Chapter 3

A cross-layer environmental approach to monitoring with WSNs

3.1 Introduction

The PRISM protocol described in the previous chapter is a viable solution for real-time structural monitoring of structures. For example, one of its foreseeable applications is the monitoring of the vibrations produced in a power plant when the generators start or stop. Their switch-on instant is known, and it is possible to determine also when one or more elements will stop, for example for periodic maintenance or due to a reduction in the network load. In this scenario, continuous monitoring is not required – structural engineers are not interested in the vibrations produced during normal operation, and they need the data acquired during specific events only.

However, this is not the case for other scenarios, such as home automation, agricultural monitoring, and environmental preservation. In these contexts, the data rates involved are limited, the hard real-time constraints may be relaxed, and new samples can be acquired after minutes instead of seconds or less [3, 4]. On the other side, such a system is supposed to acquire samples continuously during time. In addition, compared to the scenarios considered for PRISM, typical environmental applications are much more dynamic: nodes can enter or leave the network after the initial deployment, and the routing algorithm should be able to survive the sudden loss of one or more sensors.

In other words, a more efficient solution is needed for ruling both channel access and data delivery. As far as the MAC layer is concerned, the scheduled approaches previously described in Section 1.2 are those best suited for environmental applications, although some preamble sampling methods might be applied too. Obviously, the selected channel access technique should be coupled with a proper routing layer and an efficient node activation protocol.

According to the traditional design principles of communication networks, the protocols running at different levels in the stack are independent and interact through static interfaces. However, the traditional layered design approach is not well suited

for resource constrained systems – interoperability and modularity are achieved at the expense of performance and efficiency. In the last years, the cross-layer approach has emerged instead as a more efficient design solution for wireless sensor network protocols. Looking at the big picture, cross-layering violates the reference communication architecture integrating protocols and mechanism usually running at different levels.

Cross-layering can be achieved by merging multiple layers into a single one, creating new interfaces for the exchange of additional data, or establishing interdependencies between any two layers of the stack [101, 102]. For example, information about the channel status or the received signal strength, obtained from the physical layer, may be used at the routing layer or at the MAC level either the next hop of the path [103] or the best coding strategy against current channel noise [104]. Similarly, information from the lower levels could be used to identify the best transmitter for a time slot, or to modify the duty cycle of the nodes [47]. As for other example of cross-layering, the information on the remaining battery capacity may be used to determine routing metrics [105], while geographic-aware routing exploits the knowledge on the position of the nodes to forward data efficiently [106]. A unified cross-layer communication solution which merges transport, routing, MAC and physical layers in a single protocol has been also proposed [107].

CERTO (Cross-layER proTOcol) is a cross-layer protocol for slow data reporting in environmental applications based on a simple scheduled rendez-vous mechanism. CERTO integrates the routing layer with the MAC mechanism in a flexible and easily extensible way – the protocol defines a number of fixed operating phases, but the algorithms implemented during these slots are not tied to the protocol and might be modified.

Related work on cross-layering has concentrated mainly in developing general schemes for efficient integration of physical, medium access control, routing, and transport layers [108, 109]. On the other hand, only a few works have investigated the optimization of cross-layer solutions for specific applications – examples are underwater exploration [110], multimedia transmission [111], and wide-area surveillance [112]. As far as environmental monitoring is concerned, the number of solutions for this type of application is limited to a very small number of works [113, 114] and there is still room for improvements towards a more efficient system.

3.2 The CERTO Protocol

CERTO is a cross-layer protocol which integrates routing, network formation, and synchronization functionalities with the MAC level. The protocol relies on a scheduled rendez-vous mechanism consisting of three main phases. In the first phase, all the nodes wake up, turn on the radio and listen the channel. When all nodes are awake, they move to the active part of the superframe. During this second phase, the protocol implements three periods dedicated respectively to network synchronization, network management and routing path setup, and finally to scheduled data transmissions. After all the acquired samples have been transmitted and no more data are expected, nodes go again to sleep mode to save power.

