# SNOMC: An Overlay Multicast Protocol for Wireless Sensor Networks

Gerald Wagenknecht, Markus Anwander, Torsten Braun
Institute for Computer Science and Applied Mathematics
University of Bern
Neubrückstrasse 10, CH-3012 Bern, Switzerland
wagen|anwander|braun@iam.unibe.ch

***Abstract*—Using multicast communication in Wireless Sensor Networks (WSNs) is an efficient way to disseminate the same data (from one sender) to multiple receivers, e.g., transmitting code updates to a group of sensor nodes. Due to the nature of code update traffic a multicast protocol has to support bulky traffic and end-to-end reliability. We are interested in an energy-efficient multicast protocol due to the limited resources of wireless sensor nodes. Current data dissemination schemes do not fulfill the above requirements. In order to close the gap, we designed and implemented the SNOMC (Sensor Node Overlay Multicast) protocol. It is an overlay multicast protocol, which supports reliable, time-efficient, and energy-efficient data dissemination of bulky data from one sender to many receivers. To ensure end-to-end reliability, SNOMC uses a NACK-based reliability mechanism with different caching strategies.**

## I. INTRODUCTION

Wireless Sensor Networks (WSN) are sets of wireless sensor nodes, on which different applications are running to support applications such as event detection, localization, tracking, and monitoring. Independent of the performed task an application should be configured and continuously updated throughout the lifetime of the sensor nodes. Configuration and updating should be done over the air [1] and then the traffic for sensor data retrieval differs considerably from the predominant traffic pattern in WSNs, which is multipoint-to-point communication. On the contrary, during a code update point-to-multipoint communication takes place, e.g., the code update needs to reach multiple (or all) sensor nodes. The traffic nature of the updates is bulky in this case. Since node updates are rather crucial, transmissions should be reliable.

Distributing data from one sender node to many receivers can be done in different ways. The simplest one is flooding where data is transmitted using broadcast communication. Flooding, however, is inherently very inefficient, energy-consuming, and unreliable. Another way is to rely on multiple unicast connections between sender and any of the desired receivers. In the context of WSNs, however, redundancy leads to to higher probability of collisions, which can cause long transmission times. Current code dissemination schemes such as TinyCubus [2], Deluge [3], and Trickle [4] distribute code using broadcast and without any reliability mechanism. This leads to inefficient and unreliable code distribution. A more efficient distribution scheme, which better fits the requirements set by configuration and code updating, is multicast. Multicast is able to propagate data from a single sender to many receivers by affecting smaller numbers of sensor nodes in the network. It is easily extendable with any kind of reliability mechanism and it has better support for bulky traffic patterns. Thus, multicast communication fits very well for any code update task in WSNs. In the literature several solutions for multicast in WSNs are proposed, but they mainly deal with multicast routing and not with reliable and efficient data distribution. Currently, to our knowledge, there is not a single multicast protocol able to meet the combined set of requirements for reliability and efficiency (in both time and energy consumption) for bulky traffic patterns.

The question arises how can we design such a multicast protocol in a WSN. How can we support end-to-end reliability (necessary for code updates) while still keeping energy consumption and delays low? In the design process several choices need to be made including how bulky traffic is best propagated and what is an appropriate underlying MAC protocol. Further, we would like the multicast communication to be IP-based in order to access the WSN via the Internet [5].

In this paper, we propose the SNOMC (Sensor Node Overlay Multicast) protocol, which supports the reliable transfer of bulk data in a WSN from one sender to multiple receivers. It is designed as an overlay multicast protocol on top of the $\mu$IP stack from Contiki [6]. The main advantages of SNOMC compared to previously proposed solutions are: (1) its time- and energy-efficient manner of data distribution and (2) the simple NACK-based mechanism supporting end-to-end reliability.

## II. MULTICAST IN WIRELESS SENSOR NETWORKS

Wireless sensor networks are often limited in energy, memory, and CPU power. They pose new challenges not existing in wired networks. Hence, porting an existing IP Multicast solution designed for wired networks to wireless sensor networks is impractical or even impossible. In wired networks, routers are handling packet replication and forwarding while clients only send and receive simple IP datagrams. To directly port IP Multicast from wired networks to WSNs we need to introduce the IP Multicast router functionality at each sensor node. This would require memory and processing power that a sensor node may lack. Even if resources are available, the lifetime of a node will be severely affected.

