasks in such a way that each parent sends its synchronization message in a contention-free manner. Zigbee (ZigBeeTM Alliance) is a well-known implementation of cluster-tree topologies over IEEE 802.15.4, which carries out the modification of algorithm (i). In Zigbee, a node continuously scans the physical medium to learn about the time taken by the synchronization messages of the neighborhood. Thereby, if this node becomes a parent, it can select the time interval to transmit its synchronization messages, so that its ON state does not overlap with any other ON state of any neighboring device/parent. However, this operation exhibits two disadvantages: (i) deficient synchronization precision, because a new parent is only aware of a single synchronization message for each neighbor device, just adjusting the deviation between clocks but no other parameters as the drift and (ii) the reduction of the network lifetime motivated by the continuous scanning of the physical medium. The latter aspect is dealt and enhanced in the studies in (Lopez-Gomez & Tejero-Calado 2009; Li et al. 2012). On the other hand, Ganeriwal et al. (Ganeriwal et al. 2003) present a sender–receiver scheme denoted as Timing-sync Protocol for Sensor Networks (TPSN). The TPSN is a simple/light protocol regarding computational complexity where the sender node calculates the offset between two nodes and the delays due to the transmission of the messages across the physical medium, obtaining a highly accurate synchronization with the receiver node (average error of 16.9 μs). The authors in (Ganeriwal et al. 2003) remark its low number of synchronization messages, because synchronization is only carried out when both nodes need to exchange data (which also impacts positively on power consumption). Additionally, the synchronization error can be estimated because its value is bounded. These favorable results are obtained owing to the adjustment of the sender–receiver scheme to the tree topology operation where a child node can only send information to its parent. Therefore, a child is always synchronized to its unique parent. Unfortunately, its performance is not evaluated for a mesh topology where a node must be synchronized with all their neighbors to dynamically route a data message.

In a recent article (Cho et al. 2011), a synchronization method was presented and evaluated for a grid topology, a particular scenario of the mesh arrangement. This algorithm improves the Light-Weight Time Synchronization (LWTS) schemes, removing some of the duplicated broadcast messages occurring in the flooding phases of the topology construction, children discovery, and synchronization operation. In (Cho et al. 2011), TPSN and the new LWTS algorithm are compared using Carrier Sense Multiple Access (CSMA) for the MAC. TPSN follows the rules specified in (Ganeriwal et al. 2003), focused on tree-based scenarios. Under these circumstances (different from our study focused on the IEEE 802.15.5 standard), the method in (Cho et al. 2011) slightly improves the synchronization accuracy under ideal conditions at the expense of increasing the computational complexity. Furthermore, this algorithm does not consider the recovery of synchronization messages in case of loss, therefore worsening the synchronization precision on real implementations.

All these investigations and proposals make us select the sender–receiver scheme as a reference to propose and design a new synchronization algorithm. This algorithm must completely ensure all the aforementioned *mesh features*, in particular high synchronization precision, low energy and cost, and reduced computational effort. Besides that, it must be fully conformed to the IEEE 802.15.5 mesh standard and in turn, enhance the performance in mesh topologies noticeably. For all these reasons, in Sections 3 and 4, we will describe the IEEE 802.15.5 standard and its energy-constrained methods, and later, in Section 5, we will explain the proposed synchronization algorithm.

3. IEEE 802.15.5 description and operational overview

IEEE 802.15.5 is the first global standard for WPAN (IEEE Computer Society); it provides multi-hop mesh functions with the goal of increasing the network coverage without jeopardizing the energy consumption of each device. Given the huge scope for developing applications under this standard, it is divided into two parts: low-rate (LR) and high-rate (HR), both taking advantage of the MAC and Physical layers of IEEE 802.15.4 (LR) and IEEE 802.15.3 (HR) standards, respectively. In this article, our research focuses on the low-rate part, because it is conceived for the provisioning of WSN mesh capabilities.

Depending on the type of application, a mesh topology can be formed by a variable number of devices

collaborating among them; from a few nodes for the monitoring of a particular scenario (e.g., a floor of a building) to a considerable number of devices which could be deployed in large-scale sensor networks (e.g., agriculture, industry, military, etc.). In the latter, the scalability of the LR-WPAN mesh together with the constrained resources of the sensor nodes become a crucial design issue that has an impact, among others, on the addressing scheme and the corresponding routing protocol required to alleviate and cope with these constraints.

To this aim, at the discovery phase, each node learns which nodes are its neighbors, and how far it is from the root. To this purpose, the wireless nodes generate their own routing tables by means of a twofold procedure: (i) an optimized tree-based addressing scheme which reduces the size of the address field in the message header, and (ii) a local link establishment among neighbor nodes, resulting in the final mesh topology. The first procedure requires each node to check its neighborhood and to select a device to join, obtaining a tree topology where, as final result, a node can only communicate with a single parent and its children (Figure 1a). The traditional tree schemes employ 64 bits for addressing, increasing the message header but at the expense of decreasing the payload [the message size is 127 bytes in the physical (PHY) layer, which is the maximum value according to the IEEE 802.15.4 standard]. To overcome this drawback, each node of the network reports its parent about the number of children associated to it, so that this information arrives at the root of the tree in several hops (Figure 1b). The root is in charge of assigning blocks of consecutive 16-bit logical addresses to each branch below it, taking into account the number of children, therefore reducing the message header size in comparison with the traditional tree-based scheme (Figure 1c). On the other hand, the robustness of the routing process is achieved by means of additional local links which solve the main shortcomings of the tree scenario (Figure 1d). Now, messages can also be routed by alternating paths different from the original and rigid tree links.

Once the process of creating the entire mesh topology and routing tables of each node is finished, any node can start a communication, transmitting data beyond its direct coverage to any other destination node (Figure 1d). For this purpose, a path is established between source and destination using a multi-hop routing algorithm called Topology-guided Distributed Link State (TDLS), which selects the most suitable device to be involved from the set of nodes forming the mesh network. Nodes in the communication path are chosen according to the minimum number of hops to the sink. In the usual case of several optimum paths, the messages are uniformly distributed among them to satisfy that all messages do not go through the same intermediate nodes, satisfying load balance. These nodes perform the routing task, forcing their transceivers to be always ON. This fact allows listening to the physical medium for the reception of messages or their forwarding to the next routing node in the path. However, a transceiver permanently in ON mode has a clear negative influence on the power consumption of the nodes and, as a consequence, on the network lifetime. Note that the energy saving is one of the most desirable features in a WSN.

To tackle this drawback, the low-rate IEEE 802.15.5 standard provides two solutions named AES and SES, both focused on reducing the duty-cycle of the transceivers, so as to switch them for as long as possible to the sleep mode. In particular, AES implements the power consumption reduction in dynamic mesh networks. To do that, each node defines a fixed interval initiated by a *broadcasting notification*, and then it remains listening to messages for a time configured by the user during the active period (time interval within which the node operates), and it switches later to the power saving state (sleep mode). Thereby, when a node wants to transmit a message, its transceiver remains in the receiving mode, waiting for the *broadcasting notification* of the next node indicated by its routing table or by the end-destination one. Once the *broadcasting notification* is received, the node forwards the data to the next node in the route or to the end-destination within its active period, avoiding collisions between messages from other nodes thanks to the Carrier Sense Multiple Access-Collision Avoidance (CSMA-CA) contention-based algorithm. On the other hand, the SES mechanism is devoted to decreasing power consumption and the end-to-end delay mainly in static networks where nodes stay in their placements all the time or show reduced mobility with the aim of avoiding connectivity loss with the neighbors. The research presented in this article is concentrated on SES, which is described with more detail in the next section.

4. SES

SES uses a strict schedule of tasks for all network devices which are synchronized to a unique node, the mesh coordinator, mainly on static networks. The mesh coordinator is the head device of the tree topology and it is in charge of starting the synchronization process by sending a message with its clock time information twice. Each node of the network, child of the mesh coordinator, stores the clock time of the first message sent by the coordinator and a timestamp (temporal label included in the message header) with its own clock time. When the second message arrives at the children nodes, they calculate the difference between both coordinator's clock times and the difference between their own

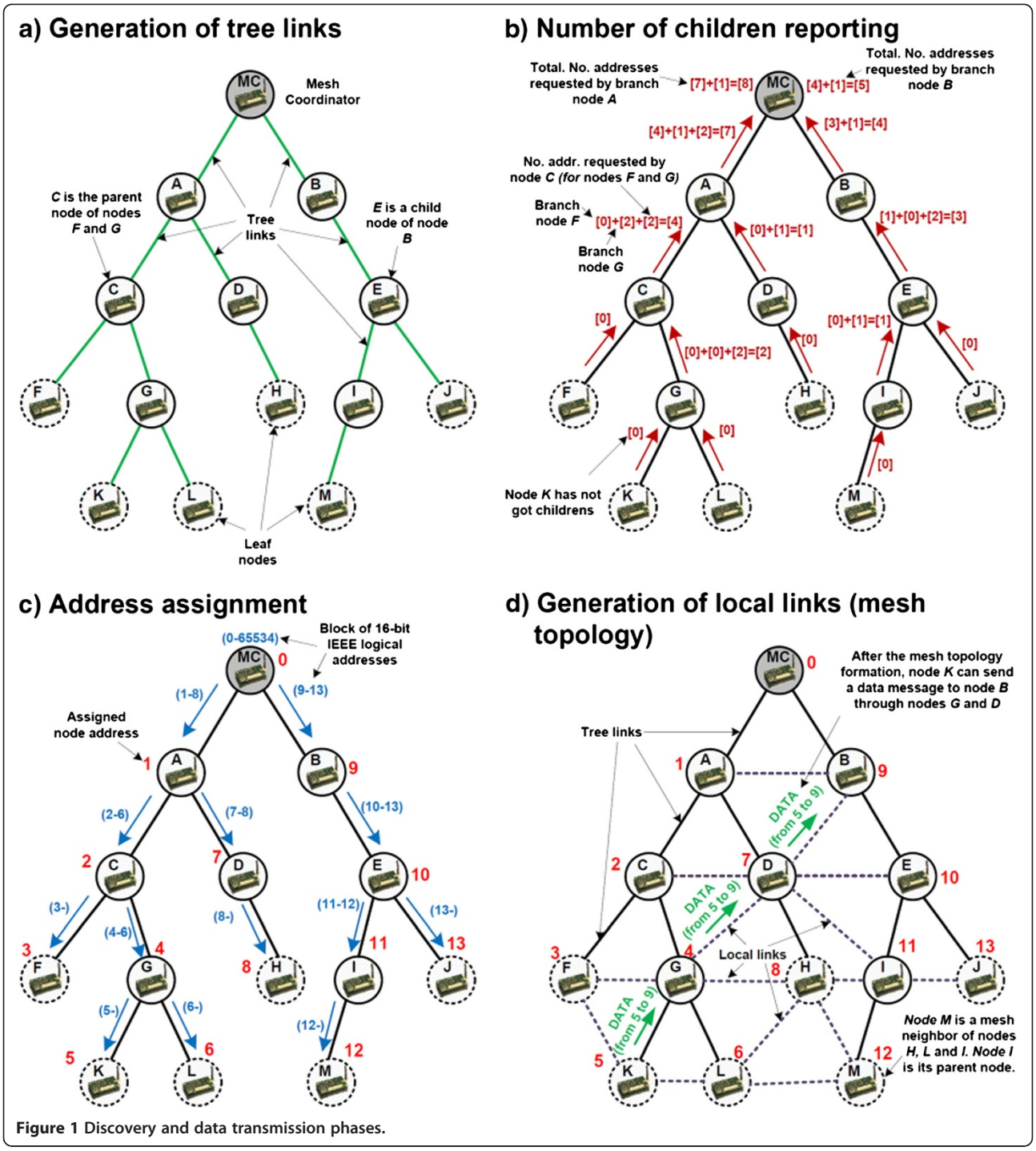

Figure 1 Discovery and data transmission phases.

current clock and the timestamp previously stored, taking advantage of the low latency between two consecutive synchronization messages. The difference between both resulting values is the drift between each one of the children and the coordinator. If all nodes on the network are not reached by the message of the coordinator in one-hop, these children retransmit the coordinator's clock times in multiple hops to the rest of network nodes, spreading the synchronization along the logical tree.