This solution combines features from some existing protocols: the introduction of a global scheduled rendez-vous has been inspired by Flexible Power Saving (FPS)

[115], the reservation of data slots and the use of the early sleeps option when no more data are expected come from T-MAC [37], while the use of a sleep/wakeup scheduling plus the separation of the data and the command phases are inherited from S-MAC [36]. To joint used of all these features allows reducing some of the issues in environmental WSNs, namely autonomy, reliability, robustness, and flexibility [116]. Environmental monitoring, in particular, is very demanding – harsh weather conditions and the remoteness of the monitored areas greatly impact hardware and software design. In these applications, periodic maintenance of the sensors is not possible – nodes should survive the whole deployment powered by a single battery. Similarly, the system has to account and solve connectivity problem or hardware failures, whereas changes in the topology should be automatically detected and managed without the need for local user intervention.

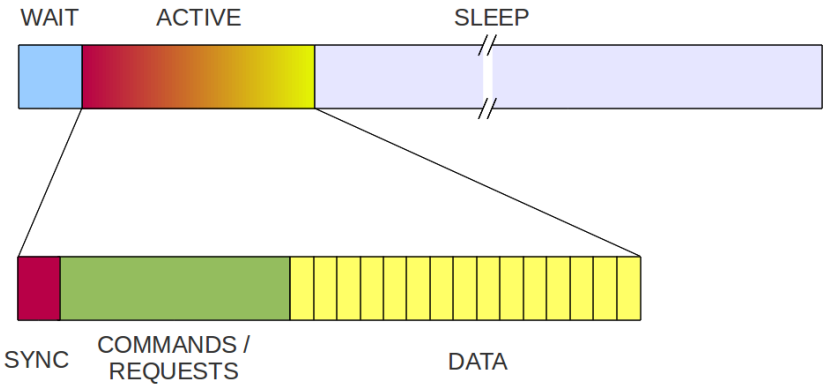

Figure 3.1: Structure of the superframe used by CERTO.

3.2.1 Structure of the CERTO Superframe

The structure of the CERTO superframe is shown in Figure 3.1.

The Wait Period

Nodes wake up during the Wait period and turn on the radio, waiting for the reception of a synchronization message. The introduction of this initial phase is motivated by the need to compensate clock drifts and errors occurred during a previous synchronization phase. Hence, this slot must be long enough to guarantee that all nodes have leaved the sleep status and are awake when the active period starts.

The Active Period

The Active period is composed of three main phases:

- **Synchronization**. This phase is used to establish a network-wide synchronization among the nodes. Node synchronization in a multi-hop network based can be achieved using different approaches, and the same solutions proposed

for PRISM in Section 2.5 may be applied also to CERTO. Specifically, a distributed algorithms like the Timing-sync Protocol for Sensor Networks (TPSN) [89] might be the best choice for the hierarchical network structure build during the network setup phase. After having synchronized its children, each parent node broadcast a short beacon containing its identifier, the amount of connected nodes and its distance in hops from the gateway. As described later, the information contained in this beacon is used by new nodes when joining the network.

- **Command**. The Command phase is used to deliver messages issued by the gateway, for example to inform the nodes about a variation in the global schedule or about a new data slot assigned to a node. Moreover, during this phase new sensors can request to join to network, existing nodes can leave, and changes in the routing topology can take place. The channel access is ruled by CSMA/CA, and unsent messages are postponed to the successive cycle.
- **Data**. The Data phase is divided into slots, and each node is assigned a dedicated time slot for transmitting the acquired samples. During a certain data slot, the source node and all the intermediate nodes of the routing path are required to be awake, while the remaining nodes of the network can turn off the radio and save energy.

The Sleep Period

During the Sleep period, nodes enter in power-save mode until the end of the superframe. This phase is crucial for saving energy: the longer it is, the lower the duty cycle will be. However, during the sleep period no commands can be issued to the network and new nodes can not enter – a trade-off is needed between system responsiveness and power saving. In addition, since a longer sleep period implies less frequent resynchronizations, the impact of clock drifts should be taken into account when configuring the network.