There are also some differences from an architectural point of view. Generally, we distinguish between two node types in WSNs, namely branching nodes and forwarding nodes. Branching nodes participate in the multicast communication, duplicate packets, and store state information about receivers and/or about other branching nodes. Forwarding nodes have less or no information about the multicast state and just forward the multicast data from one neighbor to the next one. An example of a WSN topology with multicast is shown in Fig. 1. Sender, branching nodes, forwarding nodes, and three receivers (group members) are indicated.

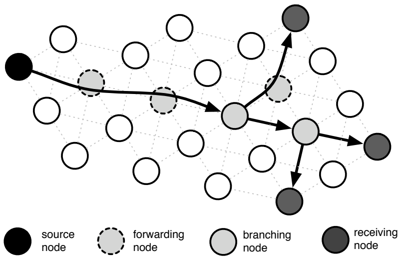

Fig. 1. Roles of the nodes in a multicast scenario.

Moreover, wireless communication links have shown to be more error prone compared to wired links. This raises additional concerns about medium availability, collisions, and reliability. A multicast solution for WSNs needs to address all these issues and in particular reliability, since code updates or other critical tasks could be solved efficiently by multicast.

## III. RELATED WORK

There are different data dissemination schemes frequently used in WSNs. One such scheme is Directed Diffusion [7] that can be used for both multipoint-to-point and point-to-multipoint communications. Broadcast-based data dissemination schemes are less complex but are also much more inefficient. Pump Slowly, Fetch Quickly (PSFQ) [8] is a reliable transport protocol and supports broadcast based code distribution. It transmits data segments relatively slow ('pump slowly') and uses an aggressive NACK mechanism to fetch missed data segments ('fetch quickly'). The aggressive NACK mechanism can lead to congestion in the WSN. Multipoint Relaying (MPR) [9] protocol is used to optimize broadcasting. Each node in MPR requires knowledge of its two-hop neighborhood. Based on this information, subsets of one-hop neighbors are forced to rebroadcast given data packets. These nodes are called multipoint relays. The multipoint relays are chosen according to its connectivity to other nodes. MPR does not support any reliability mechanism. TinyCubus [2] is an adaptive cross-layer framework for sensor networks (also broadcast-based). It deploys a role-based code distribution algorithm that uses cross-layer information, such as role assignments, in order to decrease the number of messages needed to distribute code to specific nodes. It is assumed that roles are assigned before code deployment and that the connectivity of the network for a given role can be determined up-front.

In [10] a multicast protocol called BAM (Branch Aggregation Multicast) is presented, which supports single-hop link-layer multicast and multi-hop multicast by doing branch aggregation. Another multicast protocol for sensor nodes, with support of node mobility, is VLM$^2$ (Very Lightweight Mobile Multicast) [11]. VLM$^2$ provides multicast from a base station to sensor nodes and unicast from sensor nodes to a base station. In [12] the authors present an effective all-in-one solution for unicasting, anycasting, and multicasting in wireless sensor and mesh networks. RBMulticast [13] is a stateless, receiver-based multicast protocol, which exploits knowledge on the geographic locations of nodes to reduce costly state maintenance. The authors of [14] adapt ADMR (Adaptive Demand-driven Multicast Routing), a multicast protocol for mobile ad-hoc networks, on a real wireless sensor node (MICAz). They show that protocol adaptation is not a trivial task and a number of problems have to be solved. At the same time, the authors of [15] analyze IP Multicast and show that it is possible to use it in WSNs. Further, there are several multicast solutions for WSNs based on the geographical position of the sensor nodes [16], [17], [18]. All of these protocols support neither end-to-end reliability nor energy-saving mechanisms.

## IV. PROTOCOL DESIGN

We aim to design a protocol that supports multicast in WSNs in an efficient and energy-saving way. The protocol should support bulky traffic, which is characterized by data arrivals in bursts of, e.g., 1000 bytes. A typical example of bulky traffic is code update when the update originates from one sender and is intended to reach many receivers. Since code updates are crucial for the operation of the WSN, communication must be reliable. Additionally, we also want that the multicast protocol runs on top of IP in order to access the WSN via the Internet.

