A Low-delay Protocol for Multihop Wireless Body Area Networks

Benoît Latré*, Bart Braem†, Ingrid Moerman*, Chris Blondia†, Elisabeth Reusens‡, Wout Joseph‡, Piet Demeester*
*Ghent University – IBBT – IMEC – Dept. of Information Technology – IBCN
Gaston Crommenlaan 8 box 201, B-9050, Gent, Belgium, firstname.lastname@intec.ugent.be
†University of Antwerp – IBBT – Dept. of Mathematics and Computer Science – PATS
Middelheimlaan 1, B-2020, Antwerp, Belgium, firstname.lastname@ua.ac.be
‡Ghent University – IBBT – Dept. of Information Technology – WiCa
Gaston Crommenlaan 8 box 201, B-9050, Gent, Belgium, firstname.lastname@intec.ugent.be

***Abstract*—Wireless Body Area Networks (WBANs) form a new and interesting area in the world of remote health monitoring. An important concern in such networks is the communication between the sensors. This communication needs to be energy efficient and highly reliable while keeping delays low. Mobility also has to be supported as the nodes are positioned on different parts of the body that move with regard to each other. In this paper, we present a new cross-layer communication protocol for WBANs: CICADA or *Cascading Information retrieval by Controlling Access with Distributed slot Assignment*. The protocol sets up a network tree in a distributed manner. This tree structure is subsequently used to guarantee collision free access to the medium and to route data towards the sink. The paper analyzes CICADA and shows simulation results. The protocol offers low delay and good resilience to mobility. The energy usage is low as the nodes can sleep in slots where they are not transmitting or receiving.**

I. Introduction

Nowadays, one of the major applications of wireless sensor networks is environmental monitoring. In these networks, an abundance of sensors is scattered around to collect and retrieve environmental data. A new use of sensor networks can be found in the area of wearable health monitoring. Carefully placing sensors on the human body and wirelessly connecting them to monitor physiological parameters like heartbeat, body temperature, motion et cetera is a promising evolution. This system can reduce the enormous costs of patients in hospitals as monitoring can occur real-time, over a longer period and at home [1], [2]. This type of network is called a Wireless Body Area Network (WBAN) or Wireless Body Sensor Network (WBSN) [3]–[5]. A WBAN consists of several sensors and possibly actuators equipped with a radio interface. Each WBAN has a sink or personal server such as a PDA, that receives all information from the sensors and provides an interface towards other networks or medical staff. Connecting health monitoring sensors wirelessly improves comfort for patients but induces a number of technical challenges like coping with mobility and the need for increased reliability.

An important requirement in WBANs is the energy efficiency of the system. The sensors placed on the body only have limited battery capacity or can scavenge only a limited amount of energy from their environment [6]. Consequently, in order to increase the lifetime of the network, energy efficient measures needs to be taken. From that point of view, several researchers are developing low power sensors and radios. Another possibility is the design of optimized network protocols to lower the energy consumption while satisfying the other requirements, an approach that is followed in this paper.

CICADA (*Cascading Information retrieval by Controlling Access with Distributed slot Assignment*), presented in this paper, is a low energy protocol designed for wireless, multihop, mobile body area networks. It is a large improvement of WASP (Wireless Autonomous Spanning tree Protocol) [7], where a spanning tree is set up autonomously and is used to route the data from the nodes toward the sink. Each cycle, the tree structure is used to allocate time slots to the different nodes in a distributed manner. CICADA divides such a cycle in a control subcycle and data subcycle, thus lowering the delay and introducing mobility robustness.

In the following, section II discusses related sensor network technologies. In section III, we give a protocol description where both the control subcycle and the data subcycle are explained. In section IV an upperbound for the delay is given, mobility support is examined and the energy efficiency is discussed. Finally we address simulation results in section V and section VI highlights our conclusion and future work.

II. Related work

Body area networks can be considered as a special type of sensor networks with its own specific requirements. The most important ones are the increased demand for reliability, energy efficiency and mobility support. Nodes can move with regard to each other, for example a node on the wrist moves in relation to a node attached on the hip.

The most commonly used technique for reducing energy consumption in sensor networks is controlling the power and duty cycle of the radio. Scheduled protocols, such as S-MAC [8], use scheduling to coordinate sleeping among neighboring nodes to avoid idle listening. A preamble sampling technique is used in WiseMAC [9] where a node regularly samples or polls the medium for a very brief time to check whether a packet needs to be received. To reduce the overhead associated with long preambles, a strobed sequence of short packets allowing for fast shutdown and response is used in [10]. An ultra-low duty cycle MAC that combines scheduling and channel polling is presented in [11]. Another commonly used technique is TDMA where the medium is divided into different slots in order to avoid collisions.