3.2.2 Routing in CERTO

The choice of the routing mechanism for the CERTO protocol has based on a few considerations on the functional requirements, the dynamic of the monitored environment, and the economical and energetic constraints of the target application.

A classical taxonomy of routing protocols for WSNs divides them into three main classes: flat-based, hierarchical-based and location-based [106]. In the last case, the positions of the nodes are exploited to design routing paths. The distance between neighbouring nodes and their relative location can be estimated using either a Global Positioning System (GPS) signal or local techniques, such as those based on the received signal strength, the angle of the received signal or the signal travel time. However, the use of GPS signals requires a dedicated and energy-demanding receiver, whereas the other techniques are affected by multipath noise and their accuracy is extremely variable [117].

Similarly, flat routing protocols based on gossiping or flooding schemes [118] have been discarded too – they waste energy and bandwidth sending multiple non-necessary copies of data in overlapping areas, and they can introduce significant

delays [106]. Data centric routing is a third kind of flat routing mechanism: the gateway sends requests to certain areas of the network waiting for data from one or more sensors located in the region of interest [119]. However, the aim of the CERTO protocol is to monitor the environment and periodically deliver the measured values coming from the whole network, not just a portion of it.

For this kind of monitoring applications, hierarchical routing is the best approach. Hierarchical protocols employ two types of nodes: cluster-heads and regular sensors. In this two-layers hierarchical architecture, the higher energy cluster node are used to organize the transmission and process the information, while the remaining sensor nodes perform the measurement task [106].

Routing in CERTO is therefore hierarchical, having a reservation-based scheduling, a reduced duty cycle due to periodic sleeping, global synchronization, and mechanisms for network formation. The only features inherited from flat routing is the use of homogeneous nodes, all having the same capacity in terms of computation, communication, and power. Moreover, all the nodes of the routing structure can be used both for routing data and for acquiring samples – the two tasks are not mutually exclusive.

Once deployed on the field, nodes running the CERTO protocol are able to organize themselves in a multi-hop routing tree topology rooted at the gateway, similar to the structure shown in Figure 3.2. Next, using the information acquired during the network formation steps, the nodes are able to route data towards the gateway. In addition, the network reorganizes when a new node is added or one of its elements fails.

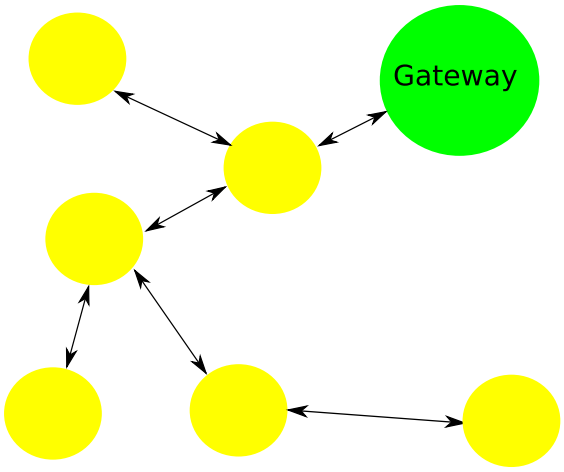

Figure 3.2: Example of a hierarchical routing tree.

The connection of a new node

When a new node enters the network, it immediately synchronizes itself with the rest of the system and chooses a parent. Then, it issues an association request and

waits for the assigned schedule from the gateway. A detailed description of the steps performed during this connection process follows.

1. First, a node enters the Beacon Scan phase, and listens the channel waiting for one or more beacons issued by the parent nodes in its range.

2. Then, the node wishing to enter chooses its parent according to the information contained in the received beacon.

The basic rule adopted by the protocol is to choose the node with the least number of associated children. However, if one the potential parents has more children but is also at an higher level in the routing path, it is elected as the new candidate parent. The adoption of this simple heuristic increases fairness and aims at building a balanced routing tree, but more complex algorithm for generating the spanning-tree could be implemented here without loss of generality.