There are several approaches towards the design of a reliability multicast solution for WSNs. On the one hand, we can distinguish between overlay multicast [19] and IP Multicast, which differ in the protocol layer, on which they are implemented. IP Multicast is implemented on the network layer. Overlay multicast is implemented on the application layer. Furthermore, we can choose between a sender-driven and a receiver-driven formation of the multicast group. In the sender-driven approach, the sender decides on the receiving nodes (the participants in the multicast group) while in the receiver-driven approach the receiving nodes decide themselves whether they want to receive data or not.

We use UDP as transport protocol. It does not support any reliability since it is stateless, but it benefits from low complexity. Acknowledgments on the application layer can be positive or negative and inform about the reception status of messages. Caching is a convenient way to have content closer to the requesting side and hence decreases delays. There are three possibilities for where to cache data: on sender nodes, on branching nodes or on all intermediate nodes (forwarding and branching).

The SNOMC (Sensor Node Overlay Multicast) protocol applies an overlay multicast approach able to operate in both sender-driven mode and receiver-driven mode. SNOMC uses UDP as transport protocol and to ensure reliability SNOMC uses a simple NACK-based mechanism with all three caching modes.

In the following we describe the technical details of SNOMC operations, including joining a multicast group and transmitting to the group. Each of the three caching possibilities is considered. A distribution tree can be built only after the receiver nodes have joined a multicast group. This can happen in two different ways, namely, sender-driven or receiver-driven. We begin with the description of the sender-driven approach. In a sender-driven approach the sender decides, which nodes should be in the multicast group as receivers. The join procedure is shown in Fig. 2. First, the sender S creates a join message (cf. Fig. 3(a)), which contains the list of receivers, the group id, and the sender address. Next, the sender transmits this join message to its next-hop neighbor **(1)**. The next-hop becomes a forwarding node F when all receivers can be reached via one next-hop neighbor **(2)**. It becomes a branching node B when the receivers are reached via different next-hop neighbors. The receiver list is split accordingly and transmitted to the respective next-hop neighbors. The branching node adds its address into the branch_id field of the join message **(3)**. When a join message reaches a receiver R it confirms the join by transmitting a join_ack message (cf. Fig. 3(a)) back to the last branching node, written in the branch_id of the join message. The branching node waits for the join_ack messages of all subordinate receivers (R1 and R2), combines these messages into one, puts its own address as branching node into the message and transmits it back to the sender node (or to the next branching node upstream on the path towards the sender) **(4)**. The sender node knows all branching nodes and can later establish an overlay connection to them.

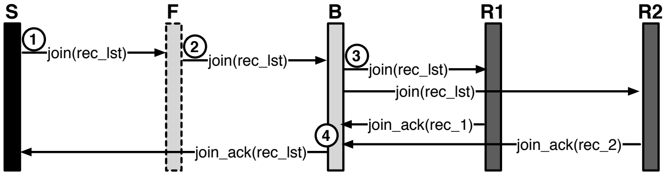

Fig. 2. SNOMC: Joining phase, sender driven mode.

(a) join and join_ack messages

(b) data message (c) nack message (d) data_ack message

Fig. 3. SNOMC: messages.

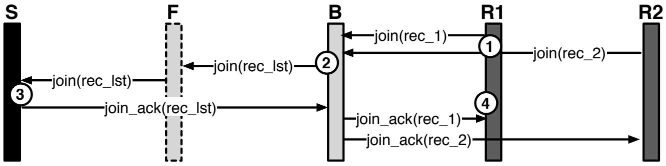

Fig. 4. SNOMC: Joining phase, receiver driven mode.

In a receiver-driven approach the receivers themselves decide whether they want to be in a multicast group. The join procedure is shown in Fig. 4. Each receiver transmits a join message to the next upstream neighbor into the direction towards the sender according to the routing table **(1)**. The neighbor node collects all incoming join messages and becomes a branching node or a forwarding node. If the node is a branching node, it adds its own identity to the appropriate field in the join message to inform the sender node about its role as branching node. Afterwards, the node transmits the message further to the sender node **(2)**. The sender node collects all join messages and creates a join_ack message with the receiver list, its own address as sender_id and the collected branch_id. The join_ack message is transmitted towards the receivers **(3)**. The branching node splits the join_ack message and transmits the messages to the according receivers **(4)**.

Propagating data from sender to receivers is done using the overlay connections established as result of the join procedure. The chosen caching strategy causes the establishment of the overlay connections. If data is cached on every intermediate node (sender, forwarding, and branching nodes) the overlay connections are established between them. If data is cached on sender or branching nodes, the overlay connections are established between sender, branching nodes, and receivers.

