CTCP: Reliable Transport Control Protocol for Sensor Networks

# Eugenia Giancoli ^{1,2} , Filippe Jabour ^{1,2}, Aloysio Pedroza ^{2}
^{1} *CEFET-MG, Centro Federal de Educação Tecnológica de Minas Gerais*
^{2} *UFRJ - PEE/COPPE/GTA - DEL/POLI*
{*eugenia, jabour, aloysio*}*@gta.ufrj.br*

Abstract

This work presents Collaborative Transport Control Protocol (CTCP), a new transport protocol for sensor networks. It aims at providing end-to-end reliability and adapts itself to different applications through a two level mechanism of reliability variation. CTCP achieves these properties using hop-by-hop acknowledgments and a storage control algorithm that operates at each node along a flow. It was observed that distributed fault recovery increases 98% the average delivery rate and that duplication of storage responsibility minimizes the message loss. Its congestion control differentiates communication losses from buffer overflow. CTCP is called collaborative because all nodes detect and act on congestion control and also because it includes distributed storage responsibility. It is scalable and independent of the underlying network layer. The protocol energy consumption overhead was calculated and discussed for two reliability levels.

1. Introduction

Sensor networks are deployed for a wide range of applications in the military, environmental, health, industrial, and office domains. These networks are characterized by long or variable delays, frequent breakdown of connectivity, high error rates and limited resources. Each application has different characteristics and requirements of data types, transmission rates and reliability. Existing transport layer protocols for sensor networks are either tailored for certain applications, or assume that nodes employ a particular network layer or MAC layer protocol. As a result, these protocols cannot be applied across many sensor network deployments. Thus, it is necessary to design a transport protocol that can support multiple applications on the same network, provide controlled variable reliability, address congestion, reduce losses and support frequent disconnections [7]. This paper explores design decisions regarding one transport protocol that supports reliable data delivery in wireless sensor networks. We examined various transport protocols for sensor networks and propose a new solution to increase the reliability of data delivery by developing a Collaborative Transport Control Protocol (CTCP).

In what follows, we discuss technical considerations that underpin the design of CTCP, section 2. Section 3 describes related work. CTCP will be specified in section 4 and we show simulation results in section 5. In the section 6, we mention conclusions and further work.

2. Design Considerations

The sensor nodes are usually deployed over a difficult access area, forming a wireless sensor network. Each sensor is capable of collecting and routing data to sink, which is connected by cable to base station. The data is routed through a multiple hop architecture. This paper considers that base station and nodes use the protocol stack suggested by [1]. The protocol stack consists of the application, transport, network, data link and physical layers.

This paper proposes solutions to create a reliable data transfer on WSNs and therefore focuses its efforts on transport layer, whose primary goal is to provide a reliable, efficient, and economical service for application layer processes. The independence of physical network and underlying layers is a feature of the protocol presented here. Note that CTCP works with any routing protocol, but assumes that one must be present.

An important issue to be considered is the trade-off between the implementation of hop-by-hop reliability into transport or link layer. According to [13], even if the MAC protocol (the link layer) can recover lost packets through bit-error mechanism, it does not usually have ways to recover the packets discarded by buffer overflow. Therefore, WSN transport protocol must have mechanisms to recover packets loss.

Moreover, [13] cites that end-to-end solutions are quite simple and robust, but inject many packets on the network. However, the hop-by-hop solutions can quickly weaken congestion and bring fewer on-going packets in networks, while it needs to change the behavior of each node on its way from source to destination. Therefore, having fewer on-going packets can result in saved energy.

In traditional networks, the intermediate nodes are just level 3 routers. In sensor networks all nodes have a transport layer which makes it possible to distribute the error recovery task. This collaboration between nodes becomes feasible because all nodes belong to the same administrative entity and want to reach the same goal.

According to [4], [6], [13], the basic requirements for a generic transport layer of sensor networks are:

Heterogeneity: Packets from a sensor constitute its application data flow. The transport layer protocol should support multiple heterogeneous applications data flows in the same network.

Reliability: Every application may require different reliability schemes. For example, in a military environment, data transmitted by sensor nodes must always reach the base station. In sensor reprogramming, data needs to reach all sensor nodes. However, in temperature monitoring, a few packets may be lost without causing major damage. The transport protocol should exploit this variable reliability model and save network resources.