3. Once the node has identified the potential parent, it goes back to sleep waiting for the next wake-up cycle – the remaining time is calculated from the value previously indicated in the beacon. At the end of the sleep period, the node wakes up and waits for a random backoff delay before sending the join request to the chosen parent node P.

The random backoff interval has been introduced to reduce the probability of collisions among multiple nodes in the same range. The join request received by the parent contains the physical unique identifier of the new node, the ID of the chosen parent P and the temporary ID of the child.

4. If the receiver node can not accept the request for some reason, it discards the request. Otherwise, it forwards the request to its parent node (the reason for ignoring the request could be that the packet has been corrupted, or the parent has already reached either the maximum number of allowed children W or the maximum hops distance H from the gateway).

5. In its turn, this node buffers the IDs of both the candidate parent P and the previous hop in the routing path. Then, it forwards the request to the next hop towards the gateway. The buffered information will be used later by the intermediate nodes to forward back the association reply. This step is repeated by the various nodes of the path, until the message reaches the gateway.

6. Once the message is received by the gateway, the scheduling algorithm running on it assigns a new ID to the requesting node.

7. Then, the new identifier assigned to the requesting node, together with its physical unique identifier and the ID of its parent P, are sent back along the path towards P.

8. When P receives the answer of the gateway, it broadcasts the message to all its listener.

9. The actual destination receives this message, recognizes itself as the recipient from the unique identifier included into the message, and so acquires the new ID assigned by the gateway.

10. Finally, the new node waits for the successive superframe cycle to synchronize itself with the parent node and to begin transmitting samples.

If the request message or the answer get lost and a reply is not received within a certain timeout, the joining node issues again the request. If no reply is received after n consecutive attempts, the node moves to the next candidate in the list of the potential father nodes. In addition, random backoffs of one or more cycles can be optionally activated by the child to separate two retransmissions of the join request.

Delivery of Data and Commands

During the Data phase, each node acquires samples during the assigned time slot and then transmits the data packet to its parent. In turn, the parent node forwards the packet to its parent. The process is repeated along the path, until the message is finally delivered to the gateway. During each time slot, the nodes involved in the forwarding path must be awake, whereas the other nodes can turn off the radio and save energy during this interval.

Within the Command phase, the exchange of messages can be local or global, and packets can travel both to and from the gateway. The delivery of a request from a node to the gateway does not differ significantly from the data transmission process described above. All the nodes which hear the message discard the message if they are not part of the path, while the next hop towards the gateway stores and forwards the packet. Similarly, a packet sent by the gateway to a specific node is forwarded only by the nodes along the path and discarded by the others. The forwarding mechanism is a destination based one, with the intermediate nodes which choose the next hop using the information buffered during the network formation steps. Finally, if a message is broadcast only locally, it is received and processed by the neighbor nodes but it is not forwarded.

Node failure or exit

During normal operation, temporary failures of links may happen, for example due to a sudden loss of connectivity produced by moving obstacle passing between two nodes, fading effects, or noise introduced by external electronic devices. Since the link is supposed to be restored quickly, there is no reason to modify the routing path – children can ignore the problem and reschedule transmissions in the successive periods.

However, if the link remains down for more than n consecutive cycles, the children can safely assume that the failure has become permanent, for example because the environment has changed or the parent node died. In this case, each child disconnects from the unreachable parent and looks for a new parent, repeating the connection process described in the previous section. After the node has found a new parent, the intermediate elements of the path is notified of the change in the routing structure. Moreover, they remain awake all the time during the first active phase – observing the passing packets, they learn the IDs and the transmission slots of the new nodes.

It is also possible that a node knows when it is going to die, namely because of the low battery level. If this is the case, it can inform its neighbors about its status, broadcasting a specific message during the Command phase. At this point, the recipients will be able to disconnect immediately from the dying node and look for new fathers as soon as possible.