At the same time as the first synchronization action is spreading, the SES mechanism divides the entire mesh network into multiple fixed regions, so that some nodes become parents of each one of these regions. Placed between two regions, the parent nodes are border devices in charge of guaranteeing the global synchronization of

the entire mesh network. To achieve this purpose, parent nodes are able to perform a twofold task simultaneously: (i) to be synchronized with the up-region and (ii) to be responsible for synchronizing all the children nodes of its own region in one or multiple-hops. This is the reason why these special nodes (parents) are denoted as *Region Synchronizers*. The goal is to share the network-wide synchronization responsibility between the mesh coordinator and the synchronizers of each region for future synchronization actions. In this context, each one of these actions is triggered by the mesh coordinator, which synchronizes all the nodes of the first region by means of a *synchronization request* message, ending its operation when the *synchronization reply* message arrives from the most remote node of the first region. Once the time assigned by the mesh coordinator for the synchronization of the first region expires, synchronizers belonging to the second region start the same procedure again and so on with the rest of regions.

Although the mesh coordinator and the regional parents govern the synchronization scheme together, the mesh coordinator is responsible for setting the duration of an entire cycle for the synchronization process, denoted as *synchronization interval* (SI). SI is a time period determined by parameters such as the size of the network and/or the number of hops by region (called the standard *Synchronization Region*—SR). Its value is a multiple of fixed intervals (*wakeup interval*—WI) as shown in Figure 2. Furthermore, the same figure shows how SES defines a strict time period per each region and SI, denoted as *Synchronization Duration* (SD) for exclusive use of synchronization tasks, during which nodes cannot transmit information. Note that in each SI the mesh coordinator starts a new synchronization process, establishing regions and selecting the most appropriate parents, which in addition must synchronize their respective children within an SD period of time. Many incidences may be solved through this operation. For instance, children nodes may be orphaned due to the battery depletion of their parent node. However, in the next SI, this problem is solved by simply selecting a new synchronizer node, therefore improving the robustness of the network.

According to SES, data transmission starts after the end of the SD. During the data transmission period, any node of the mesh network can establish a communication under strict delay requirements. To this end, SES defines the *WI*, a time interval whose length is calculated as a function of the *Wakeup Order (WO)* and the *meshBaseActiveDuration* (value fixed to 5 ms according to the standard) parameters. WI consists of two parts (Figure 2): *Active Duration* and *Inactive Duration*. Within the *active duration* all the nodes trying to transmit information compete for accessing to the physical medium according to the CSMA-CA mechanism. Following the same reasoning of the WI, the duration of the *active period* is calculated as a function of the *Active Order* (AO) and *meshBaseActiveDuration* parameters. In contrast, the *inactive period* was designed, a priori, to indicate the time interval during which nodes are in the sleeping mode (IEEE Computer Society). However, SES also uses the inactive period for transmitting data with the goal of minimizing the end-to-end delay of transmitted messages. To this aim, SES divides the *inactive period* into fixed slots which are reserved by the competing nodes during the *active duration*. Each time that a node has to forward a data message to its neighbor synchronizer, the device has to reserve a slot in the *active period* by means of *reservation request/reply* messages, in which the transmission of the message will take place. When the message transmission ends, both nodes switch to the sleep mode. If a node reaches its region synchronizer using several hops (multi-hop), the maximum number of slots reserved within an *inactive period* should match the number of hops. The transmission of a message is done sequentially, that is, each message received by a node in a slot is forwarded to the following node along the path in the next consecutive slot. This process, denoted as *reservation-based method* is repeated until the next region synchronizer or the end-destination is reached.

4.1 Performance evaluation of SES

To evaluate the performance of the SES mechanism, which, to the best of authors' knowledge, has not been assessed yet, we conducted extensive computer simulations using the ns-2 (Network Simulator 2) (USC Information Sciences Institute) simulation platform. We have selected the ns-2 framework because it is probably the most adopted simulation tool among the scientific community. In addition, it has carefully been verified and validated for different technologies, communication protocols, and network topologies.

Observing this framework in detail, we discover the possibility of running simulations using the PHY primitives of the IEEE 802.15.4 standard (Zheng & Lee 2006; Garcia-Sanchez et al. 2011). The TDLS algorithm developed by Zheng and Lee (Zheng & Lee 2007) is also implemented in the ns-2 simulator. It should be noted that the latter is the first approach to the LR-WPAN recommendation addressed to the mesh topology creation and routing issues, guaranteeing the scalability of the system and the robustness to changes. However, SES is not implemented. Therefore, over the PHY layer of IEEE 802.15.4 as well as the TDLS algorithm, our goal is to design and program a complete simulation environment for SES, including the reservation-based transmission method and the complete synchronization mechanism.

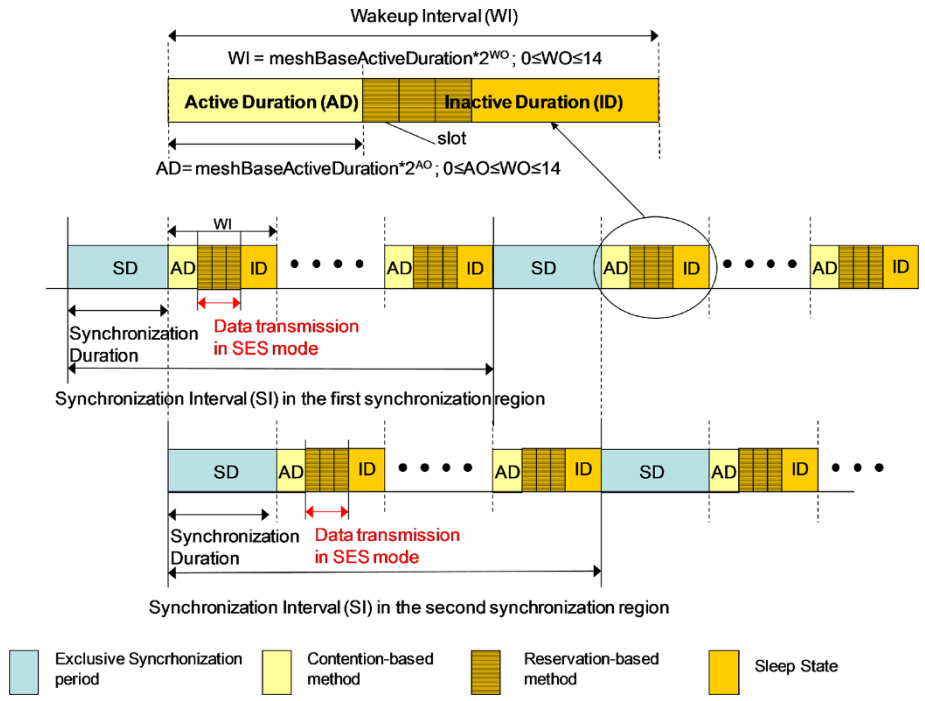

Figure 2 SES scheme.

To this aim, we have set a scenario consisting of a network of 625 (25 × 25) nodes arranged in a regular grid layout (Chen et al. 2008), where the distance between two consecutive neighbors at the same column or row is 30 m. In this topology, the mesh coordinator is placed at the grid left corner (node 0). Constant bit rate traffic is used with a transmission rate of 1 message each 0.08 s at 2.4 GHz for the ISM band (250 kbps). All these values are significantly worse than those suggested by Lee et al. (Lee et al. 2010a), being the only work until now that evaluates the performance of IEEE 802.15.5 for the AES mode. In addition, each experiment runs a set of simulations with different random seeds.

To obtain realistic simulation results in this scenario, two crucial concerns have been taken into account: (i) the desynchronization between nodes and (ii) the intrinsic drift of the nodes' clock. Both parameters take values based on the study in (Mock 2000). The desynchronization issue is reproduced by introducing a variable synchronization error with values ranging from 0.8 to 2.1 ms. On the other hand, the drift is set to a fixed value of 40 μs/s of simulation, which is obtained from the commercial specifications of a MicaZ mote (MicaZ Datasheet). However, other considerations are determined apart from the IEEE 802.15.5 standard, but they are out of the scope of this study. For instance, issues such as message collisions occurring in the access to the transmission medium due to hidden nodes are resolved; thanks to the appropriate layout of directive antennas or the use of additional time multiplexing techniques. In any case, the entire simulation framework with all the aforementioned features has been uploaded on the following website *http://www.ait.upct.es/~ajgarcia/IEEE802155/files* along with the corresponding user's manuals in order to be executed by the interested audience, or to fully reproduce our simulation experiments.

As already mentioned in Section 1, in a WMSN, data dissemination may take place in any direction when both source and sink are not in coverage. It means that our test can be carried out from any given network node delivering data to another. Accordingly, we have simulated a realistic case, where a node continuously acquires sensing data and transmits them to the mesh coordinator, acting as destination. The worst case is the one where the separation between source and destination nodes is maximum, thus allowing to study the effect of messages traveling through different regions in a multihop communication fashion. This is the reason why we have placed the source node at the right corner of the grid (node 624).

To assess the performance of SES, we have chosen different metrics of interest, which are usually observed in studies of reference such as (Akyildiz et al. 2005; Basaran 2009) with the objective of analyzing the behavior/quality of the end-to-end communications. The simulation results shown in Figures 3, 4, and 5 are throughput (bandwidth utilization), latency (average delay), jitter (average deviation from the messages latency), message

delivery ratio (percentage of messages delivered at sink successfully), and a routing-node lifetime achieved from 40 independent simulations (95% confidence intervals). It should be mentioned that, due to the small values of these confidence intervals (in the maximum range of 0.5% of the result values), they are not shown in figures, to clarify the representation of the achieved results. These metrics are obtained by appropriately setting the characteristic parameters of the SES process, such as the WO, AO, SR, and SI. For instance, the WO and the AO parameters have a direct impact on the performance metrics, since both parameters define the duration of the active and inactive periods, thus comprising a time schedule, so that a station selects the way of delivering its data: *contention-based method* or *reservation-based method*. On the other hand, the SR and the SI are parameters imposed by the synchronization scheme itself. Both contribute to the network synchronization, and its selection is non-trivial because, for instance, a tight synchronization adversely affects the network metrics, resulting in a worse performance, as will be discussed in the following section.

