Connecting Wireless Sensornets with TCP/IP Networks

Adam Dunkels[1,2], Juan Alonso[1], Thiemo Voigt[1], Hartmut Ritter[3], Jochen Schiller[3]

[1] Swedish Institute of Computer Science, Box 1263, SE-164 29 Kista, Sweden
{adam,alonso,thiemo}@sics.se
[2] Department of Computer Science and Engineering, Mälardalen University, Box 883,
SE-721 23 Västerås, Sweden.
[3] Institute of Computer Science, Freie Universität Berlin, Takustr. 9, D-14195 Berlin, Germany
{hritter,schiller}@inf.fu-berlin.de

SICS Technical Report T2003: 20
ISSN 1100-3154
ISRN:SICS-T--2003/20-SE

Abstract. Wireless sensor networks are based on the collaborative efforts of many small wireless sensor nodes, which collectively are able to form networks through which sensor information can be gathered. Such networks usually cannot operate in complete isolation, but must be connected to an external network to which monitoring and controlling entities are connected. As TCP/IP, the Internet protocol suite, has become the de-facto standard for large-scale networking, it is interesting to be able to connect sensornets to TCP/IP networks. In this paper, we discuss three different ways to connect sensor networks with TCP/IP networks: proxy architectures, DTN overlays, and TCP/IP for sensor networks. We conclude that the methods are in some senses orthogonal and that combinations are possible, but that TCP/IP for sensor networks currently has a number of issues that require further research before TCP/IP can be a viable protocol family for sensor networking.

1 Introduction

Wireless sensor networks is an information gathering paradigm based on the collective efforts of many small wireless sensor nodes. The sensor nodes, which are intended to be physically small and inexpensive, are equipped with one or more sensors, a short-range radio tranciever, a small micro-controller, and a power supply in the form of a battery.

Sensor network deployments are envisioned to be done in large scales, where each network consists of hundreds or even thousands of sensor nodes. In such a deployment, human configuration of each sensor node is usually not feasible and therefore self-configuration of the sensor nodes is important. Energy efficiency is also critical, especially in situations where it is not possible to replace sensor node batteries. Battery replacement maintenance is also important to minimize for deployments where battery replacement is possible.

Most sensor network applications aim at monitoring or detection of phenomena. Examples include office building environment control, wild-life habitat monitoring [17], and forest fire detection [24]. For such applications, the sensor networks cannot operate in complete isolation; there must be a way for a monitoring entity to gain access to the data produced by the sensor network. By connecting the sensor network to an existing network infrastructure such as the global Internet, a local-area network, or a private intranet, remote access to the sensor network can be achieved. Given that the TCP/IP protocol suite has become the de-facto networking standard, not only for the global Internet but also for local-area networks, it is of particular interest to look at methods for interconnecting sensor networks and TCP/IP networks. In this paper, we discuss a number of ways to connect sensor networks to TCP/IP networks.

Sensor networks often are intended to run specialized communication protocols, thereby making it impossible to directly connect the sensor network with a TCP/IP network. The most commonly suggested way to get the sensor network to communicate with a TCP/IP network is to deploy a proxy between the sensor network and the TCP/IP network. The proxy is able to communicate both with the sensors in the sensor network and hosts on the TCP/IP network, and is thereby able to either relay the information gathered by the sensors, or to act as a front-end for the sensor network.

Delay Tolerant Networking (DTN) [9] is a recently proposed communication paradigm for environments where the communication is characterized by long or unpredictable delays and potentially high bit-error rates. Examples include mobile networks for inaccessible environments, satellite communication, and certain forms of sensor networks. DTN creates an overlay network on top of the Internet and uses late address binding in order to achieve independence of the underlying bearer protocols and addressing schemes. TCP/IP and sensor network interconnection could be done by using a DTN overlay on top of the two networks.

Finally, by directly running the TCP/IP protocol suite in the sensor network, it would be possible to connect the sensor network and the TCP/IP network without requiring proxies or gateways. In a TCP/IP sensor network, sensor data could be sent using the best-effort transport protocol UDP, and the reliable byte-stream transport protocol TCP would be used for administrative tasks such as sensor configuration and binary code downloads.