3.3 Configuration of CERTO

The performance of the CERTO protocol depends on the parameters set by the user while configuring the network, namely the three-topology thresholds, the maximum amount of reconnection attempts, or the duration of the various superframe phases. However, there are a number of aspects which need to be considered, and different options may be selected depending on the number and the density of sensor nodes, the desired network coverage, the characteristics of the environment, and the constraints on the energy budget of the nodes.

3.3.1 Tree-topology thresholds

As mentioned in the previous section, the size of the routing tree created by CERTO is bounded by two thresholds: a first value H bounds the maximum number of children per parent, while a second parameters W limits the height of the tree.

Increasing the maximum height H of the tree means that the information can be delivered over more hops, allowing the system to cover longer distances. On the other side, an increased number of hops implies larger delays in the propagation of messages and data. On the contrary, a wider network reduces delays but requires shorter distances among nodes or higher transmission powers. In addition, the more the children per node are, the higher will be the traffic load and the power consumption of the nodes in the upper levels of the tree.

In addition, the thresholds must not be too strict – otherwise, some sensors could not join the network if all the neighbour nodes have already reached their children maximum amount.

3.3.2 Reconnection attempts

The value n is used by the routing functions to manage possible communication errors, for example when a parent can not accept requests to join from a new sensor or when it dies and is not reachable anymore. If these cases, if beacons are not received after n consecutive cycles, the child chooses another parent node. A small value of n means that the network will react faster in case of a permanent node failure. On the other hand, this could also bring to frequent changes or instability in the routing paths in presence of unreliable links subject to temporary failures.

3.3.3 Superframe duration

The duration of the superframe is not fixed, and depends on the length of its various phases. Similarly, the duration of each of the phases is related to a number of factors, among which the network size, the accuracy of hardware clocks and the desired lifetime stand out.

Wait phase duration

The Wait phase is used to compensate clock drifts and other inaccuracies in the alignment among the clocks of the sensor nodes. Given the maximum absolute error ϵ of the synchronization mechanism and the drift Δ of the physical clocks mounted on the sensor boards, the maximum clock error for a node is:

$$\epsilon + \Delta \cdot T_{superframe} \tag{3.1}$$

In the worst case, the discrepancy between the slowest and the fastest clocks will be twice the absolute error defined in Equation 3.1. Hence, this means that also the minimum duration of the Wait phase should be:

$$T_{wait} = 2(\epsilon + \Delta \cdot T_{superframe}) \tag{3.2}$$

This minimum value can guarantee that all nodes will be awake when the active period starts. Longer slots might be used for increasing the reliability of the mechanism, but at the cost of an increased energy consumption – once awake, the nodes turn on the radio and overhear the channel all the time, waiting for the synchronization signal.

Synchronization phase duration

The length of the synchronization phase is strictly related to the network size and the implemented synchronization mechanism. If a single reference signal is broadcast from the root node, its duration plus the processing time is the minimum length of the phase. If a different protocol is used instead, this value changes as well.

For example, the TPSN protocol performs synchronization in rounds along the tree – each round is initiated by the parent node, which aligns each of its children through round-trip measurements. In this case, given the duration T_S of a single child-parent measurement, the worst-case duration T_{synch} of the synchronization process for the whole network depends on the height H and width W of the routing tree:

$$T_{synch} = T_S \cdot H_{max} \cdot W_{max} \tag{3.3}$$

Command phase length

The duration of this phase is related the network management activities. Ideally, it should last enough to guarantee that all configurations and routing commands are exchanged successfully – if the interval is too short, an important message may be delayed or could even starve in the output queue. However, since all the nodes must listen the channel during this phase, longer periods increase energy consumption as well.