Congestion Control: Packages from all network nodes converge to nodes located near the base station. These nodes forward more packets and hence increase the possibility of congestion close to the base station. High data rates, sudden bursts of data and collisions are other reasons for congestion in sensor networks.

Flow Control: Hop-by-hop protocols are able to control every link separately along the route, which is a clear advantage over end-to-end protocols, especially in mobile networks [4].

Initial Connection Simplification: Transport protocols to WSN should simplify the process of connection opening or use protocols without connection to begin transmitting data as soon as possible to ensure minimum delay.

Packet loss is usual under WSN due to bad quality of wireless channels, sensors failure and congestion. WSNs must guarantee certain reliability in packet- or application-level through loss recovery in order to abstract correct information. Anyway, we first need to detect packet loss in order to correctly recover missing packets. After detecting packet loss, ACK and/or NACK (and their variant) can be used to recover missing packets based on an end-to-end or hop-by-hop approach. Likewise in congestion control, there is still a trade-off between end-to-end and hop-by-hop approach, which should be thought over. When designing transport control protocols for wireless sensor networks, we must consider energy conservation as well. Intuitively, if there are few on-going packets and few re-transmissions, energy can be saved. Effective congestion control can result in fewer on-going packets and effective loss recovery approach can result in fewer re-transmissions. So, congestion control and reliability guarantee can additionally save energy in a wireless sensor network. In short, the problem of transport control protocols for sensor networks is how to effectively control congestion and how to guarantee reliability while conserving as more energy as possible simultaneously.

This paper proposes CTCP. It is a collaborative transport control protocol based on known mechanisms of packets acknowledgments (ACK) and timeout. The two levels of reliability ensure CTCP flexibility and adapt to different applications types. It aims to support connection interrupt without data loss. Even when a node receives the data and fails before forwarding them, the protocol is able to recover the loss. Our work uses hop-by-hop acknowledgments, proved efficient in [11], but provides immediate node buffer release. Buffer release increases forwarding capacity and avoids congestion. Moreover, CTCP has a control congestion service able to avoid losses related to buffer overflow. It was designed to work with any underlying network layer.

3. Related Research

The first protocol analyzed was the TCP [2]. The TCP protocol stack was theoretically designed to operate independently of lower layers technologies. The TCP/IP networks should operate in wire reliable networks, wireless networks, satellite networks, optical networks etc. However, the TCP/IP mechanisms are based on some conventional wire networks. First, the existence of end-to-end connectivity between source and destination, throughout the session communication period. Second, the small delays communication (in the order of milliseconds). Third, low error rates. Fourth, effective mechanisms for errors repair retransmission. Fifth, symmetrical bidirectional supports data rates.

Thus, TCP is not suitable for sensor networks. These networks are characterized by long or variables delays, frequent breakdown of connectivity, high error rates and limited resources. The stack TCP/IP presents a bad performance in these networks.

Reliable Multi-Segment Transport (RMST) [11] is designed to operate in conjunction with the Direct Diffusion Protocol [3]. Thus, there is a dependency on the network layer. The RMST is a protocol based on negative acknowledgment (NACK). It works with or without cache. When the cache is enabled, the nodes intermediates store the data fragments, which may cause the buffer overflow. RMST does not guarantee the reliability in the case of a node failure after receiving and before transmitting the data fragments. Moreover, the RMST does not address the congestion in sensor networks.

Pump Slowly, Quick Fetch (PSFQ) [12] aims to reprogram the sensor networks nodes. It provides a reliable broadcast of data from base station to sensor nodes. However, it has high energy consumption, since the reliability is achieved through the increase of retransmits in the network.

Sensor Transmission Control Protocol (STCP) [6] is designed to be a generic protocol. Its means that it adapts to any application type and work with any underlying network layer. It has congestion control because packets from all nodes in the network converge to nodes located near the base station. These nodes forward more packets and there is a possibility of congestion close to the base station. Its reliability level focuses on energy economy. However, almost all its controls depends on the base station, which has adequate energy, memory and processing power. It is strange, moreover, that the reliability level required by the application is controlled by the node that originated the information. The global knowledge of an application and a network would be needed to make such a decision. Network sensor synchronization time is required to save energy in applications that have continuous data flow. In CTCP, reliability is defined by base-station and the network clock synchronization is not needed.