When considering wireless transmission around and on the body, important issues are radiation absorption by and heating effects on the human body. To reduce tissue heating the radio's transmission power can be limited. In [12], a different solution is presented. Clustering is used to distribute the energy cost and the increase of tissue temperature over all sensors.

The propagation loss around the human body is high [13], [14]. This means direct communication between the sensors and the sink is not energy efficient and even not always possible, thus there is a need for protocols supporting multihop body area networks.

A number of issues related to a body area network have been addressed in WASP [7]. This protocol sets up a spanning tree in a distributed manner. The communication uses timeslots which parents allocate to their children in a distributed way with a proprietary WASP-scheme.

III. Protocol description

A. General overview

CICADA is a cross-layer protocol. It uses the same packets to take care of both medium access as well as routing. They are used to detect the presence or absence of the children and to control medium access. The protocol sets up a spanning tree and divides the time axis in slots in order to lower the interference and avoid idle listening. The assignment of the slots is done in a distributed way, slot synchronization is possible because a node knows the length of each cycle. Each node informs its children when they are allowed to send their data. Routing itself is not complicated in CICADA anyway as data packets are routed up the tree which is set up to control the medium access.

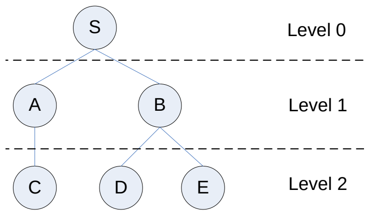

Fig. 1. Example network. The lines indicate the tree structure.

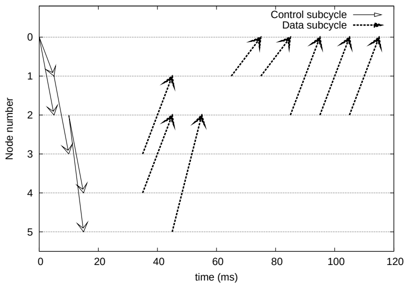

Fig. 2. Communication in the example network.

TABLE I
Steady-state control subcycle for the example network of fig. 1

Slot	1	2	3	4	5
	S	A	B	C + D	E

Data transfer is defined by a sequence of cycles. At the beginning of each cycle, the slots in the remainder of the cycle are assigned. The slot allocation is done by sending a scheme from a parent node to a child node. A node calculates its own scheme based on the scheme it has received from his parent. Each cycle is divided in two parts: the control subcycle and the data subcycle. Each subcycle has its own scheme for slot allocation: the control scheme and the data scheme respectively. These schemes are both sent in the control subcycle, it is used to propagate the schemes from the parents to their children. When all nodes have received their scheme, the control cycle has ended and the data cycle starts.

Each node is assigned 1 slot in the control subcycle. As slots in the control subcycle are only used to send the short scheme packets, slots can be shorter, e.g. by a factor 2 or more. When a node has received such a short scheme packet in a control slot, it can sleep as no more packets will arrive in that slot. The data subcycle is used to forward the data from the nodes to the sink. Unlike in WASP, the first nodes to start sending data are the nodes at the bottom of the tree. Doing so, all data can be sent to the sink in 1 cycle. This lowers the end-to-end delay tremendously.

Thus, as can be seen in figure 2, control information is sent downwards from the sink to all nodes in the control subcycle. In the data subcycle, all data is sent upwards to the sink. Further on in this section, we will discuss both cycles in detail using the example network from figure 1. The tree is set up in such a way that communication is only possible between a child and its parent or between siblings.

B. Control subcycle

The control subcycle is used for transferring the schemes (i.e. the control scheme and the data scheme) to all nodes. At the start of the control subcycle, the sink sends the first message. Table I gives an overview showing which nodes are allowed to send in which slot. The assignment of the slots in the control subcycle is done using the control scheme. Each control scheme contains the following information for the control subcycle:

- The control scheme indicates the order in which the children are allowed to send their control scheme;

TABLE II
Control subcycle information of the nodes

	S	A	B	C	D	E
TD_{CC}	0	1	1	3	3	3
WP_{CC}	0	0	0	1	0	1
remaining length	5	4	4	3	2	1

TABLE III
Steady-state data subcycle for the example network. The dots represent waiting slots.

Slots	1	2	3	4	5	6	7	8	9	10
S	.	.	.	.	A	A	B	B	B	X
A	.	C	X							
B	.	D	E	X						
C	X									
D	X									
E	X									

- The total length of the control subcycle L_{CC}, starting from the transmission of the control scheme of the sink. Stated otherwise, this is the total number of slots needed to allow all devices to send their scheme. In the example, the length is 5;
- The tree depth TD_{CC} indicates the level of the receiving node.