Data phase size

The part of the superframe devoted to the transmissions of samples consists of at least X equally sized time slots, where X is the number of active sensor nodes in the network:

$$T_{data} = X \cdot T_{slot} \tag{3.4}$$

Moreover, each slot must be large enough to guarantee that even the farthest node of the network can deliver samples in time. Hence, given the time T_{acq} taken by a node to acquire sample and the one hop propagation time T_{prop}, the total duration of the Data phase can be calculated as:

$$T_{data} = X \cdot T_{slot} = X \cdot (T_{acq} + H_{max} \cdot T_{prop}) \tag{3.5}$$

Sleep and duty cycle

Energy saving in this network is achieved by putting the sensor nodes in sleep mode as much as possible. While the length of the active part of the frame is more or less fixed, the duration T_{sleep} of the sleep phase can be adjusted to match the desired efficiency. As shown in Equation 3.6, the duty cycle δ will decrease with a longer sleep phase:

$$\delta = \frac{T_{sleep}}{T_{superframe}} = \frac{T_{sleep}}{T_{active} + T_{sleep}} \tag{3.6}$$

However, each node transmits data once in every cycle – the delay between two consecutive packets from the same sensor is therefore $T_{superframe}$.

This means that an increased duration of sleep phase can improve the energy efficiency and the lifetime of the sensor network but, at the same time, can limit the timeliness and responsiveness of the system.

3.4 Experimental evaluation

The CERTO protocol has been implemented with the purpose of evaluating its suitability for low-power and memory constrained sensor node. Then, this prototype implementation of the CERTO stack has been compiled and run in a simple network of Libelium's SquidBee devices [98]. Squibee is a wireless sensing platform based on the Atmel ATMega168 microcontroller, an 8-bit processor running at 16 MHz and featuring 1024 bytes of SRAM, an EEPROM memory of 512 bytes, and 15 KBytes of programmable flash plus 1 KB used to store the bootloader. The wireless part of this development platform is powered by an IEEE 802.15.4-compliant MaxStream XBee radio module [120].

A working implementation of CERTO written in C language fits in 325 lines of source code, coupled with a library of about 1200 lines containing basic input/output and wireless communication functions. This resulted also in an executable with a limited footprint: the binary version of stack produced by the *gcc-avr* cross-compiler for the ATMega168 processor requires just 8 Kbytes of ROM.

A number of experiments have been carried out both in indoor and outdoor scenarios in order to find an adequate configuration for the various network parameters. Similarly, an investigation of the main challenges related to radio signal propagation has suggested some node placement strategies for increasing the reliability of the wireless links. Using the results obtained from these tests, a small network was finally configured, deployed, and used to collected environmental parameters for hours.

Table 3.1: Forwarding time with different hop distances.

Distance	Time (s)
1 hop	49.8 ± 9.3
2 hops	153.3 ± 16.0
3 hops	257.0 ± 17.7
4 hops	360.0 ± 19.2

3.4.1 Network configuration parameters

A preliminary evaluation of the working protocol was made in a controlled laboratory environment, using four Squidbee devices. The network was a static one, and the nodes were forced to self-organize in a star topology. The system ran successfully for three hours, delivering samples periodically without losing any packets.

The test was repeated configuring the network in a line topology with the gateway as the first node, as shown in Figure 3.3. The protocol proved to be reliable, delivering all the packets in this multi-hop configuration. Moreover, this experiment provided useful information for determining the correct configuration values of the CERTO parameters described in the previous section

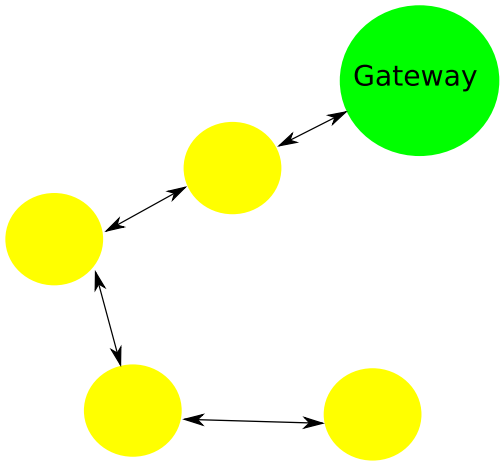

Figure 3.3: Example of line topology.