4. Protocol Specification

Two methods were used for protocol modeling and formal analysis. A mathematical formalism called Predicate-action net [10] and a specifying language called CCS (*The Calculus of Communicating Systems*), of Robin Milner [9]. These are formal specification methods that allow systems development with a minimum ambiguity, through a well-defined syntax and semantics. A formal specification of our protocol provides an analysis of certain properties, such as lack of deadlocks blocks, correct timing and message sequence, and the verification of specification consistency. The new contributions of CTCP are:

- Delivery of all segments to base station application layer, even in the presence of nodes failures and frequent disconnections.
- Two reliability profiles to save energy.
- Ability to differentiate congestion loss from transmission error loss.
- Congestion control through the interruption of packets forwards (or immediate release of packets forwards)
- Independence from underlying layers.

A. Hop-by-Hop connection open and close

Before starting data transmission, a packet (ABR) is sent hop by hop from source to base station. This packet informs the data flow identifier and the first sequence number to the base station. When the base station receives this packet, it reserves the buffers, initializes required variables and sends a response (RSP) message to source node. RSP header specifies the reliability level required by the application to which the flow belongs. It also specifies the connection identifier (ID), which is formed by the concatenation of the node ID plus first sequence number generated by this node.

The source node will be able to initiate the data transmission after it receives the response (RSP). The protocol was designed to be as generic as possible. So, CTCP was concerned to establish packet formats compatible with the majority of underlying networks. Thus, it was necessary to consider that wireless communication and especially the sensor networks have a particular difficulty to transmit packets larger than the network MTU. Despite some protocols, such as 802.11, have fragmentation and reassembly, there are limits for packets size that an entity can fragment and guarantee delivery [11]. Therefore, the sensors that use the CTCP will be pre-configured with the MSS (Maximum Segment Size) permitted by underlying network layers. This variable does not need to be negotiated during connection opening phase.

CTCP uses reliability level 1 until the RSP reaches source node with the real readability determined by base station.

When source node application layer ends its work, it sends a packet (CLO) to base station requesting the release of connection. The base station then releases buffers and variables.

B. Controllably Reliable Delivery

Each application has different requirements of reliability. Thus, in order to specify the required reliability, it must know the application and its goals. Moreover, the reliability level, which is set at the connection opening phase, may be dynamically changed, when it is necessary, by the user who interacts with the base station. This need can be represented, for example, by the nodes energy exhaustion. In this case it may be more interesting to work with less reliability to maximize useful network life. When the user changes readability level, an RSP packet is sent to source node with a new level.

Once the user sets the required application reliability level, network nodes can act in two different ways, described below.

Reliability Level 1

This reliability level is intended to save energy by reducing transmissions. It has low cost of buffers and applies mainly to applications that have some data redundancy or that can tolerate losses. After receiving a packet from node A, node B stores a copy in its buffer, triggers the timer, forwards the packet and sends an acknowledgment (ACK) to node A. Node B is temporarily responsible for packet delivery to base station. This process happens repeatedly, through the network layer stipulated route, until the base station receives data packet and sends an ACK to the preceding node. Any node that receives an ACK may discard the sent packet, saving space in its buffer. This is represented in Figure 1.

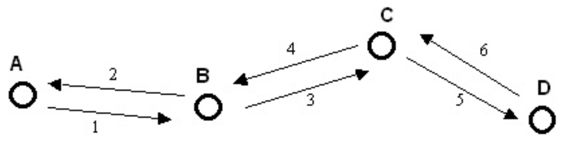

Fig. 1: Level 1 messages exchange order

Note that when the intermediary node assumes the responsibility to deliver the packet, it must keep a copy of this packet in buffer until it receives the ACK. The absence of an ACK generates a forwarding node clock fired and retransmission of the unrecognized packet.

Consider, however, the following situation: node A, source, sends data to node B. B, receives data, stores, forwards the packet and sends ACK to node A. Then, the packet does not reach node C and at this moment, node B fails. Therefore, the data that were under node B responsibility will not be retransmited to base station and node A will not notice this flaw. This situation can only be resolved by increasing the reliability level.