This information is used to calculate the exact slot at which the node may send. For example, node S sends its scheme with the following information:

- *control scheme* = AB
- *control subcycle length* L_{CC} = 5
- *tree depth* TD_{CC} = 1

Nodes A and B receive this information. Node A sees in the control scheme that it is allowed to send first, so it will send its data scheme in the following slot along with the additional information. Node B will send in the slot thereafter. The control subcycle length is the same. However, as node C can not send simultaneously with node B, node A will add a wait slot to its control scheme which becomes ".C". Thus, in the control cycle, node C has a *wait period (WP_{CC})* of length 1. The tree depth of node C is 3, which equals previous tree depth (1) + length of the control scheme (2). Node B will send the following: control scheme DE, control subcycle length 5 and tree depth 3. The control subcycle length and the tree depth are used to calculate the remaining length of the control subcycle with the following formula:

$$(L_{CC}) \text{ - } (TD_{CC}) + (WP_{CC}) \tag{1}$$

The remaining length indicates how many slots are left in the control subcycle and is thus used to know the start of the data subcycle. Table II gives an overview of the control subcycle information of the different nodes.

C. Data subcycle

For the example network of figure 1, the data schemes of the nodes or stated otherwise the division of the time slots in the data subcycle, can be seen in table III. The data scheme consists of 2 parts: a data period (length α) and a waiting period (length β). In the waiting period, the node must remain silent and should turn off its radio. In the data period, the node receives data from its children and sends data to its parent. For example, node B has a waiting period of 1 slot and a data period of 2 slots. The data scheme determines when the child nodes are allowed to transmit in the data period. The last slot of each data scheme is a contention slot which is used to allow new children to join the network, see section III-D.

TABLE IV
ChildTables

S

	α_i	β_i
A	2	3
B	3	4

A

	α_i	β_i
C	1	1

B

	α_i	β_i
D	1	1
E	1	1

Each node maintains a table for its children containing the following information:

α_i The number of data slots needed to forward the data received from node i

β_i The number of data slots needed for receiving the data from node i's children, the length of the waiting period of node i and 1 slot for contention.

This table is called the ChildTable. Each child is granted the number of data slots indicated in the ChildTable (α_i). The data period and waiting period are calculated as follows:

$$\text{data period } \alpha = \sum_{i \in Ch_n} \alpha_i \tag{2}$$

$$\text{waiting period } \beta = \max_{i \in Ch_n} \beta_i \tag{3}$$

In these formulas, Ch_n represents the children of node n.

Each time a node sends a data packet, a small amount of additional information is put in the data header. This header contains α_n and β_n, assuming that node n is sending. These values are calculated as follows.

$$\alpha_n = \sum_{i \in Ch_n} \alpha_i + \delta_i \tag{4}$$

$$\beta_n = \max_{i \in Ch_n} \beta_i + \sum_{i \in Ch_n} \alpha_i + 1 \tag{5}$$

In these formulas, δ_n represents the number of data slots that is needed for node n to transmit its own data. When a parent receives a packet from its child, it will extract this information from the header and update its ChildTable. Each child has to send this additional information each data subcycle. If a child has no data packet to send or to forward, it will send a HELLO packet to its parent containing only that information. Doing so, the parent will know that the child is still connected to the tree.

In table IV, the ChildTables for nodes S, A and B are given. As nodes C, D and E have no children, their ChildTables will be empty.

D. Joining the network

In each data subcycle, a contention slot is included to allow nodes to join the tree. A new child hears the data scheme of the desired parent and sends a JOIN-REQUEST message in the contention slot, preferably after a random delay. This join message also contains the number of slots a node needs to transmit its own data (δ_i), to enable immediate optimal resource allocation. When the parent hears the join message, it will include the node in the next cycle by updating its ChildTable. As it is assumed that a node i joining the network has no children, δ_i equals α_i and β_i equals 1.

E. Detection of connection loss

Each node will send at least two packets per cycle: a data packet or a hello packet and a control packet. If a parent does not receive a packet from a child for 2 or more consecutive cycles, the parent will assume that the child is lost and removes it from its ChildTable. If a child does not receive packets from its parent for 2 or more consecutive cycles, the child will assume that the parent is gone and will try to join another node.

IV. Discussion

A. Delay bounds

In TDMA protocols delay has an upper bound. As explained, CICADA uses a tree structure to send the data to the sink. Furthermore, as explained in section III-C, the allocation of the different slots is done in such a way that in one cycle all data can be sent to the sink. CICADA allows a flexible increase or decrease of the number of slots assigned to a node after just one cycle, represented by δ_i in (4). This guarantees reasonably low delays even with variable bitrate traffic.