4.1.1. Throughput, latency, jitter, and delivery ratio metrics

Regarding the throughput, latency, jitter, and message delivery ratio performance metrics, Figure 3 shows their

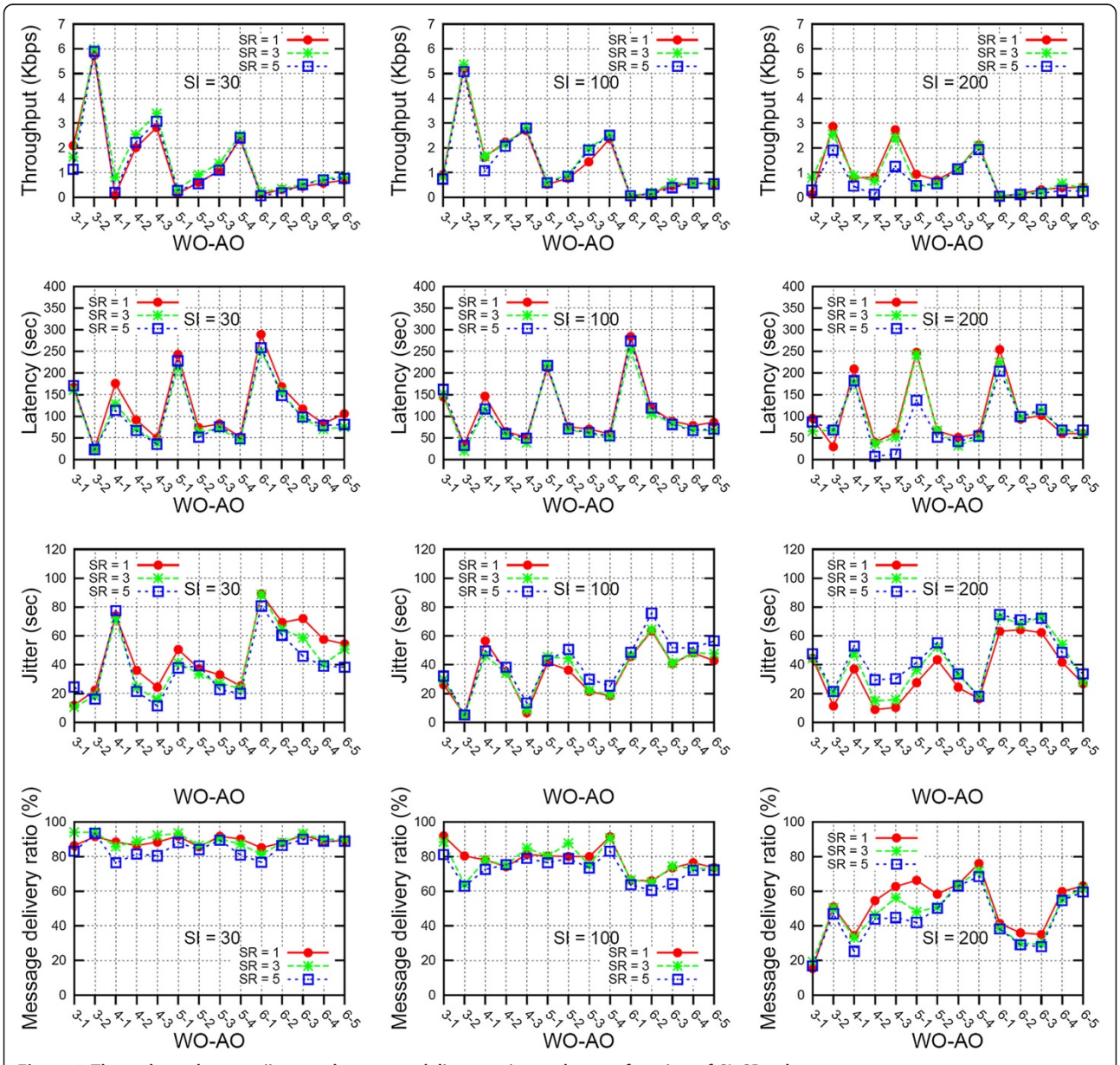

Figure 3 Throughput, latency, jitter and message delivery ratio results as a function of SI–SR values.

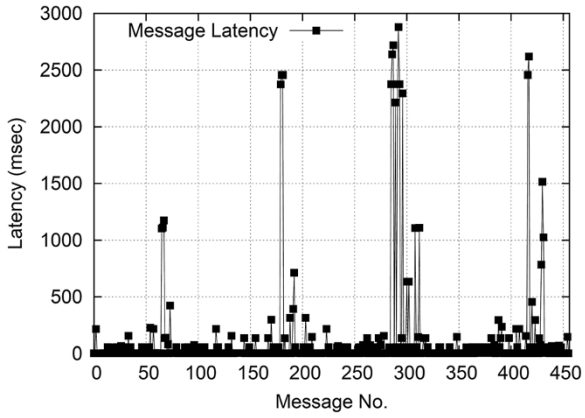

Figure 4 Latency experienced for a routing node with parameters WO = 4, AO = 3, SI = 30, and SR = 3.

evolution as a function of two main issues: the length of the active/inactive periods and the synchronization error. From their individual or joint tuning, the achieved performance in the mesh network fluctuates significantly.

Concerning the duration of *active* and *inactive periods* and observing Figure 3, we realize that at high values of WO the latency in general increases. This is because, under this consideration, a very long *inactive period* may appear which forces the nodes to switch to the sleep mode. Therefore, they have to wait longer to transmit their data and consequently the throughput is sharply reduced. Figure 3 also shows particular cases where the *active duration* (defined by the AO parameter) takes special significance in the transmission bitrate, delimiting the maximum hop distance for delivering data. When the *active duration* is a short interval, the *reservation request* message is forwarded covering few hops, and therefore few nodes. If the *inactive duration* is long enough (represented by a high WO value), only a few number of time slots are reserved and used for delivering the data message, because SES forces the *inactive period* to guarantee the same number of time slots as nodes that take part in the *reservation* process of the *active period* for a same WI. Again, this is why the rest of the *inactive period* is not used for data transmission, since the nodes switch to the power-save mode (sleep state). However, if the *active duration* is a long interval, the *reservation request* message may be forwarded over a bigger number of hops reaching more nodes as well. In this situation, it is necessary to carefully design the size of the *inactive period*, assuring the same number of slots as that of participating nodes in the reservation process. This is often possible only by increasing the value of WO, thus extending the inactive period. Otherwise, several reservation messages delivered within the active period would not have assigned any reservation slot, provoking the repetition of reservation processes in the following WI. Therefore, an appropriate adjustment of the active and inactive periods favors good throughput and latency results.

The metrics also vary according to the synchronization error required into the mesh network. A high value for the synchronization error may entail that a node is

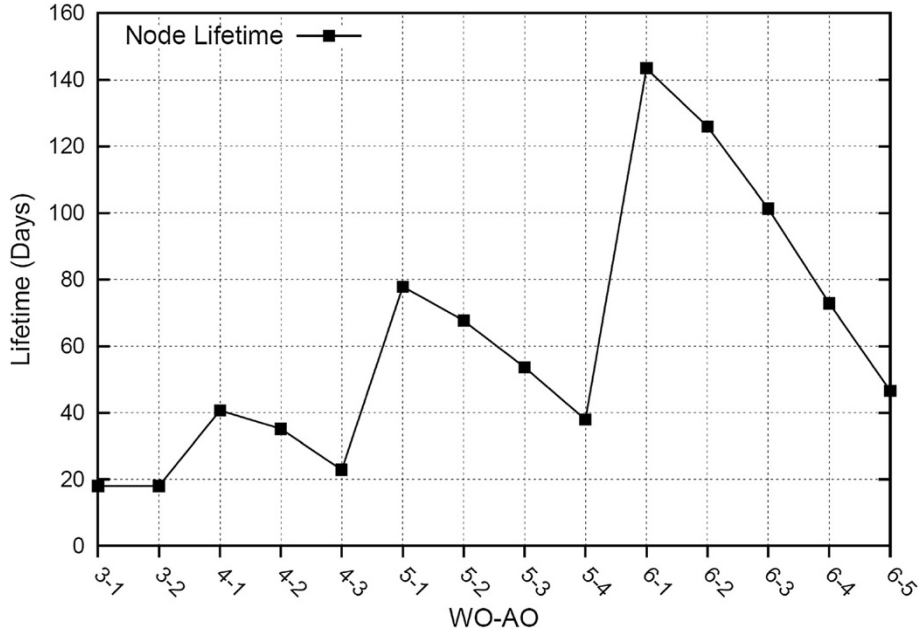

Figure 5 Energy-consumption diagram during a WI for an intermediate node. Two phases should be distinguished: retransmitting messages and synchronization with its *region synchronizer* or *mesh coordinator*.

sleeping when it receives a message from its predecessor node, provoking the loss of this message and its later retransmission, therefore increasing the delay. This issue is influenced by two different parameters, namely, the SR and the SI. Both parameters are interrelated because the number of regions into which the entire network is divided strongly affects the SI. Therefore, to avoid losing messages, the network needs frequent and accurate SIs, and obviously the selection of an appropriate number of hops by region. In this sense, a high duration of the SI, along with a high SR parameter, causes an increase in the clock's drift among nodes of a same region. This causes a poor precision and a considerable synchronization error. Figure 3 also shows how the lack of synchronization results in a higher number of messages lost, thus worsening the throughput. It should be noted that values for SR greater than 5 are not depicted because they lead to a significant accumulative clock drift that affects all metrics very negatively. Intuitively, it may be inferred that, to relieve the loss of messages, the number of synchronization actions should increase, that is, the values for the SI and the SR should be lower. However, under these circumstances, a more often stopping of the dissemination of information occurs, increasing the latency and jitter, as depicted in Figure 3. This important drawback is carefully illustrated in Figure 4, where the latency of a routing node is shown for the set of values WO = 4, AO = 3, SI = 30, and SR = 3. From all data messages injected in the network, it will have some of their corresponding transmissions which may coincide temporarily with the *synchronization period* (SD). Low values of SI and SR involve a bigger number of *undesired coincidences*. When this happens, data messages are delayed by SES, because any transmission will not progress until the synchronization process finishes. On the contrary, data and synchronization messages might collide, implying later retransmissions. Retransmitting a synchronization message in a time different from the timestamp encapsulated in it provokes an inaccurate synchronization process. As a consequence, SES does not satisfy the strict delay requirements for data transmission in the *reservation-based method*, and therefore its main design goal is not accomplished.

According to the paragraphs above, we can indicate that the best throughput is achieved when $2 \leq AO < WO$, taking into account that for this aim the WO must not be greater than 5. Furthermore, Figure 3 shows better results for the combination formed by three hops per region (SR = 3) along with values from 30 to 100 for the SI. For this scenario, in the *reservation-based method*, SES attains the greatest data transmission rates and the lowest latency and jitter.

4.1.2. Power-consumption simulation

To address the energy-consumption issue, a model based on the MicaZ mote (MicaZ Datasheet) was programmed and simulated. MicaZ is a platform which offers an excellent solution for the design and implementation of a great variety of LR-WPAN applications at low cost. MicaZ is equipped with a microcontroller (ATmega128L) and a transceiver (CC2420), which support communications at 2.4 GHz with 250 kbps nominal transmission bitrate. Both the microcontroller and the transceiver can be in four different power modes/states denoted as Transmission, Reception, Idle, and Sleep, respectively. The complete power-consumption values related to the different states and transitions between them are listed in Table 1.

The consumption of energy is obtained from the following premises:

- For the *reservation-based method*, the selection of the operational mode for continuous transmission rate is a key issue, because all nodes in a source–destination path have a message ready to be delivered within the corresponding time slot of the *inactive period*. According to the IEEE 802.15.5 specification, the value of a reserved time slot is 625 symbols in the MAC sublayer or 10 ms at 2.4 GHz. This value is smaller than the sum of transition times (C_p and C_R), both needed for switching from the sleep to the transmission/reception mode.