During this last test, the gateway collected also the time taken by each node to deliver the packets during the assigned slot. The average values obtained from the testbed are shown in Table 3.1. The generation of a packet and its transmission took about 50 ms on the source node, whereas each intermediate node introduced an additional delay of about 100 ms for processing the packet and forwarding it to the next hop of the path towards the gateway. These values are useful to determine the minimum duration of the data slots, given the maximum depth in hops of the routing tree.

According to the experimental values obtained from the controlled testbed, this

would allow to assign time slots of 0.5 s to each node. Similarly, the synchronization protocol implemented on the nodes requires about 500 ms per hop, and the process provides a clock alignment with a maximum error of 2 ms. However, the tests revealed that the radio module can take up to 300-400 ms to completely wake up and get ready to receive or transmit packets. For this reason, the minimum duration of the Wait phase must be at least 500 ms. Finally, the duration of the Command phase depends on the estimated average amount of topology changes, requests to join by new nodes or configuration command issued by users: a slot length of about 4 s can guarantee the correct delivery of 4-5 messages. This value might be sufficient for static networks of tens of nodes, while a wider network would require a longer Command phase.

These values, together with the network size, can be used to estimate the lifespan of the whole network. For example, let's consider 20 nodes deployed on the field and organized on a routing tree of 4 hops. According to the experimental values obtained from the controlled testbed, this would allow to assign time slots of 0.5 s to each node, for a total of a 10 s superframe data transmission slot. Adding other 0.5 s for the wait phase, 2 s for the synchronization mechanism and a command period of 4 s, we get an active part of the superframe lasting for 16.5 s. If the network is supposed to acquire and send data every 5 minutes, the resulting duty cycle is 0.055%. As noted in Section 2.7, the maximum lifetime of an optimized Squidbee node in active state is about 10 days - therefore, the achievable lifespan of this network can be estimated in 180 days, that is 6 months of operation.

3.4.2 Nodes placement strategies

Radio propagation is a key element for the success of an environmental wireless sensor network, and node placement is a crucial task for achieving an adequate coverage and quality of links. In this context, the height of a node over the terrain, as well as environmental factors may influence the reliability of the wireless connections and the maximum allowable distance.

In order to identify sensor placement strategies to increase the reliability of wireless links, a number of experiments have been carried out in four different environments. The first considered scenario is an indoor one, that is the Department of Electronics of the University of Pavia. It can be considered as an harsh environment due to the presence of walls, long passageways, a 4 m ceiling, active WiFi access points, and a lot of moving obstacles (people). Other tests have been carried out outdoor in the University campus, characterized by grass fields, concrete pavements, sparse trees, and again walking people crossing the scene. The same experiments have been then repeated in a crop field, with two nodes in line of sight and without notable obstacles in the surrounding environment. Finally, the last considered scenario is a small woodland in the Vernavola city park, just outside the town of Pavia.

In all the scenarios described above, the sender and the receiver nodes were put in line of sight 25 meters far from each other – this can be considered a realistic distance for wide-scale environmental application. The nodes were deployed randomly in four different positions in each scenario, and for each configuration 100 packets were transmitted from the source device to the recipient sensor node. A first batch of tests was performed putting the two nodes at ground level, but all the experiments

Table 3.2: Worst-case packet loss with varying heights of the node.

	Ground level	1.5 m
Indoor	100%	5.1%
Outdoor (crop)	100%	3.3%
Outdoor (campus)	100%	6%
Woodland	100%	7.1%

Table 3.3: Maximum achievable distance in different scenarios with a packet delivery ratio greater or equal than 90%.

Scenario	Maximum distance (m)
Indoor	60
Outdoor (crop)	150
Outdoor (campus)	118
Woodland	74

have been then repeated putting the antennas at 1.5 m height.

Table 3.2 shows the worst-case values obtained from each of the four scenarios, that is the results from the tests with the highest packet loss associated. Although these results are not sufficient to draw any general conclusions, they seem to confirm that the height of the antenna is a crucial factor for minimizing path loss in environmental monitoring applications, as suggested also in [100, 121]. Specifically, the 2.4 GHz wireless link is not usable when the nodes are put on the ground, while an height above 1 meter can guarantee a reliable link in almost all line-of-sight scenarios.