The upper bound for the delay, i.e. the highest delay experienced in the network, is lower than the length of 1 data cycle as all data is sent to the sink in only 1 data cycle. It can be considered as the time between the transmission of the first packet and the arrival of the last packet. Formula 6 gives the delay, expressed in time slots:

$$delay_{upperbound} = \sum_{i \in V_S} \delta_i + \gamma(S) \tag{6}$$

The first term is the amount of data slots needed for forwarding the data from the level below the sink, where V_n denotes the set of all nodes in the subtree below node n. The function $\gamma(n)$ gives the number of waiting slots needed by node n. In a data cycle the first slot always is a contention slot. This slot should not be considered when calculating the highest delay. Therefore, $\gamma(n)$ already equals zero in the last but one level:

$$\gamma(n) = \begin{cases} 0, & \bigcup_{i \in Ch_n} V_i = \phi \\ \max_{\forall i \in Ch_n} \left(\sum_{m \in V_i} \delta_m + \gamma(i) + 1 \right), & \bigcup_{i \in Ch_n} V_i \neq \phi \end{cases} \tag{7}$$

The $\max$ function is needed as different branches of the tree are allowed to send data simultaneously.

Formula (6) shows that the upper bound for the delay is a function of the number of nodes in the network, depending on the tree structure and the length of a time slot. A minimum upperbound and a maximum upperbound for the delay can be defined.

The minimum upperbound is achieved when all nodes are directly connected to the sink, the single-hop case. Assuming we have a network of n nodes below the sink and each node has just one slot to send in a cycle, based on (6) the minimum upperbound reduces to

$$upperbound_{min} = 1 \tag{8}$$

When using CICADA in a single-hop network it reduces to a standard TDMA protocol, with an overhead of 2 slots: 1 for the contention slot of the sink and 1 for the contention slots of all other nodes.

The maximum upperbound is found when all nodes are at different levels, i.e. the tree is actually a single path. Using the same assumptions, the maximum upperbound becomes

$$upperbound_{max} = -2 + n + \sum_{i=1}^{n-1} i \tag{9}$$

$$= -2 + \frac{n}{2} + \frac{n^2}{2} \tag{10}$$

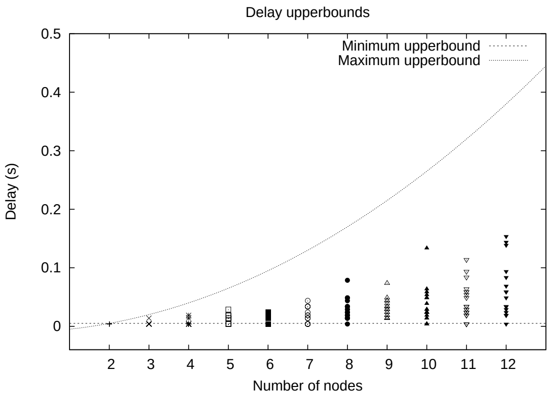

Fig. 3. Delay bounds related to the number of nodes.

Figure 3 gives the result of a delay bounds analysis in simulations. We simulated a static CICADA scenario with 2 to 12 nodes and a randomly generated connected topology. The graph shows different maximum delays experienced in the scenarios, compared to the number of nodes. Notice how all results are within the area defined by the minimum and maximum upperbounds. The calculated maximum delays are well below the maximum upperbound. This is to be expected as the topology of the maximum upperbound, i.e. a single path, is a very extreme case. Often, the nodes are placed too close together to construct such a topology.

As an example, figure 4 gives the layer 2+3 end-to-end delay, so delay caused by MAC and routing, for some of the nodes in the network shown in figure 7. This is the time between the transmission of the packet by CICADA and the arrival at the sink. The length of one cycle varies slightly and is about 110 milliseconds. It can be seen clearly that the delay is very low and limited to the duration of one cycle. The small fluctuations are caused by the change of the cycle's length as the cycles are updated according to the number of packets available. Figure 5 shows the layer 7 end-to-end delay, so the delay experienced by the application. As can be seen on the figure, a mismatch of the network scheduling and the data generation speed can introduce more delay at layer 7. When delivered to the network layer, a packet might have to wait for the next cycle until it can be sent.

B. Mobility support

CICADA supports mobility, which we believe is a necessity for multihop body area networks. This can be explained by the short range of the used wireless transceivers, as radios transmit at low power to save energy. This approach results in very small scale topologies where movements of the limbs can change the topology.

Currently there is no mobility model available for multihop wireless body area networks. In our studies of the protocol performance we looked at simple, humanly feasible movements to study the impact.

Nodes can join an existing network in 1 cycle: the moving node hears the new parent in its control subcycle and joins in its contention slot in the data subcycle. In the next cycle the node is assigned data slots and it can start sending.