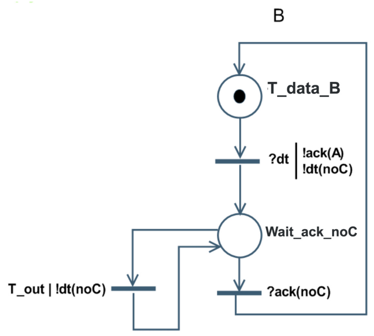

Fig. 2: Two nodes level 1 communication

In what follows, we can see reliability level 1 formally specified in CCS [9]. Consider that $B!dt$ means the sent of dt to B, and $A?dt$ means the reception of dt sent by A.

$Level\_1 = A|||B|||Timer$

$A = !dt.Wait\_ack\_A$
$Wait\_ack\_A = ?ack(A).A$
$B = ?dt.!ack.!dt(C).!starttimer.Wait\_ack\_noC$
$Wait\_ack\_C = ?ack(C).!reset\_timer.B+$
$?t\_out.!dt(C).!start\_timer.Wait\_ack\_C$
$Timer = ?start\_timer.(!t\_out.Timer+$
$?reset\_timer.Timer)$

Reliability Level 2

Node A sends data to node B and waits to receive the *double ACK*. The double ACK is generated as follows: B receives data from A and sends A the first ACK. B sends the data to C that sends B the first ACK. When B receives the first ACK from C, it forwards the second ACK to A. Immediately, A discards the data kept in buffer. All nodes repeat this process, successively, until the data reaches base station. The base station should send two ACKs for the ultimate node. Figure 3 represents this message exchange.

If node B fails before delivering data to node C, node A will not receive the second ACK (double ACK) and will retransmit the packet. Note that starting from the assumption that failures of nodes are monitored by routing algorithms, these algorithms will be responsible for reconstructing the route in the presence of one node failure or a set of them.

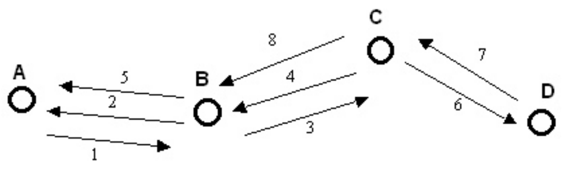

Fig. 3: Level 2 messages exchange order

The protocol described above enables the message to reach its destination with a greater probability, since one node failure in the path does not interrupt data delivery. Figure 4 shows a formal specification of level 2.

Below, reliability level 2 formally specified in CCS. Again, consider that $B!dt$ means the sent of dt to B, and $A?dt$ means the reception of dt sent by A. More details in [9].

$Level\_2 = A|||B|||Timer$

$A = B!dt.Wait\_ack1\_A$
$Wait\_ack1\_A = B?ack.Timer!start.Wait\_ack2\_A$
$Wait\_ack2\_A = B?ack.Timer!reset.A+$
$Timer?t\_out.B!dt.Timer!start.Wait\_ack2\_A$
$B = A?dt.C!dt.A!ack.Wait\_ack1\_B$
$Wait\_ack1\_B =$
$C?ack.A!ack.Timer!start.Wait\_ack2\_B$
$Wait\_ack2\_B = C?ack.Timer!reset.B+$
$timer?t\_out.C!dt.timer.start.Wait\_ack2\_B$
$Timer = ?start(!t\_out.timer+?reset.timer)$

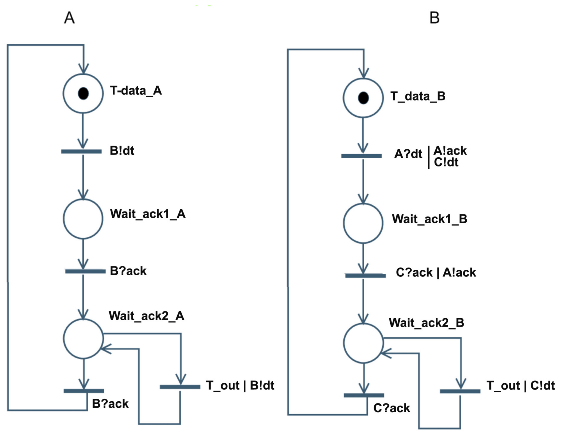

Fig. 4: Two nodes level 2 communication

C. Detection and congestion control

Discarding packets on sensor networks is not an ideal solution. Therefore, it was necessary to consider other options. In sensor networks, the packet loss usually refers to transmission errors and not to the network congestion. Any packet loss triggers the controlling congestion mechanism and transmission rate reduces without necessity. Thus, it is necessary to implement a control congestion mechanism that considers the difference between a transmission error packet loss and the buffer overflow.