Due to the power and memory restrictions of the small 8-bit micro-controllers in the sensor nodes, it is often assumed that TCP/IP is not possible to run in sensor networks. In previous work [8], we have shown that this is not true; even small micro-sensor nodes are able to run a full instance of the TCP/IP protocol stack. We have also successfully implemented our small uIP TCP/IP stack [7] on the small sensor nodes developed at FU Berlin [1]. There are, however, a number of problems that needs to be solved before TCP/IP can be a viable alternative for sensor network communication.

The rest of the paper is structured as follows. We discuss proxy architectures in Section 2, followed by a discussion of the DTN architecture in Section 3. TCP/IP for sensor networks is presented in Section 4, and a comparison of the three methods is given in Section 5. Finally, the paper is concluded in Section 6.

2 Proxy Architectures

Deploying a special proxy server between the sensor network and the TCP/IP network is a very simple and straightforward way to connect the two networks. In its simplest form, the proxy resides as a custom-made program running on a gateway computer which has access to both the sensor network and the TCP/IP network. Since all interaction between clients in the TCP/IP network and the sensor nodes is done through the proxy, the communication protocol used in the sensor network may be chosen freely.

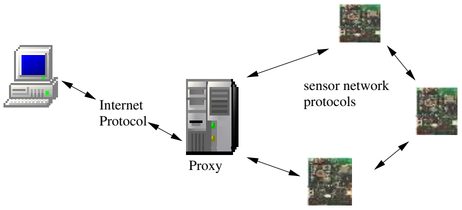

Fig. 1. Proxy architecture

The proxy can operate in either of two ways: as a relay, or as a front-end. In the first case, the proxy will simply relay data coming from the sensor network to clients on the TCP/IP network. The clients must register a particular *data interest* with the proxy, and the proxy will then relay data from the sensor network to the registered clients.

In the second case, where the proxy acts as a front-end for the sensor network, the proxy pro-actively collects data from the sensors and stores the information in a database. The clients can query the proxy for specific sensor data in a variety of ways, such as through SQL-queries or web-based interfaces.

One advantage of the proxy based approach to interconnect sensor and TCP/IP networks is that the proxy completely decouples the two networks. This naturally allows for specialized communication protocols to be implemented in the sensor network. A front-end proxy can also be used to implement security features such as user and data authentication.

Among the drawbacks of the proxy approach are that it creates a single point of failure. If the proxy fails, all communication to and from the sensor network is effectively made impossible. One possible solution would be to deploy redundancy in the form of a set of back-up proxies. Unfortunately, such a solution reduces the simplicity of the proxy approach. Other drawbacks are that a proxy implementation usually is specialized for a specific task or a particular set of protocols. Such a proxy implementation requires special proxies for each application. Also, no general mechanism for inter-routing between proxies exist.

Proxies have previously been used for connecting devices to TCP/IP networks in order to overcome limitations posed by the devices themselves, or limitations caused by the communication environment in which the devices are located. The Wireless Application Protocol (WAP) stack [15] is intended to be simpler than the TCP/IP protocol stack in order to run on smaller devices, and that is better suited to wireless environments. WAP proxies are used to connect WAP devices with the Internet. Similarly, the Remote Socket Architecture [23] exports the BSD socket interface to a proxy in order to outperform ordinary TCP/IP for wireless links.

3 Delay Tolerant Networks

The Delay Tolerant Network architecture [9] is intended for so-called *challenged environments*. Properties of such environments include long and variable delays, frequent network partitioning, potentially high bit-error rates and asymmetrical data rates. DTN is based on the observation that the TCP/IP protocol suite is built around a number of implicit assumptions that do not hold true in challenged communication environments. In particular, the underlying assumptions of TCP/IP are:

- An end-to-end path must exist between source and destination during the whole data exchange.
- The maximum round trip-time for packets must be relatively small and stable.
- The end-to-end packet loss is relatively small.

