Public Review for
The Politecast Communication Primitive for Low-Power Wireless

Marcus Lundén and Adam Dunkels

Low power wireless networks differ greatly from their high performance siblings. Rather than optimize for throughput (data bits per second), low power wireless networks seek to minimize the energy to transmit a bit (joules per bit). This simple change turns many of our assumptions of how protocols behave on their head. Receiver-initiated MAC protocols begin look very attractive; application-level information can improve performance without significantly breaking layered abstractions; explicit bursts are preferable to steady, constant bitrate traffic. Furthermore, the vagaries and challenges of long-term wireless behavior exert a strong pressure towards very simple mechanisms which are easy to understand.

This paper presents Politecast, a deceptively simple API for low-power communication. Politecast skates the difficult tension between the energy cost of transmission and the energy cost of reception by leveraging application-level information. Politecast nodes periodically advertise information; a receiver wakes up to hear advertisements when application-level knowledge leads it to believe it needs them. This mechanism might seem so simple as to be obvious, but sometimes it is these subtle twists on what are believed to be completely understood areas that have the greatest insight and contribution. For the past decade, low-power research has increasingly focused on unicast. This paper takes a step back, "rediscovering" a primitive which we can now better reconsider with a much deeper understanding of the associated problems and challenges. Best of all, the paper does more than just present the mechanism and evaluate its implementation. The paper also presents details on how two applications use it, including an operational low-power wireless application for art gallery patrons.

Public review written by
Philip Levis
Stanford University, USA

acm ◆ **sigcomm**

The Politecast Communication Primitive for Low-Power Wireless

Marcus Lundén and Adam Dunkels
Swedish Institute of Computer Science
Kista, Sweden
{mlunden, adam}@sics.se

ABSTRACT

In low-power wireless networks, nodes need to duty cycle their radio transceivers to achieve a long system lifetime. Counter-intuitively, in such networks broadcast becomes expensive in terms of energy and bandwidth since all neighbors must be woken up to receive broadcast messages. We argue that there is a class of traffic for which broadcast is overkill: periodic redundant transmissions of semi-static information that is already known to all neighbors, such as neighbor and router advertisements. Our experiments show that such traffic can account for as much as 20% of the network power consumption. We argue that this calls for a new communication primitive and present politecast, a communication primitive that allows messages to be sent without explicitly waking neighbors up. We have built two systems based on politecast: a low-power wireless mobile toy and a full-scale low-power wireless network deployment in an art gallery and our experimental results show that politecast can provide up to a four-fold lifetime improvement over broadcast.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Low-power wireless, sensor networks, duty cycling

1. INTRODUCTION

Low-power wireless networking has long been investigated in the context of wireless sensor networks, but its application space has recently been significantly broadened with new applications in the smart grid, smart cities, building automation, industrial automation, home automation, and container tracking, to name a few [14]. In such networks, the power consumption of the network nodes must be low to achieve the necessary multi-year lifetimes.

In low-power wireless, power consumption and communication is intertwined. The radio transceiver typically is the most power consuming peripheral so to achieve a low power consumption, nodes must duty cycle their radio transceivers [1, 11, 13]. With recent duty cycling mechanisms, idle nodes have their radios switched off more than 99% of the time [3] while being ready to accept incoming traffic. To
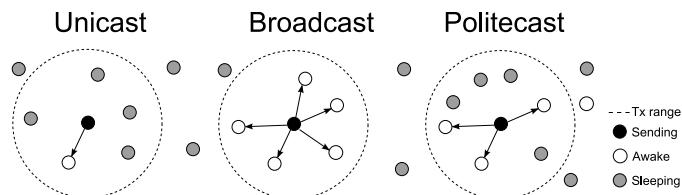

Figure 1: With unicast (left), only one receiver is awakened to receive the message; with broadcast (middle), all neighbors are awakened to receive the message; with politecast (right), only nodes that for themselves have decided to be aware receives the message.

send a message in a duty cycled low-power network, a sender must either wait until the receiver is known to be awake or explicitly wake up the receiver before sending the message. To send a unicast, only the receiver needs to be awakened. To send a broadcast, all nodes in the neighborhood must be awakened.

Many have found the traditional unicast and broadcast primitives to be too constraining for low-power wireless applications. For example, in achieving a low power consumption for neighbor discovery mechanisms, in the Disco and U-Connect neighbor discovery mechanisms [5, 8] nodes periodically transmit beacons that are sent to the link-layer broadcast address but that do not wake neighbors up. A similar communication pattern is found in receiver-initiated duty-cycling protocols such as RI-MAC [13], where nodes periodically transmit beacons to the link-layer broadcast address but without any explicit wake-up of neighbors.

We argue that there is an additional class of traffic for which the traditional unicast and broadcast primitives are too coarse-grained: periodic redundant transmissions. Periodic redundant transmissions are messages that is periodically sent and that contain information that is already known to neighbors. Examples are neighbor and router reachability advertisements or routing metric beacons. This information must be periodically transmitted since new or mobile nodes may join the network at any time. This type of traffic is traditionally sent using broadcast. But in a low-power wireless network, broadcasts are expensive in terms of power consumption and bandwidth since all neighbors must be explicitly awakened to receive the information. For this reason, protocols attempt to reduce the amount of periodic broadcasts they transmit [3, 7].