Table 1 Power-consumption states of the MicaZ mote

Acronym mode	Mode definition	ATmega128L CPU	CC2420 radio module	Total
S_0	Sleep	45 μW	3 μW	48 μW
C_P	Transition between S_0 and S_1	10.3 μJ	691 pJ	10.3 μJ
		22 ms	970 μs	22.712 ms
S_1	Idle	23.94 mW	60 μW	24 mW
C_R	Transition between S_1 and $S_{2/3}$	–	6.63 μW	6.63 μJ
		–	192 μs	192 μs
S_2	Reception	12.18 mW	59.1 mW	71.28 mW
S_3	Transmission	14.47 mW	52.2 mW	66.67 mW

Therefore, in order to wake itself up and to carry out the transmission/reception tasks, a node needs more than one slot, making this procedure unfeasible. In this case, the usual operation is to maintain all nodes always in the idle mode (S_1 state). Then, the only transition state (C_R) executed guarantees the complete transmission and reception tasks in a slot.

- The Mesh Coordinator is always in the receiving mode (S_2 state). In real implementations of the mesh coordinator node, it usually incorporates an external power supply, so as to alleviate the power-consumption associated to being always ON.

In order to conduct a comprehensive power-consumption evaluation study, we present a brief classification according to the node's role as designed by the IEEE 802.15.5 standard to find out how much energy is wasted by each type of node. To this aim, the device's role falls into three categories: (1) mesh coordinator or region synchronizer; (2) sender or receiver node; and finally (3) intermediate nodes which are responsible for routing data from the sender to the receiver. It should be remarked that case (1) is focused on the synchronization process, and cases (2) and (3) are associated to data transmission. Consequently, a node different from the mesh coordinator (and region synchronizer) that does not perform any data transmission should present minimum power-consumption, given by its contribution to the synchronization process. In this context, the mesh coordinator or region synchronizer node is responsible for starting the synchronization process with their children. The energy consumption of these nodes is the one corresponding to the transmission mode (S_3 state), because they have to broadcast the *synchronization request* message and then automatically switch to the reception mode (S_2 state) so as to receive the *synchronization reply* messages of their respective children. These children remain in the reception mode so that they can listen to the *request* messages, changing again to the transmission mode in order to forward the *reply* messages.

On the other hand, in data transmission, the power consumption is referred to cases (2) and (3). For case (2), a receiver node is only in the S_2 state during the time required for receiving the data message; while a sender node remains in the state S_3 for delivering one message to the network. Both, receiver and sender nodes occupy a unique slot within the *inactive period* with these power states, but using a different reserved slot. The sender always reserves the same time slot (number 0). However, the time slot used by the receiver depends on the hop distance of the current sender [e.g., if the *reservation request* message has been forwarded over two nodes (two-hops) before reaching the receiver, it will use

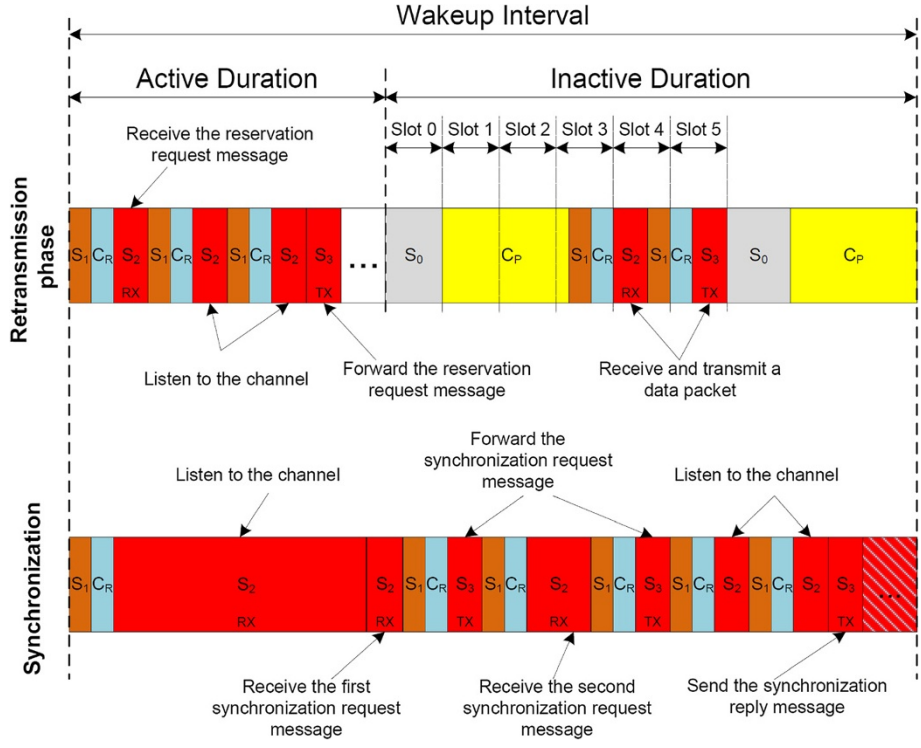

Figure 6 Intermediate node lifetime for different time periods and SI = 30, SR = 4.

the time slot number 2]. Finally, for case (3), each intermediate node needs two slots per data message within the *inactive period,* one for listening to the physical medium and receiving the message under consideration, and another slot for retransmitting it to the next intermediate node along the communications path. This way, the amount of energy consumed is greater than the one corresponding to case (2), where a single slot is used for transmitting (sender) or receiving (receiver).

As it was already mentioned, in large multi-hop networks, the network lifetime is mainly influenced by the intermediate nodes. Hence, for a better understanding, Figure 6 illustrates the energy-consumption diagram of a WI and a SD of a routing node. In the WI, the retransmission of the data is carried out by an intermediate node which always works under a continuous flow of messages. To this purpose, first, this node completes the reservation process within the *active duration* and second, it receives data from the previous node and forwards it to the next node in the route during the *inactive duration*. Then, this node itself is synchronized with its region parent node within an SD period, which, as indicated by the standard, takes the same time as the WI.

Figure 5 represents the metric lifetime for an intermediate (routing) node measured from its starting point of operation in the network to the instant that its battery depletes. These values are defined by a particular *wakeup* and AO parameters. We follow the same scheme of Figure 6, but intended for the case of a low SI to achieve a tight and accurate synchronization among the members of a same region. To this end, we have chosen that the synchronization process of the node be carried out each 30 WI*s*, that is, a node operates in the transmission data phase during 30 WI*s* plus one additional WI dedicated to perform the synchronization process. The number of hops per region (SR) is set to 4. It should be remarked that the routing device selected, a MicaZ, is operating all the time, that is, retransmitting messages and that each node is powered by two lithium AA batteries with a capacity of 3000 mA/h each (Energizer-L91).

Figure 5 shows that the best results are obtained when WO = 6 and AO = 1, where the batteries of a device working continuously deplete in 143 days. However, Figure 5 also reveals that low values of the WO parameter or high ones of AO restrict the lifetime of nodes. In particular, at low WO, the short duration of the *inactive period* implies not having enough time to do the transition to the low power states, therefore not saving energy. On the other hand, a high value for the AO parameter means that the node is in the *active duration* for a long period of time, unnecessarily wasting energy listening to the physical medium. In addition, an increase in the SI parameter slightly improves the lifetime of an intermediate node, resulting in a 7% energy saving for values of SI close to 200 but at the expense of deteriorating other metrics**,** such as the throughput or the delivery ratio.

To sum up, according to the simulation results and depending on the particular requirements of an application, we can obtain solutions where SES achieves transmission rates from 4 to 6 kbps for a combination of WO–AO parameters with values ranging 3–2 or 4–3, respectively. These values are, a priori, enough for transmitting video in MPEG4 format (Koenen); note, however, that if the transmission flow is continuous, the battery depletes in a few days. On the contrary, if the application requires less bandwidth (e.g., a continuous audio MPEG flow at 1.2 kbps for a WO–AO ranging from 5–3), more energy can be saved and, as a consequence, the lifetime of the network improves. Furthermore, as can be observed, nodes need frequent SDs, thus reducing the number of messages lost but affecting adversely other network metrics such as throughput and latency. Nevertheless, *the main problem of a tight SES synchronization process (following the non-flexible rules of the IEEE 802.15.5 standard) is that it does not allow a regular bitrate data flow between source and destination, making its use for many delay-sensitive applications impossible, such as those supporting audio or video*. To overcome these inconveniences, we propose a new synchronization process denoted as *HIPESYN* that significantly achieves better network performance. This proposal is described in the following section.

5. HIPESYN synchronization algorithm approach

Our proposal for the synchronization scheme is based on the intuition that a more precise synchronization system must provide better performance results than the current operation of the SES synchronization algorithm. Our target is thus to design a new mechanism which reduces the intrinsic synchronization error introduced by the conventional SES algorithm and facilitates the development of applications with strict delay requirements without increasing the memory and computing overhead. It extends the traditional communication among WMSN nodes focused on sensor monitoring to other types of services such as delay-sensitive ones, therefore enabling the development of added-value applications over a WSN. The reduction of the intrinsic synchronization error will facilitate a precise estimation of the synchronization between a parent node and its children, or between different neighbors. In addition, our method is executed without generating special time periods to avoid the current high values for the latency and jitter figures. Notice, as it will be discussed in Section 5.3, that reducing the synchronization error implies less power consumption.

The SES synchronization scheme provides a model based on the broadcasting of simple messages to the nodes of the same region. It introduces an interval (the SD) in which the transceiver of nodes is permanently in the receiving state to listen to the synchronization messages, and it therefore consumes energy. To avoid this significant source of power consumption, SES tries to alleviate it by broadcasting synchronization messages between synchronizers and child nodes within the same region, together with an increase in the SI parameter. However, this operation presents serious limitations, as has been shown previously in Section 4. In particular, a high value of SI is associated with a loose synchronization of the entire network and, as a result, a higher number of data messages lost (that require additional energy for their retransmissions). Furthermore, the loss of broadcast messages without any possibility of recovery persists in a loose synchronization of the network with future data message retransmissions.

To overcome these drawbacks inherent to the synchronization messages broadcasting, a different model is necessary. Reviewing Section 2, where different synchronization algorithms (probabilistic, broadcasting, etc.) were discussed, we concluded that those based on sender–receiver techniques combine a reduced synchronization error along with a low implementation complexity and better energy efficiency. In particular, this type of schemes follows a request/reply transmission pattern which facilitates the detection of messages lost. Both, sender and receiver know exactly when their request/reply synchronization messages are sent. Therefore, if one of them does not arrive at its destination, the sender or receiver is quickly notified, retransmitting the message lost. The high precision of these algorithms (Freris & Kumar 2007) should also be remarked, which provides a drastic reduction of the synchronization error in comparison to the standard SES mechanism, further reducing the number of resynchronization tasks. The sender–receiver scheme is the foundation that inspires our new synchronization procedure operating by pairs. This new algorithm, HIPESYN, is addressed to improve the results of the IEEE 802.15.5 mesh networks with the goal of generalizing services incorporating video/audio or exploiting new applications. HIPESYN is described and quantified in the following section.

5.1 Synchronization operation of HIPESYN

HIPESYN consists of a sender–receiver scheme where the node that initiates the synchronization process (sender) is the same node as the one that is synchronized with the receiver. To this aim, each node of the pair involved in the synchronization must send one message with its corresponding timestamps which are stored and computed by the other node of the pair. Figure 7a illustrates the entire process of synchronization of node A with respect to node B by exchanging a pair of unicast messages and computing four different time values. Each one of these times addresses a specific functionality: T1 and T4 represent the time measured by the local clock of node A, while T2 and T3 are the time values registered by the clock of node B.