In this proposal, congestion control is implemented through the participation of all nodes. These nodes manage congestion using signaling messages. Thus, a node refuses to receive more packets, if their buffer is up the threshold. So when buffer nodes reach threshold T, the node broadcasts a packet flag *(STOP)* for all its neighbours. This signaling packet warns that packets can no longer be sent to it as its buffer is overflowing. That would reduce the neighbours' transmission rate. This reduction in transmission rate may be flooded across the network, depending on the congestion level. When nodes achieve their empty buffer, or when they are below threshold T, a new packet flag *(START)* is sent to release the forwarding of new packets. Each node in the WSN maintains a table with its active neighbours ID and the corresponding connection identifiers. After sending the START package, these neighbours start transmitting their data packages again. If this does not happen, it is possible to identify those that did not receive the START package. Thus, a new START package will be sent in unicast for those neighbors that did not resumed its transmissions. The computation of threshold T must consider that, when a node i sends the STOP package in broadcast, one or more neighbours may not receive it due to the increased probability of transmission collisions during congestion. Noting that one or more neighbours are still sending packages, node i sends the STOP package in unicast only for these nodes. So, it is possible, through the mechanism described in this section, to conclude that all packet losses will be due to errors in transmission rather than congestion.

5. Simulation Results

The major functions of transport control protocols for wireless sensor networks are congestion control, reliability guarantee, and energy conservation that can be passively realized by congestion control and reliability guarantee [13]. Thus, to understand the behavior of CTCP reliable delivery mechanism, we performed extensive simulations using TinyOS Simulator (TOSSIM) [8]. TOSSIM's network is a directed graph, in which each vertex is a node, and each edge has a bit error probability. Each edge (a, b) in the graph means a's signal can be heard by b. Every edge has a value in the range (0,1), representing the probability a bit sent by a will be corrupted when b hears it. For example, a value of 0.01 means each bit transmitted has a 1% chance of being flipped, while 1.0 means every bit will be flipped, and 0.0 means bits will be transmitted without error. Each bit is considered independently.

We adjusted CTCP parameters based on empirical observations. We simulated networks with 25 sensor nodes randomly distributed in a 50m X 50m square sensor field. The transmission range of the nodes was set to 50 feet (15.24 m). Combined with the bit error rate, this means each mote transmits its signal in a disc of radius 50 feet, with the bit error rate increasing with distance from the center. The data packet length is 216 bits while acknowledgement packet length is 72 bits. To route packets from sensor nodes to base station, we use the TOSSIM's standard routing protocol, called MintRoute. MintRoute is a proactive routing protocol in which nodes send periodic routing messages to declare their local states. The simulator had a default CSMA MAC layer protocol for channel access. Simulations were executed for 1000 simulator seconds. Only one source node generated one measured, and consequently sent one packet, every 50 simulator seconds.

To evaluate the performance of CTCP, we used the following metrics:

- Fraction of packets successfully received.
- Energy consumption.

A. *Fraction of packets successfully received*

Reliability is measured as the fraction of transmitted packets successfully received in the presence of packet error rates and node's failure. Figure 5 shows the average delivery rate in three different situations. At first, the application runs directly over network layer, without a transport protocol. In second situation, CTCP was introduced and set to reliability level 1. In the last simulation series, CTCP was adjusted to reliability level 2.

From figure 5, we observe that application delivery rate is 29,82% without any transport protocol. The same application, runing with CTCP reliability level 1, delivers the double, 61,11%. Finally, runing CTCP with reliability level 2, the delivery rate is 98,15%. Note that reliability level 2 is appropriate to applications where reliability is essential. Level 1, doubles the delivery rate, being an intermediary option, while "no reliability" may be used with loss tolerant applications.