The DTN architectural design contains several principles to provide service in these environments:

- DTN uses an overlay architecture based on store-and-forward message switching. The messages, called *bundles*, that are transmitted contain both user data and relevant meta-data. A message-switched architecture provides the advantage of a priori knowledge of the size and performance requirements of the data transfer. The bundle layer works as an application layer on top the TCP/IP protocol stack.
- The base transfer between nodes relies on store-and-forward techniques, i.e., a packet is kept until it can be sent to the next hop. This requires that every node has storage available in the network. Furthermore, this allows to advance the point of retransmission towards the destination.

A DTN consists a set of *regions* which share a common layer called the *bundle layer* that resides above the transport layer. The bundle layer stores messages in persistent storage if there is no link available, fragments messages if necessary, and optionally implements end-to-end reliability. The layers below the bundle layer are not specified by the architecture, but are chosen dynamically based on the specific communication characteristics and the available protocols in each region. One or more DTN gateways exists in each DTN region. The DTN gateway forwards bundles between regions, and takes care of delivering messages from other regions to hosts within the local region.

The DTN architecture has been designed with the sensor network paradigm in mind. In sensor networks, the network may be partitioned frequently when nodes go into sleep mode or because of node failure. This will disrupt any end-to-end paths through the network. Also, packet loss rates in sensor networks can be very high [28] and routes may be asymmetric.

When connecting sensor networks to a TCP/IP network using the DTN architecture, we have at least two regions as depicted in Figure 2: one TCP/IP region where the TCP/IP protocol suite is used and one sensor network region where specialized sensor network protocols are implemented. A DTN gateway node is put in between the two networks, similar to where a proxy would have been placed.

The DTN gateway acts much as a relay proxy as discussed in the previous section, and the relay proxy approach can be viewed as a specific instance of the DTN architecture. The DTN architecture is much more general than a simple proxy based approach,

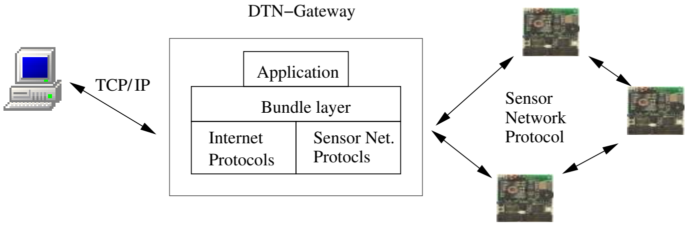

Fig. 2. Connecting using the DTN architecture

however, as the DTN architecture even allows mapping the sensor network into more than one DTN region, with DTN gateways located within the sensor network. For sensor networks where network partitioning is frequent, or where end-to-end communication is impossible, such a network design would be appropriate. A fully DTN enabled sensor network would easily be extended to a TCP/IP network, simply by connecting one or more of the DTN gateways to the TCP/IP network.

4 TCP/IP for Sensor Networks

Directly employing the TCP/IP protocol suite as the communication protocol in the sensor network would enable seamless integration of the sensor network and any TCP/IP network. No special intermediary nodes or gateways would be needed for connecting a sensor network with a TCP/IP network. Rather, the connection would simply be done by connecting one or more sensor nodes to the TCP/IP network. TCP/IP in the sensor network would also provide the possibility to route data to and from the sensor network over standard technologies such as General Packet Radio Service (GPRS) [4]. This leads to an architecture as shown in Figure 3.

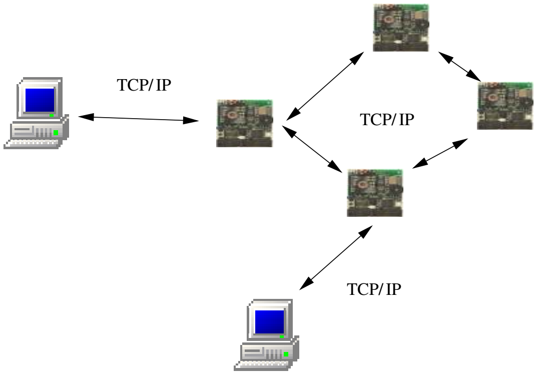

Fig. 3. Connecting using TCP/IP in the sensor network

Until recently, many believed that tiny sensor nodes would lack the necessary memory and computational resources to be able to run a full instance of the TCP/IP protocol stack. Therefore, the idea of using TCP/IP for sensor networks have not been given much research attention. We have showed that a full TCP/IP stack indeed can be run even on very limited devices [8], and have implemented our small uIP TCP/IP implementation [7] on the sensor nodes developed at FU Berlin [1]. These nodes are equipped with an 8-bit Texas Instruments MSP430 low-power micro-controller with a built-in memory of 2048 bytes. Our TCP/IP implementation requires only a few hundreds bytes of memory to operate, which leaves plenty of memory for the actual sensor node applications.