Concerning only the synchronization process, at time T1, node A sends a *synchronization request* message to node B. The *synchronization request* message contains, among other data, *the level number* (number of hops to the *MeshCoordinator*) of node A and the clock value of T1. Node B receives this message at time T2, quantified as the sum of T1, d and Δ; where d and Δ denote the clock drift between both nodes and the propagation delay, respectively. Then, at time T3, node B sends back a *synchronization reply* message to node A. This reply message contains the *level number* of B and the T1, T2, and T3 timestamps. Finally, the process ends when node A receives this *synchronization reply* message at T4.

Assuming that the clock drift and the propagation delay do not change from the transmission of the *request* message to the reception of *reply* one; node A calculates their value by using the following expressions (1):

$$d = \frac{(T_2 - T_1) - (T_4 - T_3)}{2};$$

$$\Delta = \frac{(T_2 - T_1) + (T_4 - T_3)}{2} \quad (1)$$

Once this computation is done, the clock of node A is updated by setting it to the current timestamp of node B. Thereby, a new time value is set for the clock of node A. This new timestamp, which is equal to $T3 + \Delta$, is intended to reduce the difference between both clocks.

In our solution, when a node needs to synchronize with its neighbor, the algorithm guarantees a faster process, not requiring the use of a particular SI, as in the case of the SES scheme (SD). This is possible because our algorithm takes benefit from very low delays introduced by the wireless physical medium, and it avoids the contention-based medium access method (CSMA-CA) for transmitting the synchronization messages. In particular, the omission of the CSMA-CA protocol gives high priority to these synchronization messages, and as a result the accuracy of the synchronization process improves significantly. Together with the fact that all concerned nodes in the synchronization process operate within the same WI, this implies that the different timestamps may be computed in any moment during the *active duration* of the WI in course. Thereby, the synchronization messages are rapidly transmitted during the *active period*, when nodes demand a new synchronization process.

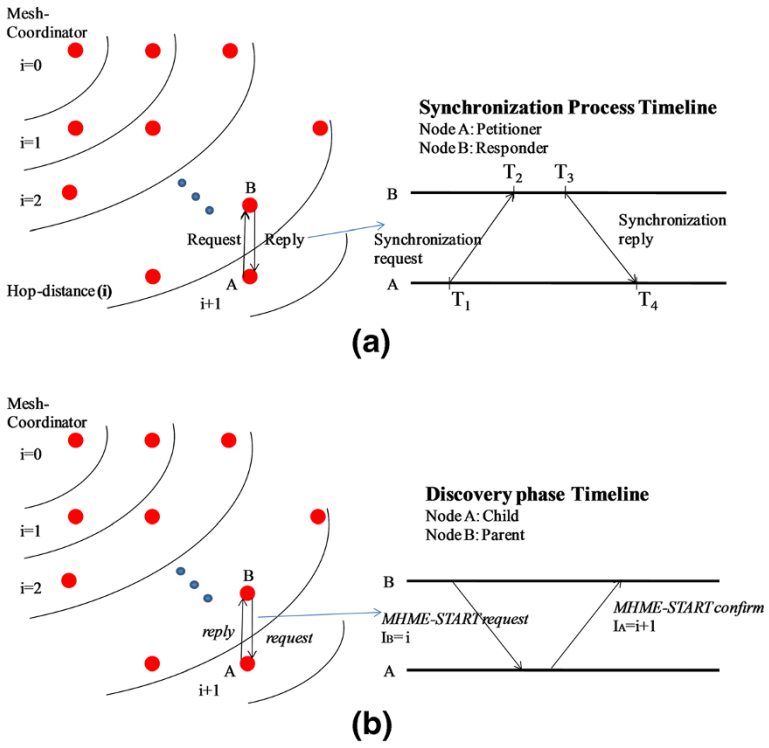

Figure 7 Normal synchronization operation and discovery phase.

5.2 HIPESYN description

Our proposal adopts the concept of a mesh topology based on regions, maintaining the active/inactive SES structure but adding a more precise synchronization algorithm. According to the standard, the creation of regions pursues the distribution of network tasks among their members, in order to alleviate the *mesh coordinator* load. In our case, the concept of region is also conceived as a dynamic factor, where the number of hops that take part in a region (and therefore the number of regions) may vary according to the maximum number of synchronization messages reported during the same *active duration* with the goal of enhancing the entire network performance. In addition, HIPESYN is used when neighbor nodes require a very accurate synchronization among them, simultaneously maintaining the compatibility with the mesh topology defined by the IEEE 802.15.5 standard. In particular, the idea behind this process is twofold: (i) our synchronization process does not occur over dedicated periods. Instead, each node calculates the instant to complete its own synchronization process and (ii) the synchronization does not always happen between a parent and its children. Therefore, any node is able to synchronize with any other neighbor node (at a distance of one hop). Given these principles, the synchronization process is divided into two phases: the first denoted as *network discovery*, where nodes share operational networking information requiring their activation, and the second phase or *normal operation*, where nodes synchronize. Both phases are described in the following sections.

5.2.1 Network discovery phase

The discovery phase is focused on the creation of links among nodes, providing robustness and flexibility to the data transmission. However, other stages of the communication such as the synchronization process are also influenced by an appropriate operation of this phase. HIPESYN is also critically affected by this issue, even more than in the case of IEEE 802.15.5. To guarantee these and other premises such as the efficiency and agility of the discovery phase, the formation of our mesh network is divided into two well-distinguished stages, where any node must be always awake and ready to send and receive information. These two stages are (i) the creation of the tree network and (ii) the local link establishment between one-hop neighbor nodes. Both follow the rules defined by the IEEE 802.15.5 standard but adapting them to our particular case.

Throughout the creation of the tree network, each node discovers its corresponding parent and children. It follows the same principles as specified by IEEE 802.15.5 to generate a tree network: the *Mesh Coordinator* broadcasts a *request* message using the *MHME-START.request* primitive to inform about the creation of a new network to its neighbor nodes, also reporting itself the number of hops, in this case, 0 hops. Therefore, any node that listens to

this message replies with a *MHME-START.confirm* primitive notifying that it is a new child of the *Mesh Coordinator* and reporting the number of hops to it, that is, 1 hop. Continuing this procedure, the whole network is discovered by sending a request message to its members that is confirmed by the nodes listening. When there are no more nodes to join the network, the address reporting process starts in reverse order, from the last leaf to the *mesh coordinator*. Each parent node in the network receives the number of current children of its branches which, in turn, is retransmitted to its corresponding parent. This procedure is repeated until the *Mesh Coordinator* is reached, which begins to assign addresses of two bytes size in the forward direction. As a result, each node finally knows the addresses of its parent and its children (if any), but also the number of hops to the *Mesh Coordinator* (see network on Figure 7b).

Differing from the IEEE 802.15.5 specifications, the number of hops to the mesh coordinator, denoted as *hop-distance i*, is recorded in each node because it is a relevant value that allows the calculation of the synchronization error between the nodes involved in a sender–receiver communication. Furthermore, each node listens to the discovery messages from other nodes, obtaining all available information of the vicinity. This information is stored and organized in each node, so as to build a *connectivity table* which determines the best paths to disseminate messages. In addition, the *neighbor/routing table* resulting from the execution of the TDLS algorithm (Zheng & Lee 2007) of the IEEE 802.15.5, which only contains information about all the neighbors placed one hop of each arbitrary node, is also extended to the maximum number of available nodes. The information of both tables, including the *hop-distance* of each node, is essential for a fast synchronization of any pair of neighbor nodes, which is one of the bases to significantly enhance the precision of the synchronization algorithm. Thereby, each node creates one-hop link with their neighbor nodes using the information available in its connectivity and routing tables in order to achieve the best path to the destination.

5.2.2 Normal operation phase

In contrast to the SES procedure, where all nodes are forced to synchronize in strict time intervals following a pattern (see Figure 2), in the HIPESYN scheme any node is able to synchronize to any neighbor node when the synchronization error between both exceeds a fixed threshold, that is, when they really need to synchronize. This fact gives high flexibility to the system. In this sense, the synchronization operation between neighbor nodes may be fulfilled at any moment during the *active duration*, by sending a *synchronization request* message from the denoted by us as petitioner node (thus starting the synchronization process) and by receiving the *synchronization reply* message. This last message is sent from the node we have called responder. Furthermore, these unicast synchronization messages have a high priority as requirement, prevailing over any other data transmission within the same *active duration*.

Thanks to this high priority design requirement and due to the accuracy of the HIPESYN operation, the first step of the normal operation phase is to find out the maximum residual synchronization error (denoted as e) when a pair of petitioner–responder nodes completes its synchronization process. According to the study in (Ganeriwal et al. 2003), this residual error is set up to 43 μs for each single petitioner–responder pair, which is an upper bound for computing the error incurred in the one-hop synchronization process. Finally, note that this error value is further required to compute the estimation of the synchronization error, as it is explained below.

Once the last synchronization process expires, each node is able to estimate its current synchronization error with the objective of updating this value. In this context, the estimation is computed in two different instants within the same *active duration*: (i) at the beginning of an *active duration* to control long periods without synchronization, and (ii) before transmitting or retransmitting any message from another neighbor. This estimation avoids performing complex operations in the sensor nodes, and considers two sources for the desynchronization: the clock drift since the last synchronization and the number of hops between source and destination. The first desynchronization component (clock drift d) accounts for the differences between clock drifts, in particular, between the clock drift of the petitioner and the clock drift of the responder. The maximum value for the clock drift in commercial motes is estimated to be 40 μs/s (Mock 2000), being the main reason to synchronize the nodes when they remain desynchronized after a long time. The second desynchronization source is due to the residual synchronization error (e) produced by the synchronization process itself, which is being accumulated by each petitioner–responder link. Therefore, each additional hop in the path from source to destination to forward a message has a predictable error associated to it, which is the sum of the errors experienced in each hop of the transmission path. As a result, each arbitrary node x estimates its current synchronization error, denoted as $\tilde{E}_x$ by means of the following expression

$$\tilde{E}_x = \left(T_{current} - T_{lastsynchronization}\right) \times d + \left(i_x + i_{\text{sink}}\right) \times e \quad (2)$$

where T_{current} and $T_{\text{last synchronization}}$ denote the current time and the time of the last synchronization, respectively. Values d and e (expressed in μs) are the nominal values of the error produced by the clock drift and the residual synchronization error when the synchronization process

has finished. An additional correction must be considered in expression (2), which is caused by the transmission of messages between neighbors located in the same or different *hop-distances*. This additional error is also considered in Equation (2) as the sum of the different *hop-distances:* $(i_x + i_{\text{sink}})$, being i_x the *hop-distance* of a node under study (x) and the i_{sink} *the hop-distance* of the sink. However, Equation (2) may be simplified when considering two special cases: (i) the sink node is the petitioner of the same responder as node x and (ii) a node does not forward messages during the *active duration*. In these two cases, Equation (2) can be simplified according to expressions (3a) and (3b) for (i) and (ii), respectively.

$$\tilde{E}_x = \left(T_{current} - T_{lastsynchronization}\right) \times d + 2 \times e \qquad (3a)$$

$$\tilde{E}_x = \left(T_{current} - T_{lastsynchronization}\right) \times d \qquad (3b)$$

Once the synchronization error is updated, it may exceed a given threshold, triggering again the synchronization process. This threshold value is determined according to the expected synchronization performance by the user. As a first approach, and in order to satisfy a tight synchronization as the one achieved by current IEEE 802.15.5 at SI = 30, the threshold is assigned to 2.1 ms, the maximum intrinsic synchronization error of the SES mechanism for these values.