We propose that the communication patterns found in Disco, U-Connect, RI-MAC, and other low-power mechanisms should be generalized into a new communication primitive that can be efficiently used to reduce the power consumption for periodic redundant transmissions. We call this communication primitive *politecast.*

A politecast is a transmission that potentially reaches all neighbors, but only if the neighbors are actively listening for the transmission. The receivers independently decide if they are interested in hearing the transmission or not. Unlike unicast, a politecast is not directed at any particular receiver but potentially reaches multiple nodes. Unlike broadcast, a politecast is not intended to reach all nodes in the neighborhood. The relationship between unicast, broadcast, and politecast is shown in Figure 1.

With politecast, periodic redundant transmissions can be sent without disturbing those neighbors that already are aware of the information. New or mobile nodes, that do not have knowledge of the information, can independently listen for politecasts from neighbors in order to gather their information.

The contributions of this paper are twofold. First, we identify and generalize the politecast communication primitive based on communication patterns used in existing low-power wireless mechanisms. Second, we apply the politecast primitive to a class of traffic for which it has not previously been used and, with results from two politecast-based systems, show that politecast can provide significant energy savings.

The name of the politecast primitive comes from its politeness. A sender does not push its message onto its neighbors, but politely delivers it only to those that want to hear it.

2. POWER AND COMMUNICATION ARE INTERTWINED

In a low-power wireless network, nodes must duty cycle their radio transceivers to achieve a long system lifetime. Listening for transmissions is as expensive as sending data, so radio transceivers must be switched completely off to save power. To be able to communicate with a duty cycled radio, nodes must participate in a wake-up scheme that allow nodes to communicate while keeping the radio transceiver switched off as much as possible. Many such duty cycling schemes exist, but sender-initiated asynchronous duty cycling protocols are the most commonly used. Examples of sender-initiated asynchronous mechanisms are B-MAC [11], X-MAC [1], and ContikiMAC [3].

In sender-initiated asynchronous duty cycling protocols, the radio transceiver is periodically switched on to check for the presence of a wake-up signal. If a wake-up signal is heard, the radio is kept on for a while longer in anticipation of the data packet. Several methods to determine when to wake up exist. X-MAC uses a string of strobe packets to the intended receiver as the wake-up signal. When the sender receives an acknowledgment for the strobe, it sends the data packet. ContikiMAC repeatedly transmits the data packet itself instead of sending dedicated strobe packets. In receiver-initiated schemes such as RI-MAC [13], the roles are reversed: the sender periodically wakes up to transmit a probe packet, waits for a reply, then switches the radio off. A sender listens for a probe packet from the receiver and transmits its data packet in reply to the probe.
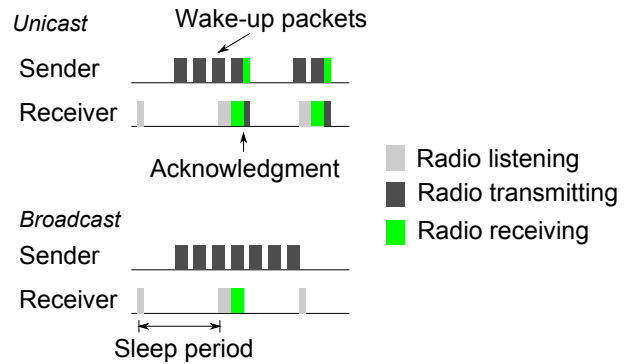
Figure 2: Unicast and broadcast transmissions with a sender-initiated asynchronous duty cycling mechanism. Unicast transmissions (top) awaken only the receiver. After learning the receiver phase, fewer wake-up packets need to be sent. Broadcast transmissions (bottom) always need to send a full sleep period of packets to reach all neighbors.

In both sender-initiated and receiver-initiated protocols, unicast transmissions can be made very efficient since senders can learn the wake-up phases of their neighbors [3]. The wake-up signal can then be sent when the neighbor wakes up to listen for it, without wasting transmissions when the receiver is sleeping. A sender learns a neighbors wake-up phase after a successful transmission.

Broadcast transmissions cannot be made as efficient as unicast transmissions, because a broadcast needs to reach all neighbors. Therefore a broadcast transmission must wake up every receiver in range. Thus a broadcast is inherently more expensive than a unicast. The difference is shown in Figure 2.

The behavior of a synchronous duty cycling mechanism is different from an asynchronous mechanism. In a synchronous mechanism, all nodes are time-synchronized and the global time is divided into time slots. Nodes are scheduled to listen for transmissions during specific time slots and a sender waits until its receiver has a listen slot before sending its message. Broadcast can be performed either by scheduling specific broadcast slots, during which all nodes listen for transmissions, or by having a sender transmit a packet for all neighbors' listen slots. In either case, broadcast is inherently more costly than unicast.