A parent can monitor messages it expects to come from its children and vice-versa. Simulations show that marking a node as lost after 2 missed control packets suffices, as waiting for more cycles causes slow responses to a changed topology while tree stability is not improved much.

These two simple mechanisms make mobility support possible in CICADA: noticing parent loss and joining the new parent will take at most 3 cycles. Given the fact that cycles are short this results in good mobility support.

C. Energy efficiency

The most important causes of wasting energy in radio communication are idle listening, overhearing and collisions. CICADA takes care of all by assigning slots in the control cycle and using them in the data cycle. All slots are assigned so a node perfectly knows when it is allowed to sleep, when it has to send or when it has to switch on his radio to receive data. However, idle listening and overhearing can occur in the control subcycle as the nodes have to wait for the control scheme of their parent and consequently have to switch on their radios. However, the slots of the control cycle are shorter than the ones of the data cycle and nodes can sleep when a scheme from their parent has been received. In the datacycle, nodes only have to wake up when transmitting or receiving data. Using these mechanisms, the dissipation of energy is minimized.

The network topology plays an important role in the energy efficiency of the protocol, as the time spent idle listening and overhearing in the control cycle depends on the depth of the tree. If the tree structure is rather flat, the nodes will only spend a very short amount of time waiting inactively for the control scheme. Further, we envisage that a network topology with a tree depth of more than 4 levels is highly unrealistic, thus idle listening and overhearing will only happen during a short time. Overall, the tree should not have too many levels. That way, the nodes can sleep more and the upperbound for the delay will be lower, see (9).

CICADA has been developed to support high-traffic body area networks where delays should be low, i.e. all sensors send data often instead of buffering it locally. When used for low-traffic networks, an optimization is possible. If the sink detects that the nodes in the network have very few data packets to send, the data subcycles will be made longer without actually assigning those extra slots. Using this simple mechanism the sink can insert longer inactive periods without losing synchronization.

D. Using trees

Our basic assumption is that building a spanning tree is possible. Supporting mobility, which can be defined as transforming one tree in another one, is feasible because of the fast detection of movement.

Nodes higher in the tree are relaying a lot, which requires a lot of energy. This is due to the network structure based on trees. However, if a node really can not relay data because of energy shortage, it should not act upon joining nodes or drop children. Those nodes will then look for another parent and save power for the energy-constrained node. Another possible solution is the use of dedicated relay nodes near the top of the tree, i.e. near the sink. These relay nodes do not do any sensing, so they can use more power to support the network.

E. Differences with WASP

In CICADA all traffic reaches the sink in 1 cycle, which results in a lower delay. Each CICADA-cycle consists of 2 different subcycles: the control subcycle and the data subcycle. This results in enhanced mobility support: it is possible to leave a parent, to detect loss of a parent or a child and to join a node with much lower delays. CICADA does not rely on broadcasting to pass information from children to their parents, instead additional information is put in the data packets or HELLO packets are used. Moreover, generating CICADA schemes is easier because of the simple computation of the waiting period and the data period. This is important as CICADA is meant to run on sensors where computational resources are scarce.

V. Simulation

A. Implementation

We have implemented the protocol in nsclick, a system that allows Click Modular Router [15] instances to run inside the ns-2 network simulator [16]. In order to take the specific properties of communication near the human body into account, we have used the advanced propagation model described in [17].

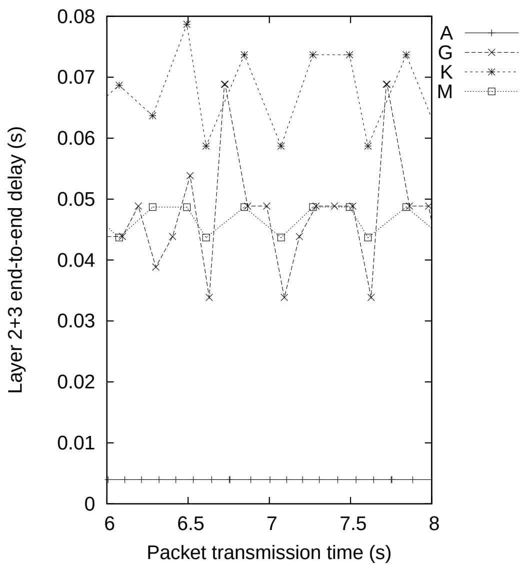

Fig. 4. Layer 2+3 end-to-end delays in a static network.