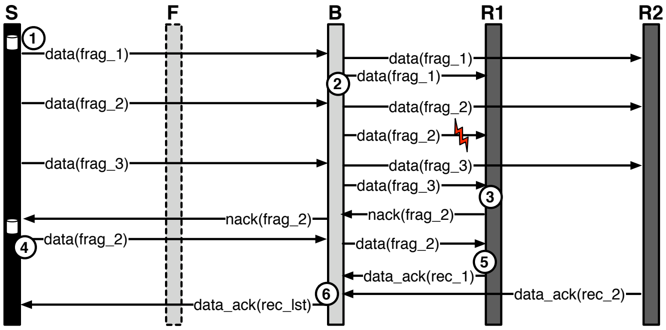

Fig. 5. SNOMC: Transmission phase, caching on sender node.

The caching strategy on the sender node is depicted in Fig. 5. The sender node fragments the data and caches them. Afterwards it transmits data messages (cf. Fig. 3(b)) to the next branching node using the overlay connection **(1)**. The branching node duplicates the data message and transmits the duplicates to the receivers or to the next branching nodes respectively **(2)**. Eventually, a receiver collects all fragments. If a fragment gets lost the receiver detects a gap in the fragment sequence and requests the missing fragment. The receiver generates a nack message (cf. Fig. 3(c)), which is sent towards the sender **(3)**. Since the sender node has the fragment in the cache, it retransmits it towards the requesting receiver **(4)**. If the receiver finally gets all fragments successfully, it confirms this with a data_ack message (cf. Fig. 3(d)) **(5)**. The branching nodes accumulate all incoming data_ack messages into a combined data_ack message towards the sender **(6)**.

If data is cached not only at the sender node, but also at the branching node the protocol operation changes slightly as shown in Fig. 6. The sender node again fragments data, caches them and transmits them to the next branching node **(1)**. The branching node duplicates the data, in addition it caches them and transmits them to the according receivers **(2)**. The benefit of this additional caching is that if a receiver detects a fragment as lost it can request it directly from the branching node **(3)**. The branching node will then read the fragment from the cache and retransmit it to the according receiver **(4)**. The confirmation with a data_ack message **(5)**, **(6)** works identically as in the other caching modes.

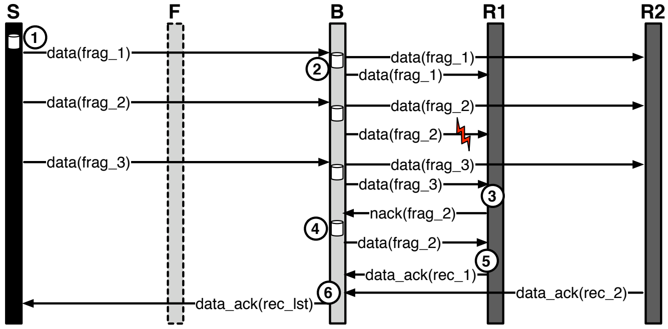

Fig. 6. SNOMC: Transmission phase, caching on branching node.

If data is cached at every intermediate node (forwarding and branching nodes), the overlay connections change such that every node knows its predecessor and successor in the distribution tree. If a receiver detects a missing fragment, it requests the fragment from its predecessor node. If the latter has the fragment cached, it retransmits it. Otherwise it forwards the nack message upstream along the distribution tree until a node is found where the fragment was cached.

## V. EVALUATION

In an initial evaluation, we compared the SNOMC protocol to other common protocols such as Flooding, MPR, TinyCubus, Directed Diffusion, UDP, and TCP in terms of transmission time and energy consumption. We implemented the protocols in the OMNeT++ simulator and run the simulations in a 6x6 grid with one sender and three receivers. To ensure end-to-end reliability we added a simple NACK-based reliability mechanism to all protocols (except TCP). Preliminary results show that SNOMC outperforms the other protocols. It transmits 1000 bytes to three receivers two times faster than UDP, four times faster than TCP, and five to ten times faster than Directed Diffusion, TinyCubus, MPR, and Flooding. Moreover, SNOMC saves three times more energy than UDP, four times more than TCP, and 15 to 20 times more than Directed Diffusion, TinyCubus, MPR, and Flooding.