The fact that we are able to run the TCP/IP stack even on tiny sensor nodes suggest that TCP/IP for sensor networks may be within reach. Sensor networks running the TCP/IP protocol suite would be very easy to connect to existing TCP/IP networks, and would also able to benefit from the wealth of readily available applications such as file transfers using FTP or HTTP and possibly time synchronization with NTP. There are, however, a number of problems with using TCP/IP for wireless sensor networks that need to be addressed before TCP/IP is a viable alternative for sensor networks:

- The addressing and routing schemes of IP are host-centric.
- The header overhead in TCP/IP is very large for small packets.
- TCP does not perform well over links with high bit-error rates, such as wireless links.
- The end-to-end retransmissions used by TCP consumes energy at every hop of the retransmission path.

IP is designed so that every network interface connected to a network has its own IP address. The prefix of the address is the same for all network interfaces in the same physical network and routing is done based on the network prefixes. This does not fit well with the sensor network paradigm, where the main interest is the data generated by the sensors and not the individual sensors as such. Most of the proposed communication protocols for sensor networks use data centric routing and addressing [10, 12] and even though similar mechanisms have been developed as overlay networks on top of IP [21], these usually require too much state to be kept in the participating nodes to be feasible to run on limited sensor nodes.

The size of TCP/IP packet headers is between 28 and 40 bytes, and when sending a few bytes of sensor data in a datagram the headers constitute nearly 90% of each packet. Energy efficiency is of prime importance for sensor networks, and since radio transmission often is the most energy consuming activity in a sensor node [20], a header overhead of 90% is not acceptable, and most protocols developed for sensor networks strive to keep the header overhead as low as possible. For example, the TinyOS [11] message header overhead is only 5%. The header overhead in TCP/IP can be reduced using various forms of header compression [13, 6, 16, 5]. These mechanisms are commonly designed to work only over a single-hop link, but work is currently being done in trying to adopt these mechanisms to the multi-hop case [19].

Furthermore, since TCP was designed for wired networks where bit-errors are uncommon and where packet drops nearly always are due to congestion, TCP always interprets packet drops as a sign of congestion and reduces its sending rate in response to a dropped packet. This leads to bad performance over wireless links where packets frequently are dropped because of bit-errors. TCP misinterprets the packet loss as congestion and lowers the sending rate, even though the network is not congested.

Also, TCP uses end-to-end retransmissions, which in a multi-hop sensor network requires a transmission by every sensor node on the path from the sender to the receiver. Such a retransmission consumes more energy overall than a retransmission scheme where the point of retransmission if moved closer to the receiver would be able do. Protocols using other mechanisms to implement reliability, such as reliable protocols especially developed for sensor networks [22, 27, 26], are typically designed to be good at conserving energy.

Methods for improving TCP performance in wireless networks have been proposed [2, 3, 14], but these are often targeted towards the case where the wireless link is the last-hop, and not for wireless networks with multiple wireless hops. In addition, traditional methods assume that the routing nodes have significantly larger amounts of resources than what limited sensor nodes have.

5 Comparison of the Methods

The three methods for connecting sensor networks to TCP/IP networks presented here are in some respects orthogonal—it is possible to make combinations such as a partially TCP/IP-based sensor network with a DTN overlay connected to the global Internet using an front-end proxy. It is therefore not possible to make a direct comparison of the methods. Instead, we will state the merits and drawbacks of each of the methods and comment on situations in which each method is suited.