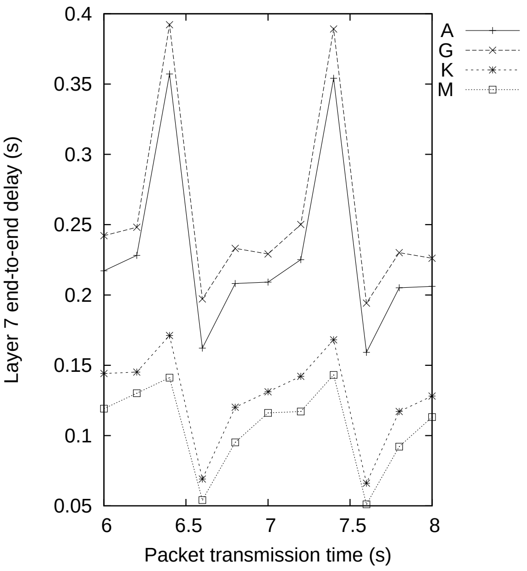

Fig. 5. Layer 7 end-to-end delays in a static network.

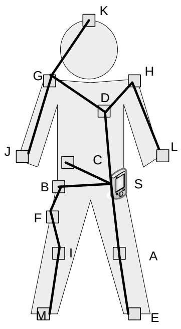

Fig. 6. Network on the body.

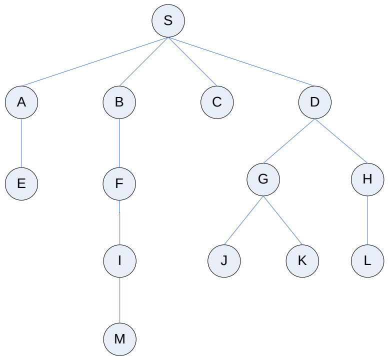

Fig. 7. Abstract view of the network.

This model calculates the path loss on the human body using the distance between the sender and receiver and a path loss coefficient of 3.3, whereas the path loss coefficient in free space equals 2. We have adapted the existing propagation model of ns-2 to use the formula given in [17]. Using this path loss on the human body, we have a more realistic view on the losses experienced near the human body and the corresponding influences on the network topology. We have validated the protocol in multiple scenarios with different topologies and mobility parameters.

B. Example scenario

Figure 6 shows an example network where some sensors are placed on a human body, figure 7 shows the generic tree view of this network. 13 nodes are each sending a constant bit rate stream to one sink with radios capable of transmitting up to 1Mbps. When configuring CICADA with control slots of 5 ms and data slots of 10 ms, the tree was set up after 192ms. When data was being generated at each node, the cycle length stabilized to 110ms.

We tested various scenarios, here we will discuss two scenarios with mobility as an example. In the first scenario, after 10 seconds node D is moved from its original position towards node G, which drastically changes the topology. Ten seconds later, node D is brought back to its original position. In the second scenario, node J moves closer to the sink. This corresponds with the movement of the right arm.

Figure 8 shows the layer 2+3 end-to-end delays of nodes D, G, K and J for the first scenario. Clearly these delays are very low due to the ability to deliver all data packets to the sink in one cycle. Between 10s and 20s, node D joins with node C. This influences all the nodes below node D as an extra hop is required to reach the sink. Hence the delay increases for all the depicted nodes. When node D moves, a transitional phase is noticed with larger delays. These are caused by the fact that the node waits 2 cycles before searching a new parent. As a node releases all of its children when it has lost his parent, all the nodes below node D need to rejoin. This will give cause to additional delay. The delay is the highest for node K. Furthermore, data was being buffered in the nodes while being disconnected. When nodes join the network, they empty their data buffers. In this case this lengthens the cycles, increasing the delay. The combination of joining and buffering effects explains the large delay. This phase lasts about 2 seconds. Notice that when the network connection is restored, delays are low again. The delay at the application layer, i.e. layer 7 end-to-end delay, experiences the same evolution as the delay at layer 2+3. The delay is higher as there is a mismatch between the moment the packet is delivered to the network layer and the moment the packet can be sent in the following cycle. As the cycle length is low, i.e. 110 ms in this scenario, the delay remains below 0.3 seconds in steady state. Again, a transitional phase happens. This figure was omitted due to reasons of brevity.

The results of the second scenario are shown in figures 9 and 9. At 10s, node J chooses the sink as parent. Consequently, the layer 2+3 delay for node J decreases. Looking at the application delay, we once again see a small transitional phase with higher delay.

VI. Conclusion and future work

In this paper we have presented CICADA, a cross-layer protocol for wireless body area networks. Based on a tree structure, CICADA controls communication with distributed slot assignment. Using both a control subcycle and a data subcycle, CICADA makes low delays and energy efficiency possible while preserving network flexibility.

CICADA can be improved in the future. The shape of the tree has a strong influence on the efficacy of the protocol. In the current implementation, the tree is formed depending on link formation times. However, smarter strategies exist [18] and will be investigated in future research. A possible solution might consist of adding extra information to the control scheme in order to communicate parameters like signal strength, load, remaining energy, etc.