A second test has investigated instead the maximum achievable distance between two radio nodes in these four scenarios. In each specific environment, the sender has been put in a fixed position and the receiver has been moved far from it until the average packet loss ratio remained below 10%. Likewise the previous experiments, the test has been repeated with four random position for each scenario. Both the nodes were put at about 2 m height.

The worst-case maximum distances for each scenario are shown in Table 3.3. As expected, the maximum distance is limited in harsh environments, while more than 100 m can be covered by the wireless signal in open space. However, it is worth noting that these values might be considered as an upper bound, useful only for preliminary analysis. In reality, much shorter distance should be used in order to guarantee an acceptable reliability of the links.

In addition, the tests within the woodland showed that the presence of dense foliage may cause sizable changes in the connectivity among the nodes; this can result in unreliable links and frequent topology changes. Hence, seasonal changes should be taken into account while deploying nodes in an outdoor environment [121].

3.4.3 Network deployment

As a final test, 4 sensor nodes were deployed indoor within the Department of Electronics of the University of Pavia. The network parameters were configured

Table 3.4: Results obtained from the final indoor testbed.

	Avg. delivery ratio	Lifetime
Node 1	100%	16h 30m
Node 2	99.7%	21h 20m
Node 3	100%	21h 45m
Node 4	93.4%	16h 30m

according to the values presented above, and the nodes were required to acquire and transmit temperature and humidity samples every 1 minute. All the devices were powered by 9V batteries having a nominal capacity of 150 mAh. The nodes self-organized in a tree topology like the one shown in Figure 3.4.

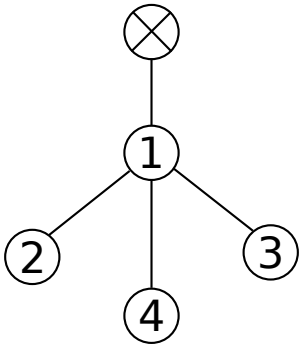

Figure 3.4: Topology of the final testbed.

After about 16 hours and half of operations, the node n°1 died – as a result, the node n°4 was disconnected too, since it was too far from the other two nodes and the gateway. The other two nodes reconfigured themselves and connected directly to the gateway, continuing to deliver sample for about 5 additional hours. We also calculated the average ratio of successfully delivered packets over the first 10 hours of operation. As shown in Table 3.4, the packet loss associated to the farthest node is about 7% while the other nodes successfully delivered almost all the packets.

3.5 Conclusions

The preliminary results obtained from the laboratory testbed confirm that a cross-layer approach, like the one adopted in CERTO, can provide significant benefits in terms of efficiency and memory footprint.

The mechanisms adopted for network formation and time slots assignment have been kept as simple as possible. Nonetheless, the protocol is flexible and robust. Moreover, some solutions currently used might be improved, and this without affecting the functionality of the system. For example, the choice of the parent node is currently based on a few simple considerations, but different heuristics might include also the remaining battery of the nodes to improve the energy efficiency of the routing tree.

A second test outdoor in an open field, showed that burst of packets are lost if the distance among two connected nodes is higher than 50-60 meters. The reasons for this unwanted behaviour are the changes in the network topology caused by the instability of low-power wireless links over long distances. This advocates for the implementation of a more complex heuristic for the organization of the routing paths, including for example information coming from the physical layer. The use of the Received Signal Strength Index or the Link Quality Indicator obtained from radio module could provide additional criteria for the choice of the best parent node.

Further investigations might concentrate also on the duration of the various phases of the CERTO superframe, which currently need to be set empirically by the user. A theoretical analysis of the clock drifts, the synchronization errors and the processing times could provide optimal values for these intervals.

Finally, the responsiveness of the system is related to the sampling frequency used to monitor the process – messages can not be delivered to the network during the sleep phase. A possible improvement would be the introduction of one or more short wake-ups during the inactive phase, useful for re-synchronizing nodes and issuing commands with reduced delays.