A pure proxy method works well when the sensor network is deployed relatively close to a place where a proxy server can be safely placed. Since the proxy server by design must have more processing power and more memory than the sensors, it is likely to require an electrical power supply rather than a battery. Also, the proxy may need to be equipped with a stable storage media such as a hard disk, which may make the proxy physically larger than the sensor nodes. One example of a situation where these criteria are met is an office building environment. Here, a proxy server can be placed close to the sensor network, perhaps even in the same room as the sensors, and have immediate access to electrical power. Another example would be a nautical sensor network where the proxy could be equipped with a large battery pack and placed in the water with a buoy such that the significance of the physical size of the proxy node would be reduced.

Front-end proxies can also be used for a number of other things, besides for achieving interconnectivity, such as sensor network status monitoring, and generation of sensor failure reports to human operators.

The DTN architecture can be viewed as a generalization of the proxy architecture and indeed a DTN gateway shares many requirements with a proxy server. A DTN gateway in the sensor network region will be placed at the same place as a proxy server would have been placed, and also requires more memory and stable storage media than the sensor nodes. There are, however, a number of things that are gained by using the DTN architecture rather than a simple proxy architecture. First, DTN inherently allows for multiple DTN gateways in a DTN region, which removes the single-point-of-failure problem of the simple proxy architecture. Second, while a proxy architecture usually is specialized for the particular sensor network application, DTN provides general mechanisms and an interface that can be used for a large number of occasions. Also, if the sensor network is deployed in a place with a problematic communication environment, the DTN architecture provides a set of features which can be used to overcome the communication problems. Examples of such situations would be deep-sea exploration or places where seismic activity can disrupt communication.

From an interconnectivity perspective, running native TCP/IP in the sensor networks is the most convenient way to connect the sensor network with a TCP/IP network. One or more sensor nodes would simply be attached to the TCP/IP network, and the two networks could exchange information through any of those nodes. The attachment can be done either using a direct physical link, such as an Ethernet cable, or over a wireless technology like GPRS.

While a TCP/IP enabled sensor network may provide the easiest way to interconnect the networks, it is usually not a complete solution, but must be integrated into a larger architecture. The proxy and DTN architectures discussed here are examples of such an architecture. We can e.g. imagine an office building TCP/IP sensor network that is connected to a front-end proxy located in the cellar of the building. The connection between the proxy and the sensor network would be made using the regular TCP/IP local-area network in the building. Another example would be a TCP/IP sensor network for monitoring the in-door environment in a train. A DTN gateway would be placed in the same train, and the sensor network and the gateway would communicate using TCP/IP over the train's local area network. The DTN gateway would be able to transmit the gathered information over the global Internet at places where the train has Internet access.

Finally, from a security perspective, the front-end proxy architecture provides a good place to implement user and data authentication, since all access to the sensor network goes through the proxy. The DTN architecture is inherently designed for security and uses asymmetric cryptography to authenticate both individual messages and routers. TCP/IP as such does not provide any security, so security must be implemented externally either by using a front-end proxy, DTN, or any of the existing security mechanisms for TCP/IP networks such as Kerberos. It should also be noted that security methods developed especially with wireless sensor networks in mind [18, 25] can be implemented as application layer security in TCP/IP sensor networks.

6 Conclusions

We have presented three methods for connecting wireless sensornets with TCP/IP networks: proxy architectures, Delay Tolerant Networking (DTN) overlays, and TCP/IP for sensor networks. The three methods are orthogonal in that it is possible to form combinations, such as a DTN overlay on top of a TCP/IP sensor network behind a front-end proxy.

The proxy architectures are simple and make it possible to use specialized communication protocols in the sensor network, but are application specific and creates a single point of failure. The DTN architecture also allows for specialized protocols, but provides a much more general communication architecture than that of the proxy architectures. DTN is also useful if the sensor network itself is deployed in a challenged communication environment.

Finally, by using the TCP/IP protocol suite for the sensor network, connecting the sensor network with another TCP/IP network is simply done by attaching one or more sensor nodes to both networks. However, attaching the sensor nodes to the TCP/IP network may not always be enough, but a combination of either a proxy architecture and TCP/IP, or DTN and TCP/IP, is usually required.

TCP/IP for sensor networks currently has a number of problems, and therefore further research in the area is needed before TCP/IP can be a viable alternative for sensor networking.