Another possible future amelioration is the addition of on-the-fly slot resizing. This way, when less data is generated, slots and cycles can become shorter, further decreasing delays. Currently CICADA does not support traffic from the sink to the nodes. Implementing it would involve introducing uplink slots in the control cycle and adding a lightweight routing table to the nodes. A final improvement is the inclusion of slot synchronization, similar to [19]. CICADA has a tree structure so we expect this to be straightforward.

VII. Acknowledgments

This research is partly funded by the Fund for Scientific Research - Flanders (F.W.O.-V., Belgium) project G.0531.05, by The Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT - Vlaanderen) through the contract number 020152 and a PhD.grant for B. Latré and by the Flemish IBBT project IM3.

References

[1] S. Park and S. Jayaraman, "Enhancing the quality of life through wearable technology," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 22, no. 3, pp. 41–48, May-June 2003.

[2] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton, "Sensor networks for emergency response: Challenges and opportunities," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 16–23, 2004.

[3] C. Otto, A. Milenkovic, C. Sanders, and E. Jovanov, "System architecture of a wireless body area sensor network for ubiquitous health monitoring," *Journal of Mobile Multimedia*, vol. 1, no. 4, pp. 307–326, 2006.

[4] E. Jovanov, A. Milenkovic, C. Otto, and P. C. de Groen, "A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 2, no. 6, pp. 16–23, March 2005.

[5] E. Farella, A. Pieracci, L. Benini, and A. Acquaviva, "A wireless body area sensor network for posture detection," in *ISCC '06: Proceedings of the 11th IEEE Symposium on Computers and Communications*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 454–459.

[6] J. A. Paradiso and T. Starner, "Energy scavenging for mobile and wireless electronics," *IEEE Pervasive Computing*, vol. 4, no. 1, pp. 18–27, 2005.

[7] B. Braem, B. Latré, I. Moerman, C. Blondia, and P. Demeester, "The Wireless Autonomous Spanning tree Protocol for multihop wireless body area networks," in *Proceedings of the First International Workshop on Personalized Networks*. San Jose, California, USA: ICST, 2006.

[8] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *Proceedings of the IEEE Infocom*, USC/Information Sciences Institute. New York, NY, USA: IEEE, June 2002, pp. 1567–1576.

[9] A. El-Hoiydi and J.-D. Decotignie, "Wisemac: an ultra low power mac protocol for the downlink of infrastructure wireless sensor networks," in *ISCC '04: Proceedings of the Ninth International Symposium on Computers and Communications 2004 Volume 2 (ISCC 04)*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 244–251.

[10] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-mac: A short preamble mac protocol for duty-cycled wireless networks," University of Colorado at Boulder, Technical Report CU-CS-1008-06, May 2006.

[11] W. Ye, F. Silva, and J. Heidemann, "Ultra-low duty cycle mac with scheduled channel polling," in *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2006, pp. 321–334.

[12] T. Qinghui, N. Tummala, S. Gupta, and L. Schwiebert, "Communication scheduling to minimize thermal effects of implanted biosensor networks in homogeneous tissue," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 7, pp. 1285–1294, July 2005.

[13] L. Roelens, S. V. den Bulcke, W. Joseph, G. Vermeeren, and L. Martens, "Path loss model for wireless narrowband communication above flat phantom," *IEE Electronics Letters*, vol. 42, no. 1, pp. 10–11, January 2006.

[14] B. Latré, G. Vermeeren, I. Moerman, L. Martens, and P. Demeester, "Networking and propagation issues in body area networks," in *11th Symposium on Communications and Vehicular Technology in the Benelux, SCVT 2004*, November 2004.

[15] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, 2000.

[16] M. Neufeld, A. Jain, and D. Grunwald, "Nsclick:: bridging network simulation and deployment," in *MSWiM '02: Proceedings of the 5th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*. New York, NY, USA: ACM Press, 2002, pp. 74–81.

[17] E. Reusens, W. Joseph, G. Vermeeren, and L. Martens, "On-body measurements and characterization of wireless communication channel for arm and torso of human," in *4th Internation Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007)*, March 2007, pp. 264–269.

[18] S. Upadhyayula and S. K. S. Gupta, "Spanning tree based algorithms for low latency and energy efficient data aggregation enhanced convergecast (dac) in wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 5, pp. 626–648, 2007.

[19] L. Dai, J. Redi, and P. Basu, "An energy efficient and accurate slot synchronization scheme for wireless sensor networks," *Broadnets '06 Wireless Communication, Networks and Systems Symposium*, 2006.