Our HIPESYN synchronization scheme is also performed in cascade in a similar way to that in which IEEE 802.15.5 operates, maintaining the structure of SRs. This cascade sequence is executed in diverse cases such as in the first synchronization process or when consecutive nodes in the path to the sink are desynchronized. In these cases, when a petitioner node of the *hop-distance i* is synchronized to its corresponding responder node, it starts its own synchronization task, using the same *active duration*, saving time and energy. For instance, for two-hops, the responder receives the *synchronization request* from a petitioner, and instead of sending back the *synchronization reply*, it transmits a *synchronization request* to the next node in the path toward the destination, adding one more hop. When the *synchronization reply* from the next node in the path is received, the responder delivers the *synchronization reply* to its petitioner. In the case of the first synchronization process, this cascade sequence for the synchronization processes can be extended until it covers the *active duration* completely. In this way, the total number of hops taking part in the entire HIPESYN process within an *active duration* defines an SR. Note that in the conventional SES, the mesh coordinator establishes regions in which the data transmission process stops when no more nodes are synchronized during the *active duration* (AD). However, our concept of regions implies that a single AD is occupied, a priori, by a higher number of synchronization processes than in the SES mode (even if the length of the SD is longer than the AD of the HIPESYN algorithm), allowing to disseminate a bigger number of messages as well. This fact helps the progress of data to the sink be faster than in SES, thus improving the performance significantly, as it will be shown in the following section.

5.3 Performance evaluation

In order to show the real potential of the synchronization algorithm proposed, it will be evaluated and compared with the current IEEE 802.15.5 based on the SES mode. To this end, we have first calculated different figures of merit analytically, and then programmed a simulation environment that makes a complete performance evaluation of our HIPESYN process possible. In particular, the mathematical analysis allows for the calculation of the throughput and latency upper bounds, whereas the simulator aids to obtain the same performance figures as the ones discussed in Section 4. The final results reveal that our proposal offers a lower synchronization error during the normal operation and, as a consequence, better performance than IEEE 802.15.5 in the reservation mode.

We start with the analytical study; the first step consists of calculating the synchronization precision for both mechanisms under comparison. To this end, we compare the synchronization error, taking into account that the one referring the HIPESYN scheme can be obtained by applying Equation (2), presented in the previous section. Then, the second step is to find out the synchronization error introduced by the IEEE 802.15.5 standard. To obtain it, we use the study in (Mock 2000). This work conducts an intensive study about the synchronization accuracy in wireless networks, demonstrating that it is directly related to the time interval between two broadcasting synchronization messages, offering less precision when this time interval is small. According to research (Mock 2000), we have set the value of 2.1 ms as the maximum precision error. This value considers intervals of 1 s between synchronization messages. It implies a favorable case because the conventional IEEE 802.15.5 operation usually employs a lower time interval (one WI), resulting in a much worse precision. Taking into account the clock drift of commercial motes (d), the synchronization error in IEEE 802.15.5 (denoted as $\tilde{E}_{broadcast-x}$) can be expressed, following the same consideration as the estimation done in the calculation of the synchronization error, by the following equation

$$\tilde{E}_{broadcast-x} = Threshold + \left(T_{current} - T_{lastsynchronization}\right) \times d \quad (4)$$

On the other hand, the synchronization error of the HIPESYN algorithm is calculated periodically as shown by Equation (2). When this error exceeds a given threshold (selected by the designer), the synchronization process starts during the *active duration* in course. Comparing both expressions, (2) and (4), it is clear that

$$\tilde{E}_x = \tilde{E}_{broadcast-x} - Threshold + (i_x + i_{\text{sink}}) \times e \quad (5)$$

Observing Equations (4) and (5) the following lemmas can be enunciated.

Lemma 1. When the sum of hops for the forward–reverse path is less than 50, the synchronization error provided by the HIPESYN algorithm is always smaller than the one obtained by the IEEE 802.15.5 synchronization model.

Proof: By replacing values in Equations (4) and (5) (threshold = 2.1 ms and e = 43 μs) and making equal both errors ($\tilde{E}_x = \tilde{E}_{broadcast-x}$), we obtain that $(i_x + i_{\text{sink}}) \approx 50$.

In other words, the network dimension (parameter identifying the largest number of hops on the network) must be equal or less than 25 hops so as to achieve better performance than the one accomplished by the SES scheme. Therefore, the metrics obtained by HIPESYN for this network dimension are significantly better than the ones achieved by the SES approach.

On the other hand, when the source and sink are separated by more than 25 hops, the solution for a better performance implies an appropriate design of the SRs. For this purpose, it must be taken into consideration that the size of the regions depends on the length of the *active duration*. If the length of the *active duration* enables the number of synchronization processes between petitioner and responder to exceed the value of 25, our HIPESYN algorithm automatically will assign this value as upper limit. It will also define a new region and establish the node at *hop-distance* 25 as the synchronizer of this region. Once the regions are defined, synchronizers become sink nodes; therefore all the sources of each region will disseminate data using its corresponding new sink. This process guarantees a better operation than the SES mode as Lemma 1 indicates.

Lemma 2. Any node placed farther (at higher *hop-distance*) from the sink node requires tighter synchronization than a node closer to the sink.

Proof: This is directly obtained by considering Equation (2). Replacing the *hop-distance* on that equation by two values, i_1 and i_2, where $i_1 > i_2$, the estimated error is bigger for node i_1 than for node i_2. Therefore, node i_1 demands more synchronization processes.

This behavior is relevant to obtain a good network performance, since the closest nodes to their sink act as routing intermediate nodes to reach it, thus supporting more data message forwarding than the farthest nodes. Therefore, the number of synchronization processes on the closest nodes is reduced, leaving the available *active durations* for data message transmissions.

Taking into account the two lemmas above, the calculation of the throughput and latency for the HIPESYN algorithm is based on the IEEE 802.15.5 data transmission operation introduced previously in Section 3, but adding its own features. In particular, our algorithm replaces the SD by an *active period*. Therefore, the available bandwidth for the data message transmissions occurring in the *reservation-based method* is the same for both schemes (HIPESYN and current IEEE 802.15.5). However, our algorithm takes advantage of the filling up of the *active duration* by *reservation request/reply* messages in order to occupy a bigger number of reservation slots within the *inactive duration*. Finally, it should be noted that losses of data messages in the reserved slots (configured to 10 ms, according to the IEEE 802.15.5 standard) are not considered. The threshold imposed (2.1 ms, maximum acceptable synchronization error) is short enough to guarantee the message transmissions during the remaining length of the reserved slots (7.9 ms). As a result, the loss of data messages by synchronization error is discarded in the analysis.

The probability to send and receive successfully the *request* and *reply reservation* messages, called P_{RESERVE}, is written in expression (6). It refers to a single node during a complete SI (time interval employed as a unit of comparison with SES). This equation represents all the bandwidth required to send and receive reservation messages and consequently to transmit data messages in the appropriate slots of the *inactive period*. To do this, formula (6) includes (i) an additional AD_0 of special size WI substituting the SD and (ii) AD different from AD_0 during the entire SI. Note that the bandwidth available for the reservation messages is reduced by the time occupied by transmitting synchronization messages. This time is evaluated by means of the probability P_{SYNC} (since synchronization messages may not be required) and the time elapsed for delivering the synchronization messages ($2 \times T_{\text{TX/ONE-HOP}}$).

$$P_{RESERVE} = 1/SI \times (AD_0 + AD \times (WI/SI - 1) - (1 + P_{SYNC}) \times 2 \times T_{TX/ONE-HOP}) \quad (6)$$

In expression (6), WI, SI, and AD are the parameters specified by the IEEE 802.15.5 standard. Additionally, P_{SYNC} is the probability of performing a synchronization operation before sending the *reservation request* message

during the SI. $T_{\text{TX/ONE-HOP}}$ is also used to denote the time to transmit a synchronization message between a petitioner and its corresponding responder. Analyzing all parameters of (6), the only one that must be calculated separately is the probability of completing the synchronization process within the SI under study, that is, P_{SYNC}, since the remaining parameters are determined in the design phase of the network. In practice, this probability determines the throughput and latency achieved. This is because, when a long time is employed to transmit the synchronization messages, the time devoted to disseminate the reservation messages within the same *active duration* decreases, and, as a result, a less number of slots are dedicated to data message transmission, thus decreasing the throughput as well. Therefore, P_{SYNC} is an essential parameter in the evaluation of our algorithm.

The calculation of P_{SYNC} for HIPESYN is conducted according to the estimation of the synchronization error for this algorithm. Consequently, this estimation is evaluated in two different instants within the same *active duration*; at the beginning of the AD and before transmitting any message. Expression (7) computes $P_{\text{SYNC}}(i_{x,\text{WI}})$, which represents P_{SYNC} during a WI of the SI under study (denoted as WI_x). In this equation, $P_{\text{SYNC}}(i_{x,\text{WI}})$ includes the following two probabilities: (i) probability of an independent event of a synchronization process during the WI_x triggered by a distant node to the sink, and (ii) probability of the non-independent events after a long time since the last synchronization. In the case of the non-independent events, probabilities conditioned to previous WI slots are used. Therefore, we must also take into account that the probability of transmitting the synchronization message is also conditioned to the occurrence of the synchronization event during the previous *active durations*, as Equation (2) indicates.

$$\begin{aligned} P_{SYNC}(i_{x,WI}) = {} & P_{INST-SYNC}(i_{x,WI}) \\ & + \left(1 - P_{SYNC}(i_{x,WI-1})_0\right) \\ & \times P_{INST-SYNC}(i_{x,WI}) / \left(1 - P_{SYNC}(i_{x,WI-1})\right) \end{aligned} \tag{7}$$

where $P_{\text{INST-SYNC}}(i_{x,\text{WI}})$ is the probability that any node at the *hop-distance i* during an interval WI_x triggers and performs the synchronization process regardless of the previous WIs. $P_{\text{SYNC}}(i_{x,\text{WI-1}})_0$ is the probability (*a posteriori*) of executing the synchronization in the previous WI (denoted as WI_{x-1}) when the synchronization is also performed within the current WI_x. Analogously to $P_{\text{SYNC}}(i_{x,\text{WI}})$, $P_{\text{SYNC}}(i_{x,\text{WI}-1})$ represents the probability of accomplishing the synchronization process in the previous WI_{x-1}. All these probabilities are introduced and discussed in the following paragraphs.