References

1. CST Group at FU Berlin. Scatterweb Embedded Sensor Board. Web page. 2003-10-21. **URL:** *http://www.scatterweb.com/*
2. H. Balakrishnan, S. Seshan, E. Amir, and R. Katz. Improving TCP/IP performance over wireless networks. In *Proceedings of the first ACM Conference on Mobile Communications and Networking*, Berkeley, California, November 1995.
3. K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. *ACM Computer Communications Review*, 27(5):19–43, October 1997.
4. J. Cai and D. Goodman. General packet radio service in GSM. *IEEE Communications Magazine*, 35:122–131, October 1997.
5. S. Casner and V. Jacobson. Compressing IP/UDP/RTP headers for low-speed serial links. RFC 2508, Internet Engineering Task Force, February 1999.
6. M. Degermark, M. Engan, B. Nordgren, and S. Pink. Low-loss TCP/IP header compression for wireless networks. *ACM/Baltzer Journal on Wireless Networks*, 3(5), 1997.
7. A. Dunkels. uIP - a TCP/IP stack for 8- and 16-bit microcontrollers. Web page. 2003-10-21. **URL:** *http://dunkels.com/adam/uip/*
8. A. Dunkels. Full TCP/IP for 8-bit architectures. In *Proceedings of The First International Conference on Mobile Systems, Applications, and Services (MOBISYS '03)*, May 2003.
9. K. Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of the SIGCOMM'2003 conference*, 2003.
10. J. S. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan. Building efficient wireless sensor networks with low-level naming. In *Symposium on Operating Systems Principles*, pages 146–159, 2001.
11. J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*, November 2000.
12. C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Mobile Computing and Networking*, pages 56–67, 2000.
13. V. Jacobson. Compressing TCP/IP headers for low-speed serial links. RFC 1144, Internet Engineering Task Force, February 1990.
14. J. Liu and S. Singh. ATCP: TCP for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 19(7):1300–1315, 2001.
15. Wireless Application Protocol Forum Ltd. *Official Wireless Application Protocol: The Complete Standard*. Wiley Computer Publishing, 2000.

16. S. Pink M. Degermark, B. Nordgren. IP header compression. RFC 2507, Internet Engineering Task Force, February 1999.
17. A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *First ACM Workshop on Wireless Sensor Networks and Applications (WSNA 20029*, Atlanta, GA, USA, September 2002.
18. A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar. SPINS: security protocols for sensor netowrks. In *Mobile Computing and Networking*, pages 189–199, 2001.
19. S. Mishra R. Sridharan, R. Sridhar. A robust header compression technique for wireless ad hoc networks. In *MobiHoc 2003*, Annapolis, MD, USA, June 2003.
20. V. Raghunathan, C. Schurgers, S. Park, and M. Srivastava. Energy aware wireless microsensor networks. *IEEE Signal Processing Magazine*, 19(2):40–50, March 2002.
21. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *Proceedings of ACM SIGCOMM 2001*, 2001.
22. Y. Sankarasubramaniam, O. Akan, and I. Akyildiz. ESRT : Event-to-Sink Reliable Transport in Wireless Sensor Networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing (MobiHOC 2003)*, 2003.
23. M. Schlger, B. Rathke, A. Wolisz, and S. Bodenstein. Advocating a remote socket architecture for internet access using wireless lans. *Mobile Networks and Applications*, 6(1):23–42, 2001. ISSN: 1383-469X
24. S. N. Simic and S. Sastry. Distributed environmental monitoring using random sensor networks. In *Proceedings of the 2nd International Workshop on Information Processing in Sensor Networks*, pages 582–592, Palo Alto, California, 2003.
25. F. Stajano. *Security for Ubiquitous Computing*. John Wiley and Sons, February 2002. ISBN: 0-470-84493-0
26. 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*, pages 102–112, Anchorage, Alaska, USA, April 2003. IEEE.
27. C.Y. Wan, A. T. Campbell, and L. Krishnamurthy. PSFQ: A Reliable Transport Protocol For Wireless Sensor Networks. In *First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, Atlanta, September 2002.
28. J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *The First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, Los Angeles, California, November 2003.