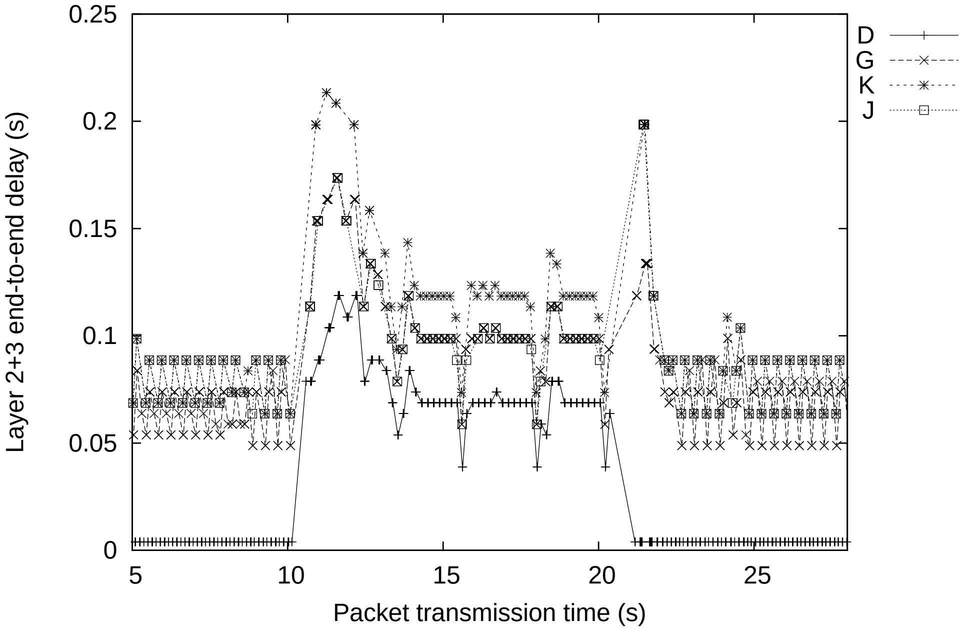

Fig. 8. Layer 2+3 end-to-end delays in the sample network.

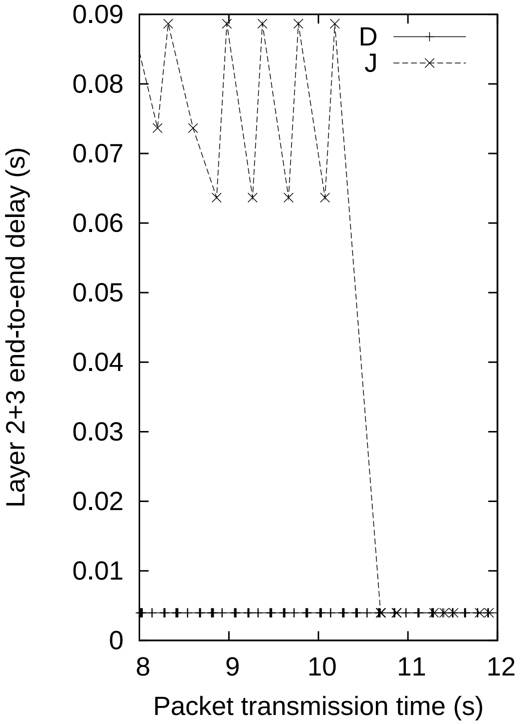

Fig. 9. Layer 2+3 end-to-end delays in a mobile network.

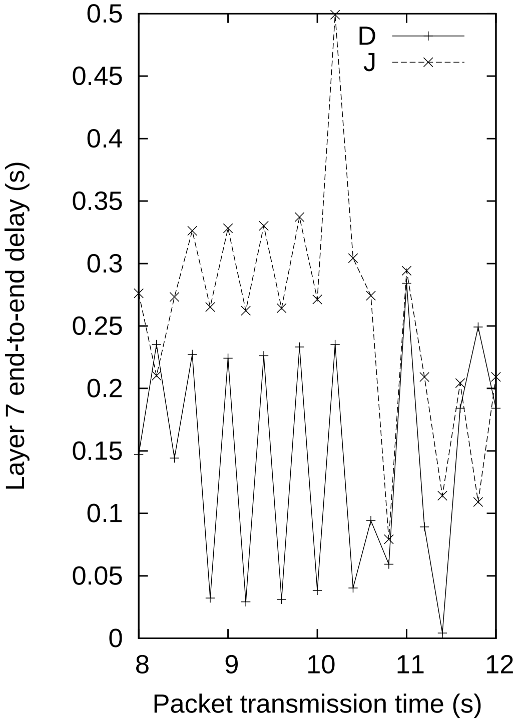

Fig. 10. Layer 7 end-to-end delays in a mobile network.