## VI. CONCLUSION

We propose the Sensor Node Overlay Multicast (SNOMC) to support reliable, time-efficient, and energy-efficient dissemination of bulky code data from one sender node to many receivers. To ensure end-to-end reliability we designed and implemented a NACK-based reliability mechanism. Further, we propose different caching strategies to avoid costly end-to-end retransmissions. We conclude that SNOMC can offer a robust, high-performing solution for the efficient distribution of code updates in a WSN.

## REFERENCES

[1] G. Wagenknecht, M. Anwander, T. Braun, T. Staub, J. Matheka, S. Morgenthaler: MARWIS: A Management Architecture for Heterogeneous Wireless Sensor Networks. *WWIC'08, Tampere, Finland, May'08.*

[2] P. J. Marron, A. Lachenmann, D. Minder, J. Hähner, R. Sauter, and K. Rothermel: TinyCubus: A Flexible and Adaptive Framework for Sensor Networks. *EWSN'05, Istanbul, Turkey, Feb'05.*

[3] J. W. Hui, D. Culler: The Dynamic Behavior of a Data Dissemination Protocol for Network Programming at Scale. *SenSys'04, Baltimore, MD, USA, Nov'04.*

[4] P. Levis, N. Patel, S. Shenker, D. Culler: Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks. *NSDI'04, San Francisco, CA, USA, Mar'04.*

[5] M. Anwander, G. Wagenknecht, T. Braun: Management of Wireless Sensor Networks using TCP/IP. *IWSNE'08, Santorini, Greece, Jun'08.*

[6] A. Dunkels, B. Grönvall, T. Voigt: Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors. *EmNetS'04, Tampa, FL, USA, Nov'04.*

[7] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva: Directed Diffusion for Wireless Sensor Networking. *ACM/IEEE Transactions on Networking, 11(1):2-16, 2002.*

[8] C. Y. Wan, A. T. Campbell, L. Krishnamurthy: PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks. *WSNA'02, Atlanta, GA, USA, Sep'02.*

[9] A. Quayyum, L. Viennot, and A. Laouiti: Multipoint relaying: An Efficient Technique for Flooding in Mobile Wireless Networks. *INRIA, Sophia Antipolis, France, Tech. Rep., 2000.*

[10] A. Okura, T. Ihara, and A. Miura: BAM: Branch Aggregation Multicast for Wireless Sensor Networks. *MASS'05, Washington, DC, USA, Nov'05.*

[11] A. Sheth, B. Shucker, and R. Han: VLM2: A Very Lightweight Mobile Multicast System For Wireless Sensor Networks. *WCNC, New Orleans, LA, USA, Mar'03.*

[12] R. Flury, and R. Wattenhofer: Routing, Anycast, and Multicast for Mesh and Sensor Networks. *INFOCOM'07, Anchorage, Alaska, USA, May'07.*

[13] C. H. Feng, and W. B. Heinzelman: RBMulticast: Receiver Based Multicast for Wireless Sensor Networks. *WCNC'09, Budapest, Hungary, Apr'09.*

[14] B. Chen, K. Muniswamy-Reddy, and M. Welsh: Ad-Hoc Multicast Routing on Resource-Limited Sensor Nodes. *REALMAN'06, Florence, Italy, May'06.*

[15] J. S. Silva, T. Camilo, P. Pinto, R. Ruivo, A. Rodrigues, F. Gaudncio, F. Boavida: Multicast and IP Multicast Support in Wireless Sensor Networks. *Journal of Networks, 3(3), 19-26, 2008.*

[16] D. Koutsonikolas, S. Das, Y. C. Hu, and I. Stojmenovic: Hierarchical Geographic Multicast Routing for Wireless Sensor Networks. *SENSORCOMM'07, Valencia, Spain, Oct'07.*

[17] J. A. Sanchez, P. M. Ruiz, and I. Stojmenovic: Energy Efficient Geographic Multicast Routing for Sensor and Actuator Networks. *Computer Communications, 30(13), 2519-2531, 2007.*

[18] J. Lee, E. Lee, S. Park, S. Oh, and S. H. Kim: Consecutive Geographic Multicasting Protocol in Large-Scale Wireless Sensor Networks. *PIMRC'10, Istanbul, Turkey, Sep'10.*

[19] M. Hosseini, D. T. Ahmed, S. Shirmohammadi, N. D. Georganas A Survey of Application-Layer Multicast Protocols *Communications Surveys & Tutorials, 9(3), 58-74, 2007*