Simulation logs showed that the protocol with reliability level 2 would achieve 100% of delivery rate with a few more

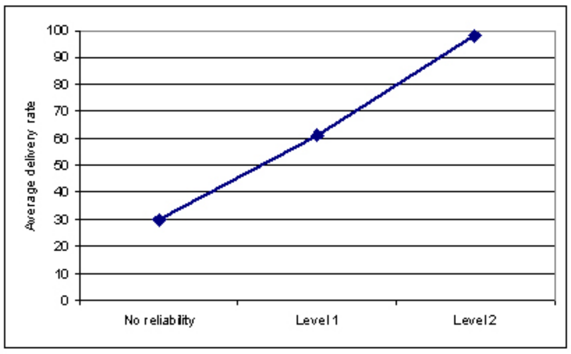

Fig. 5: Average delivery rate.

seconds of execution. Thus, we see that level 2 would offer 100% of guaranteed delivery. In this case, the amount of packets not delivered (1.85%) is related to a higher latency and not to permanent losses.

Increasing the number of sources into the network (with more sensor readings per time unit), we will have a higher probability of congestion. In this case, it will be very important the use of a congestion control mechanism (section 4-C). Congestion control reduces the number of retransmissions due to buffer overflow, thus reducing the average latency in the network. The development of this mechanism (section 4-C) is ongoing.

B. *Energy Consumption*

In section 5-A, we saw that CTCP is capable of providing high levels of reliability. Since energy is a crucial constraint in wireless sensor networks, it is necessary to measure the energy consumption overhead.

For three situations described above (no reliability, level 1 and level 2), the simulation counts the total number of data packets and acknowledgment packets (if any).

The size of data packets is 216 bits, including the application message (the sensor reading), CTCP, network layer and MAC layer headers.

The acknowledgment packet is sent directly to the API of layer 2. It is destinated to the neighbor's MAC address. ACKs packets are just 72 bits long.

We assumed that radio dissipates 50 nJ/bit to run the transmitter or receiver circuit and 0.1nJ/(bit.m2) for the transmitter amplifier to achieve an acceptable SNR [5].

Based on values above, we calculated application and CTCP layers energy consumption of each node. The sum of the individual consumption is the network consumption. Figure 6 shows the application and CTCP layers energy spent by the network. All values are in millijoules (mJ).

As expected, figure 6 shows a higher energy consumption to provide a higher reliability level. After 1000 seconds, the total amount of energy spent in the network is 12.07 mJ level 2, 6.29 mJ for level 1 and 2.19 mJ without reliability. Observe that low values (in order of millijoules) occurs because there is only one source and just twenty reading messages are generated

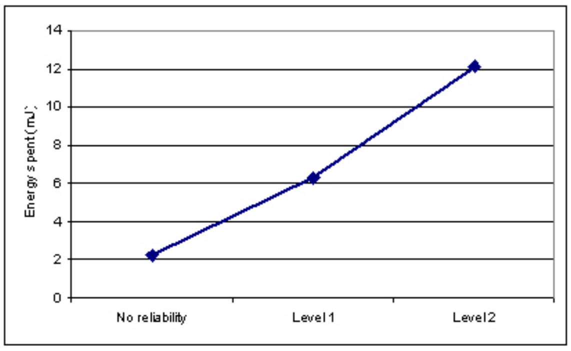

Fig. 6: Energy consumption for a 25 nodes network without node failure

for each simulation. That is acceptable, as the main goal of simulation is to compare the three reliability levels.

The energy spent in providing reliability level 2 is 1.91 times the energy spent for level 1 and 5.51 times the amount spent without reliability.

The energy consumed by underlying layers (network, link and physical) is the same in the three cases studied. So, the proportional overhead imposed by CTCP becomes less significant.

We used MintRoute as routing protocol. The original cost metric in MintRoute combines hop count and link quality. To compute link quality, a node snoops on the packets sent by each neighbor, and checks the sequence number of the packets. A node determines the link quality to a neighbor by monitoring the ratio of packets received from that neighbor to the number of packets sent by that neighbor. In order to maintain routing tables, MintRoute changes control messages every two seconds. There is still the CSMA MAC layer protocol for channel access that represents an additional amount of energy consumption. In this application, the node's total energy consumption is much higher than the total of transport and application layers consumption. Thus, the CTCP consumption represents a small part of the total energy consumption and its use is feasible for applications that require delivery reliability.