$P_{\text{INST-SYNC}}(i_{x,\text{WI}})$ is also calculated as the probability that the synchronization error during an isolated AD exceeds the threshold within the WI_x and therefore the synchronization process is triggered by the petitioner. Similarly to the P_{SYNC} calculation, $P_{\text{INST-SYNC}}(i_{x,\text{WI}})$ is evaluated at the same two time instants: (1) at the beginning of the AD and (2) previous to transmitting a message. Considering an isolated AD, the trigger of the synchronization process at the time instant (1) is only caused by the clock drifts. It is calculated as $d \times WI/Threshold$, (where d is the clock drift and *threshold* the upper bound value of the HIPESYN algorithm to deliver the synchronization messages). Applying the aforementioned values for d (40 μs/s) and the *threshold* (2.1 ms), the result is $0.02 \times \text{WI}$ (WI expressed in ms). For the second case (2), before delivering a message, our method estimates the synchronization error caused by the distance to the sink node, independently of the last synchronization process. The probability of performing the synchronization process at this second time is denoted as $P_{\text{SYNC-TX}}$ and expressed by (8)

$$P_{SYNC-TX} = e \times (i_x + i_{sink-x}) \times P_0/Threshold \tag{8}$$

where P_0 is the probability of transmitting a message in the *reservation based-method* generated by the node under consideration or propagated from any other node of the network. Formula (8) can be simplified by assuming that the sink node is also the mesh coordinator. In this situation, $i_{\text{sink-}x} = 0$. Finally, the $P_{\text{INST-SYNC}}(i_{x,\text{WI}})$ in an interval WI_x is shown by expression (9)

$$\begin{aligned} P_{INST-SYNC}\left(i_{x,WI}\right) = {} & d \times WI/Threshold + e \\ & \times i_x \times P_0/Threshold \end{aligned} \tag{9}$$

Following with Equation (7), the second probability to calculate is $P_{\text{SYNC}}(i_{x,\ \text{WI-1}})_0$, which incorporates the probability to transmit the synchronization messages on the previous WI_{x-1}, in the case that the synchronization message is also transmitted at the current WI_x. This only happens when the length of the previous WI_{x-1} interval provokes that the clock drift exceeds the corresponding *threshold*. It should be remarked that the calculation of the probability $P_{\text{SYNC}}(i_{x,\text{WI}-1})_0$ is a constant value imposed by the length of the WI and the *threshold*. This probability is obtained by Equation (10).

$$\begin{aligned} P_{SYNC}\left(i_{x,WI-1}\right)_0 = {} & \frac{\left(Threshold - (d \times WI) \times \left\lfloor \frac{d \times WI}{Threshold} \right\rfloor\right)}{Threshold} \\ & \times \frac{d \times WI}{Threshold} \end{aligned}$$

$$= \left(1 - \left(\frac{d \times WI}{Threshold}\right) \times \left\lfloor \frac{d \times WI}{Threshold} \right\rfloor\right) \times \frac{d \times WI}{Threshold} \tag{10}$$

In the same way as the computation of $P_{\mathrm{SYNC}}(i_{x,\mathrm{WI}})$, we may obtain the synchronization probability in the previous interval, $P_{\mathrm{SYNC}}(i_{x,\mathrm{WI}-1})$, using the same expression as (7), but with the particularity that $P_{\mathrm{SYNC}}(i_{x,\mathrm{WI}-1})$ depends on $P_{\mathrm{SYNC}}(i_{x,\mathrm{WI}-2})$. Resolving the recursive operations and assuming that x tends to infinite (stationary behavior), Equation (7) is computed as shown by the expression (11). Intuitively, the result can be justified as follows: when the number of WI increases, the probability to perform one synchronization process over accumulated WI also increases, tending to the particular case of considering infinite WIs (stationary state) to the limit value of 1. Then, $P_{\mathrm{SYNC}}(i_{x,\mathrm{WI}})$ is computed by a simple sum of probabilities for all the WIs until the current one (WI_x), each one of them balanced by the independent component of the probability, $P_{\mathrm{INST\text{-}SYNC}}(i_{x,\mathrm{WI}})$, to carry out the synchronization process during an isolated WI, as referred by Equation (7).

$$P_{SYNC}(i_{x,Wi}) = P_{INST-SYNC}(i_{x,WI}) + \sum_{n=1}^{WI_x} \left[\left(1 - P_{SYNC}(i_{x,n-1})_0\right) \times P_{INST-SYNC}(i_{x,n})\right] \tag{11}$$

This expression (11) identifies the probability to fulfill the synchronization process within a unique WI. The accumulative probability during an entire SI gives the value searched. Taking into account the duration of a SI, the number of slots is obtained as $\mathrm{WI}_x = \mathrm{SI}/\mathrm{WI}$ and then Equation (7) can be expressed by Equation (12):

$$P_{SYNC} = SI/WI \times \left(P_{INST-SYNC}(i_{x,WI}) + SI/WI \times \left(1 - P_{SYNC}(i_{x,WI-1})_0\right) \times P_{INST-SYNC}(i_{x,WI})\right) \tag{12}$$

Finally, substituting Equations (9), (10), and (11) into (12), this expression depends only on the network design parameters and on the position of the node (defined by its *hop-distance*). The throughput is evaluated using the former results, and considering that all reservation messages are processed and accepted. To do this, we select a node at the *hop-distance* 15 ($i = 15$) as a representative element because this node has an equivalent traffic load in comparison to the intermediate node evaluated for the SES scenario (regarding the number of messages to transmit). Under these conditions, throughput is calculated as the number of messages delivered per second, multiplied by their length, as Equation (13) indicates.

$$Throughput = P_{RESERVE} \times messagesize/SI \tag{13}$$

Using the results from expression (6), the latency (T) may also be evaluated as the average delay suffered by the reservation messages on the network to reach the sink node. Therefore, we first calculate the average service rate (μ) of delivering messages to the sink as the average rate of reservation messages transmitted through the network from the source node i_x. Then, the next step consists of calculating the latency as the inverse of the previous metric, which is described by expression (14).

$$[T] = \frac{1}{[\mu]} = \frac{\frac{1}{SI}}{WI \times P_{RESERVE} \times i_x} + AD_0 \times \left\lfloor \frac{SI}{WI \times P_{RESERVE}} \right\rfloor \tag{14}$$

In order to solve Equation (14), it should be noted that the second term computes the time employed by a data message in the *reservation based-method* waiting for the end of the period AD_0. These results are also presented in Figure 8.

In addition to the analytical results for the throughput and latency metrics, we also present the evaluation outcomes of the HIPESYN obtained by computer simulation. The simulation was programmed in ns-2 and developed with the goal of validating the analytical expressions. The simulation scenario is the same already described in Section 4, both in requirements (size of messages, type of traffic, transmission rate, topology, etc.) as well as in the metrics to evaluate. This study can be found in our website at the URL http://www.ait.upct.es/~ajgarcia/IEEE802155/files.

Figure 8 shows the simulation and analytical results obtained for HIPESYN. The simulation results achieved for the SES standard operation are also plotted in this figure. It can be observed that the throughput for the HIPESYN proposal is higher than the one achieved for the original IEEE 802.15.5 SES. Our algorithm avoids dedicated synchronization slots, releasing more time available for data transmission. Figure 8 also reveals that for HIPESYN the delivery ratio approaches 100% of successful data messages, avoiding retransmissions. As for the throughput, it should be observed that, from Equation (2), nodes with low synchronization demands (neighbors to the sink node) trigger synchronization actions after long time periods (in the range of 30 times less than the node under study in *hop-distance* 15), thus being able to deliver more reservation and data messages

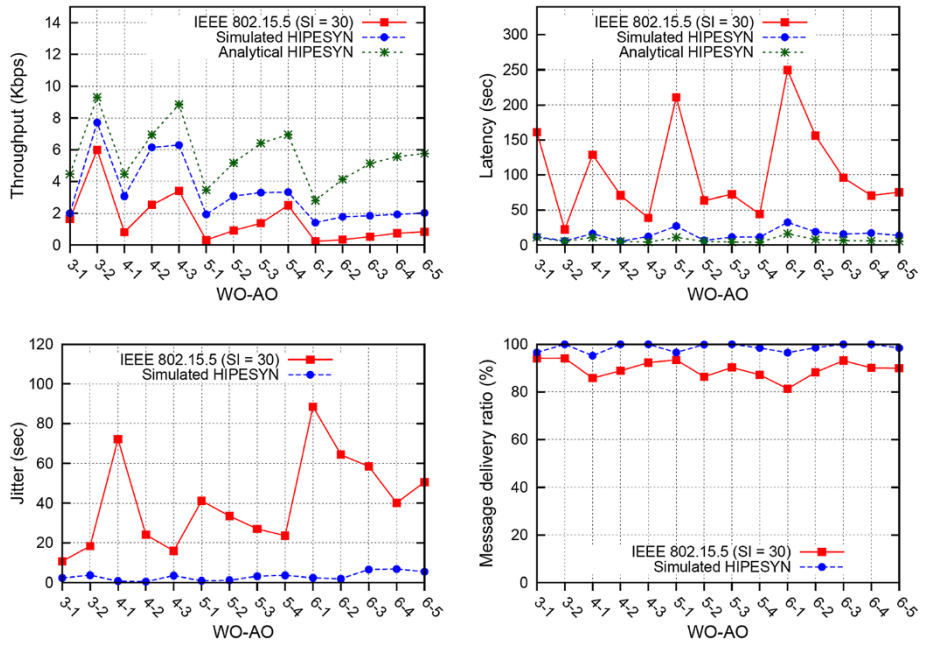

Figure 8 Throughput, latency, jitter, and message delivery ratio for the HIPESYN algorithm (analytical and simulated results) and the SES standard operation (simulated results).

than those with higher synchronization demands (located far from the sink node). This behavior of HIPESYN adjusts much better than the standard procedure of the IEEE 802.15.5 to the real network traffic demands, where neighbors of the sink node deliver more data messages than nodes located farther.

Concerning the latency, Figure 8 shows that the average delay suffered by the messages is substantially reduced. From these results, we can deduce that the probability of coincidence in time of the synchronization processes and the data transmission operations is extremely low in a petitioner–responder link. Therefore, the information flows continuously between source and sink, suffering minimum delays only owing to the sleep states of the nodes. On the other hand, the jitter is drastically reduced, since the synchronization process does not increase the delay variability of the transmitted messages. Overall, it should be noted that Figure 8 shows that analytical equations match the simulation results, being in general an upper bound.

Figure 9 compares the energy-consumption per bit obtained for the SES synchronization algorithm with the one achieved by our proposal. In both cases, the energy wasted in the synchronization process is taken into consideration. For the current SES, the simulation scenario agrees with the one described in Section 4, but particularized to the case of one SD by each 30 WIs (SI = 30) and three hops per region (SR). In the case of the HIPESYN simulation, a node placed at the *hop-distance* 15 is evaluated, which has a traffic load similar to that of the routing node selected in SES mode (Figure 5). This figure confirms that the SES synchronization algorithm consumes more energy per bit than our HIPESYN scheme, in order to maintain a tight synchronization. This is because our approach presents a better synchronization precision, which also has a noticeable decrease in the number of messages lost associated to it. Consequently, we obtain a more efficient utilization of the WI, delivering a higher number of reservation messages during the *active duration*, and thus reducing the time for transmitting the same amount of data. Observing Figures 8 and 9 together, we can affirm that our synchronization method

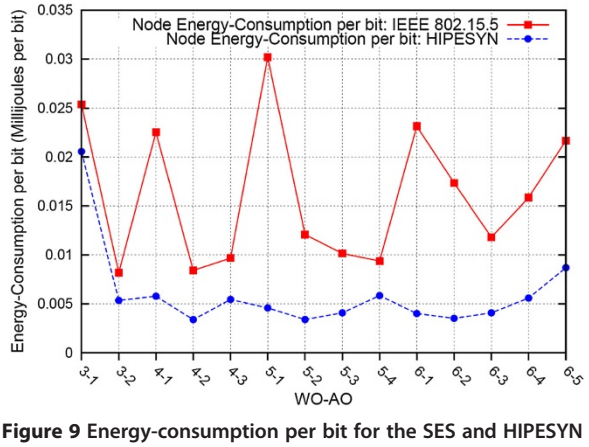

Figure 9 Energy-consumption per bit for the SES and HIPESYN synchronization algorithms.

achieves greater throughputs and lower latencies, simultaneously obtaining a more efficient use of the energy.

6. Conclusions

The possibility of delivering intensive traffic together with sensor monitoring data under efficient power consumption conditions will greatly contribute to the final commercial expansion of WMSNs. The synchronization process of SES is the mechanism specified by the recent IEEE 802.15.5 standard, which has been arisen to solve these issues. Unfortunately, to the best of authors' knowledge, there is not yet a thorough evaluation of the operation of the SES scheme, what creates doubts about its capabilities. In this paper, we have addressed the evaluation of this part of the standard, designing, and implementing a first simulation framework for this purpose. The performance evaluation of SES conducted reveals that the delivery of data messages presents high delay variability, mainly because of a deficient synchronization process, therefore jeopardizing the development of applications with strict latency and jitter requirements. Furthermore, another important drawback of the SES scheme is shown, namely the significant energy consumed by the intermediate/routing nodes which may lead to a fast battery depletion, reducing node, and network lifetime.

For these reasons, a new algorithm, called HIPESYN, based on a sender–receiver synchronization scheme is proposed. It is particularly focused on solving the shortcomings of SES. The idea behind this scheme is to create an extremely low computational, energy efficient, and accurate synchronization for WMSN which can be fully integrated in the IEEE 802.15.5 standard in order to improve its performance. The details of HIPESYN are presented along with an analytical and simulation study on its performance. The obtained results show a better exploitation of the communication channel than in the case of SES mode, and a noticeable energy saving for each one of the network nodes as well. As an added-value contribution, our synchronization algorithm provides a reduction of the synchronization error without generating special time periods, thus enabling the appropriate support for delay-sensitive traffic. All these contributions make our design suitable for the development of applications incorporating video or audio over WMSNs.

Our future research in this field is aimed at further improving the synchronization accuracy of the HIPESYN algorithm. In this sense, our current algorithm only evaluates the precision of the synchronization for the local petitioner, disregarding the synchronization error of the petitioner in relation to their neighbors. However, a better knowledge of these neighbors' synchronization errors should help to select a responder node that improves the precision of the petitioner's synchronization, but at the expense of increasing the nodes computation effort. A tradeoff between both issues should be investigated and might offer the best design option. Finally, other improvements should be addressed to better support the mobility operation.

7. Methods

Simulations of the IEEE 802.15.5 SES mode and the HIPESYN algorithm are conducted by using the Network Simulator 2 (ns-2) framework, a well-known C++ based simulation tool. For the HIPESYN analysis a MATLAB software environment is employed, the expressions obtained are based on Markov Chains Theory.

Competing interests

The authors declare that they have no competing interests.

Acknowledgments

This study was supported by the MICINN/FEDER project grant TEC2010-21405-C02-02/TCM (CALM). It was also developed in the framework of "Programa de Ayudas a Grupos de Excelencia de la Región de Murcia, de la Fundación Séneca, Agencia de Ciencia y Tecnología de la RM".

Received: 28 October 2011 Accepted: 30 May 2012
Published: 21 June 2012

References

IF Akyildiz, X Wang, W Wang, Wireless mesh networks: a survey. J. Comput. Netw. ISDN Syst **47**, 445–487 (2005). 10.1016/j.comnet.2004.12.001

IF Akyildiz, T Melodia, KR Chowdhury, A survey on wireless multimedia sensor networks. J. Comput. Netw: Int. J. Comput. Telecommun. Netw **51**, 921–960 (2007). 10.1016/j.comnet.2006.10.002

C Basaran, KD Kang, *Quality of Service in Wireless Sensor Networks, Guide to Wireless Sensor Networks* (Springer, London, 2009), pp. 305–321

J Chen, H Shen, H Tian, Energy balanced data gathering in WSNs with grid topologies, in *Proceedings of the Seventh International Conference on Grid and Cooperative Computing (GCC)*, ed. by B Schmitz and D Yang (, Shenzhen, China, 2008), pp. 362–368

H Cho, J Kim, Y Baek, Enhanced precision time synchronization for wireless sensor networks. Sensors **11**(8), 7625–7643 (2011). doi:10.3390/s110807625

J Elson, L Girod, D Estrin, Fine-grained network time synchronization using reference broadcasts, in *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, ed. by D Culler and P Druschel (Boston, MA, 2002), pp. 147–163

NM Freris, PR Kumar, Fundamental limits on synchronization of affine clocks in networks, in, Proceedings of the 46th IEEE Conference on Decision and Control ed. by D Castañón ed. by (Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, 2007), pp. 921–926

S Ganeriwal, R Kumar, MB Srivastava, Timing-sync protocol for sensor networks, in *Sensys '03*, in *Proceedings of the first international conference on Embedded networked sensor systems*, ed. by I Akyildiz and D. Estrin (Los Angeles, California, USA, 2003), pp. 138–149

AJ Garcia-Sanchez, F Garcia-Sanchez, J Garcia-Haro, F Losilla, A cross-layer solution for enabling real-time video transmission over IEEE 802.15.4 networks. Multimed. Tools Appl **51**, 1069–1104 (2011). doi:10.1007/s11042-010-0460-z

SN Gelyan, AN Eghbali, L Roustapoor, SAYF Abadi, M. Dehghan, SLTP: scalable lightweight time synchronization protocol for wireless sensor network, in *Proceedings of the third international conference Mobile Ad-Hoc and Sensor Networks (MSN)*, ed. by H Zhang, S Olariu, J Cao and D Johnson (Beijing, China, 2007), pp. 536–547

J Hui, P Thubert, Compression format for IPv6 datagrams in low power and lossy networks (6LoWPAN). IETF Internet draft-ietf-6lowpan-hc-15. February 2011, http://tools.ietf.org/html/draft-ietf-6lowpan-hc-15. Accessed 6 April 2012

A Hussey, A Nasipuri, J Sorge, R Cox, *Feasibility of using a wireless mesh sensor network in a coal-fired power plant* (SoutheastCon, Concord, NC, 2010), pp. 384–389

IEEE Computer Society, IEEE Standard for Recommended Practice for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements, part 15.5. Mesh Topology Capability in Wireless Personal Area Networks (WPANs). May 2009

IEEE Computer Society, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs). September 2006

M Jabbarifar, AS Sendi, H Pedram, M Dehghan, M Dagenais, L-SYNC, larger degree clustering based time-synchronisation for wireless sensor network, in *Proceedings of the Eighth ACIS International Conference on Software Engineering Research, Management and Applications (SERA)*, ed. by O Ormandjieva, C Constantinides, A Abran and R Lee (Montreal, QC, Canada, 2010), pp. 171–178

R Koenen, Overview of the MPEG-4 standard. ISO/IEC JTC1/SC29/WG11 N4668. March 2002, http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm. Accessed 6 April 2012

D Kominami, M Sugano, M Murata, T Hatauchi, Energy-efficient receiver-driven wireless mesh sensor networks. Sensors **11**, 111–137 (2010). doi:10.3390/s110100111

A Koubâa, A Cunha, M Alves, E Tovar, TDBS: a time division beacon scheduling mechanism for zigbee cluster-tree wireless sensor networks. Springer Real-Time Syst. J. **40**(3), 321–354 (2008). doi:10.1007/s11241-008-9063-4

MJ Lee, J Zheng, Y-B Ko, DM Shrestha, Emerging standards for wireless mesh technology. IEEE Wirel. Commun. **13**(2), 56–63 (2006). doi:10.1109/MWC.2006.1632481

MJ Lee, R Zhang, J Zheng, G-S Ahn, C Zhu, TR Park, SR Cho, C-S Shin, JS Ryu, IEEE 802.15.5 WPAN mesh standard—low rate part: meshing the wireless sensor networks. IEEE J. Sel. Areas Commun. **28**(7), 973–983 (2010a). doi:10.1109/JSAC.2010.100902

MJ Lee, R Zhang, C Zhu, T Park, C-S Shin, YA Jeon, SH Lee, SS Choi, Y Liu, SW Park, Meshing wireless personal area networks: introducing IEEE 802.15.5. IEEE Commun. Mag. **48**(1), 54–61 (2010b). doi:10.1109/MCOM.2010.5394031

X Li, CJ Bleakley, W Bober, Enhanced beacon-enabled mode for improved IEEE 802.15.4 low data rate performance. Wirel. Netw **18**(1), 59–74 (2012). doi:10.1007/s11276-011-0387-y

Y Liu, G Zhou, J Zhao, G Dai, X-Y Li, M Gu, H Ma, L Mo, Y He, J Wang, M Li, K Liu, W Dong, W Xi, Long-term large-scale sensing in the forest: recent advances and future directions of GreenOrbs. Front. Comput. Sci. China **4**(3), 334–338 (2010). doi:10.1007/s11704-010-0123-2

MA Lopez-Gomez, JC Tejero-Calado, A lightweight and energy-efficient architecture for wireless sensor networks. IEEE Trans. Consum. Electron. **55**(3), 1408–1416 (2009). doi:10.1109/TCE.2009.5278007

MicaZ Datasheet, http://www.memsic.com/support/documentation/wireless-sensor-networks/category/7-datasheets.html?download=148%3Amicaz. Accessed 6 April 2012

M Mock, Continuous clock synchronization in wireless real-time applications, in *Proceedings of the 19th IEEE Symposium on Reliable Distributed Systems (SRDS)*, ed. by B. Werner (Nurnberg, Germany, 2000), pp. 125–132

J Nieminen, L Qian, R Jäntti, Network-wide time synchronization in multi-channel wireless sensor networks. Wirel. Sensor Netw. **3**(2), 39–53 (2011). doi:10.4236/wsn.2011.32005

S Palchaudhuri, A Saha, DB Johnson, *Probabilistic clock synchronization service in sensor networks* (TR 03–418, Department of Computer Science, Rice University, Houston, 2003)

S Ping, *Delay measurement time synchronization for wireless sensor networks* (IRB-TR-03-013, Intel Research Berkeley Lab, 2003)

Römer, Time synchronization in ad hoc networks, in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc)*, ed. by N H Vaidya, M S Corson and S Das (Long Beach, USA, 2001), pp. 173–182

ML Sichitiu, C Veerarittiphan, *Simple, accurate time synchronization for wireless sensor networks* (IEEE Wireless Communications and Networking (WCNC), New Orleans, LA, USA, 2003), pp. 1266–1273

W Su, IF Akyildiz, Time-diffusion synchronization protocol for wireless sensor networks. J. IEEE/ACM Trans. Netw. **13**(2), 384–397 (2005). doi:10.1109/TNET.2004.842228

USC Information Sciences Institute, in, ed. by (, Marina del Rey, CA). Network Simulator—NS-2, http://www.isi.edu/nsnam/ns/. Accessed 6 April 2012

WirelessHart® Technology,http://www.hartcomm.org/protocol/wihart/wireless_technology.html. Accessed 6 April 2012

F Zhang, GY Deng, Probabilistic time synchronization in wireless sensor networks, in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing*, ed. by J N Liu R Conners and H B Zhou (Wuhan, China, 2005), pp. 980–984

J Zheng, MJ Lee, *A comprehensive performance study of IEEE 802.15.4* (Sensor Network Operation (IEEE Press, Wiley Interscience, 2006), pp. 218–237

J Zheng, MJ Lee, A resource-efficient and scalable wireless mesh routing protocol. Ad Hoc Netw. **5**(6), 704–718 (2007). doi:10.1016/j.adhoc.2006.11.003

ZigBeeTM Alliance, http://www.zigbee.org/. Accessed 6 April 2012

doi:10.1186/1687-1499-2012-198
Cite this article as: Garcia-Sanchez *et al.*: On the synchronization of IEEE 802.15.5 wireless mesh sensor networks: Shortcomings and improvements. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:198.