6. Conclusions and future work

In this paper, we present the Collaborative Transport Control Protocol for Sensors Network. This protocol proposes reliable messages delivery between a source node and its respective base station. It aims to detect and control congestion through the differentiation between transmission error losses and buffer overflow. In case of losses, it offers recovery in two reliability levels and, in case of congestion, provides explicit signaling to break and resume data transmission.

CTCP delivers 61,11% when working with reliability level 1 and 98,15% with reliability level 2. The protocol presents a significant increase in delivery reliability and decrease in definitive message loss. It is achieved by distributing responsibility of temporary message storage between two adjacent nodes (reliability level 2). Moreover, distributed message storage demonstrates the protocol robustness during periods of disconnection, prior to an ACK confirmation.

Considering the energy consumed by underlying layers (network, link and physical), the proportional overhead imposed by CTCP keeps the protocol feasible and usable.

CTCP makes no restriction on the protocol lower levels to be used.

The base station takes the decisions that depend on the application requirements knowledge (reliability required) and control parameters with central characteristics. By the other hand, the functions of congestion control, reliability implementation and connections openness are distributed by the WSN.

In future work, we will investigate the effect of increasing the number of hops in the generation of multiple ACKs in reliability model level 2. We intend to evaluate the protocol performance with multiple sources(multiple flows), implement and test the congestion control mechanism.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] M. Allman, V. Paxson, and W. Stevens, "Tcp congestion control," RFC 2581 (Proposed Standard), apr 1999, updated by RFC 3390. [Online]. Available: http://www.ietf.org/rfc/rfc2581.txt

[3] D. E. C. Intanagonwiwat, RC. Govindan, "Direct diffusion: A scalable and robust communication paradigm for sensor networks," in *In Proceedings of the Sixth Annual ACM International Conference on Mobile Computing and Networking*, Aug 2000, pp. 56–67.

[4] S. Heimlicher, R. Baumann, M. May, and B. Plattner, "Saft: Reliable transport in mobile networks." Vancouver B.C., Canada: IEEE MASS 2006, Oct. 2006, pp. 477–480.

[5] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*. Washington, DC, USA: IEEE Computer Society, 2000, p. 8020.

[6] Y. G. Iyer, S. Gandham, and S. Venkatesan, "Stcp: A generic transport layer protocol for wireless sensor networks," in *Proceedings of 14th International Conference on Computer Communications and Networks*, Houston, TX, USA, 2005, pp. 449–454.

[7] S. Kim, R. Fonseca, P. Dutta, A. Tavakoli, D. Culler, P. Levis, S. Shenker, and I. Stoica, "Flush: a reliable bulk transport protocol for multihop wireless networks," in *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*. ACM, 2007, pp. 351–365.

[8] P. Levis, N. Lee, M. Welsh, and D. Culler, "Abstract tossim: Accurate and scalable simulation of entire tinyos applications," 2003.

[9] R. Milner, *A Calculus of Communicating Systems*. Springer Verlag, 1980.

[10] M. Nielsen, G. Plotkin, , and G. Winskel, *Petri Nets, Event Structures and Domains, Part I, Vol. 13*. Theoretical Computer Science, 1981.

[11] F. Stann and J. Heidemann, "Rmst: Reliable data transport in sensor networks," in *Proceedings of the First International Workshop on Sensor Net Protocols and Applications*, Anchorage, Alaska, USA, 2003, pp. 102–112.

[12] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy, "Pump-slowly, fetch-quickly (psfq): A reliable transport protocol for sensor networks," in *IEEE Journal on Selected Areas in Communications, Vol. 23, No. 4*, Atlanta, Georgia, USA, 2005, pp. 862–872.

[13] C. Wang, K. Sohraby, B. Li, and W. Tang, "Issues of transport control protocols for wireless sensor networks," in *Communications, Circuits and Systems, 2005. Proceedings. 2005 International Conference on*, vol. 1, Arkansas Univ., Fayetteville, AR, USA, May 2005, pp. 422–426.