3. BROADCAST IS OVERKILL

We argue that there is a class of transmissions for which broadcast is overkill: periodic redundant transmissions of semi-static information that is already known to all or most neighbors. Examples of such periodic redundant transmissions are neighbor or router reachability messages and periodic advertisements of local non-changing routing metrics. Today, such information is sent using broadcast. Since the information already is known to the receivers, the use of broadcast for this traffic type results in at least three problems:

Redundant receptions cost energy. We define a redundant reception as a transmission that reaches nodes that are not interested in the transmission, i.e., when a transmis-

sion contains information that the receiver already knows. To receive a broadcast transmission, all receiving nodes must be awakened, which costs energy for all receivers, regardless if the transmission was necessary or not.

Increased congestion reduces application performance. Congestion occurs when nodes transmit more information than can be accommodated by the communication bandwidth. For broadcasts, the wake-up signaling consumes bandwidth and contributes to congestion. When congestion occurs, application performance suffers. To reduce the risk of congestion, application and protocol designers need to reduce their broadcast rate, potentially reducing application performance.

Unbalanced effort causes infrastructure strain. The cost of a broadcast transmission is typically higher for the sender than for the receivers. In applications where one class of nodes, such as an infrastructure, uses broadcast transmissions to distribute state to its network, these nodes will spend more energy than the others.

To quantify the effect of periodic redundant transmissions in a real-world setting, we set up a small testbed experiment: a data collection network with 23 Tmote Sky motes running the Contiki operating system version 2.5, the ContikiMAC duty cycling protocol with the default channel check rate of 8 Hz, and the Contiki collect data collection protocol. The Contiki collect protocol is similar to the TinyOS Collection Tree Protocol [7] and uses periodic broadcasts to disseminate each node's depth in the routing tree. The routing metric reflects the expected number of transmissions to reach the root node and potentially changes every time a message must be retransmitted due to packet losses. The protocol has been optimized to reduce the amount of control traffic through the use of adaptive beaconing [7]. We used the Contiki power profiling mechanism to measure the nodes' power consumption [4]. Each node sent one message per two minutes, common traffic intensity [7], towards the root of the collection tree. Routing beacon transmissions that contained the exact same routing metric as the previous beacon were defined to be a redundant transmission. We let the network stabilize, then run for one and a half hour during which we collected between 61 and 63 messages from each node. Our measurements showed that beacon transmissions and receptions made up 40.6% of the total network power consumption and that 49.7% of all periodic beacon transmissions were considered redundant. Thus roughly one fifth of the energy spent by the nodes in the network was spent on periodic redundant transmissions and their corresponding receptions.

We argue that these problems call for a new communication primitive that allows information to be transmitted to neighbors, but without waking them up. Instead, the neighbors themselves decide if they are interested in the information or not. If so, they are awake and can receive the politecast. Although being awake requires nodes to spend energy, they need to do this infrequently.

4. POLITECAST: REACH ONLY THOSE WHO LISTEN

Politecast is a communication primitive in which a transmission only reaches those neighbors that explicitly listen for a politecast transmission. Nodes that are not interested in hearing politecasts do not receive politecasts transmission.

Receivers independently decide whether or not to listen for politecast transmissions.

Like broadcast, politecast messages are not directed to any specific node but may be received by any node in the range of the transmitter. But unlike broadcast, politecast messages does not necessarily reach any of the nodes in range of the transmitter, if no nodes are listening for politecasts.

Politecast is intended for transmissions with information that most neighbors are likely to already being aware of. Nodes that are not already aware of the information, such as mobile nodes that have reached a new location and nodes that have recently been switched on or rebooted, can explicitly listen for the politecast messages from neighbors.

A politecast transmission is realized in different ways depending on the duty cycling mechanism used. In many cases, such as for asynchronous protocols, a politecast transmission is simply a link-layer transmission that is addressed to the link-layer broadcast address. To listen for politecasts, nodes simply keep their radios switched on. For synchronous protocols, more elaborate schemes are needed.

Nodes must have a listen policy for deciding when to stay awake to listen for politecasts. This policy is not dictated by the politecast primitive, but is application specific. In an application with mobile nodes, nodes will typically listen for politecasts when they have detected physical movement, as suggested by previous work [6, 10]. At their new location, the nodes can learn about their new network topology by listening for politecasts from neighbors.

4.1 Deciding when to Listen

To receive politecasts, receivers must explicitly listen for them. When a node will listen for transmissions is not dictated by the politecast primitive, but controlled by the application. Depending on the duty cycling mechanism used, listening may be expensive in terms of power, so the application must choose a power-efficient strategy for deciding when to listen for politecasts.

Different applications may choose different listen strategies. In an application with mobile nodes and a set of fixed infrastructure nodes, as in our art gallery deployment, the mobile nodes will listen when they have moved to a new location [6, 10]. The infrastructure nodes periodically politecasts information about the network topology. When a mobile node has received a politecast message from the infrastructure, it stops listening for politecast messages. With such a strategy, only the mobile nodes take a performance penalty for their mobility.

4.2 Politecast Under Different Duty Cycling Schemes

Although we thus far have discussed politecast under the assumption of a sender-initiated asynchronous duty cycling mechanism, the politecast primitive can be implemented in any duty cycling mechanism. With a receiver-initiated protocol such as RI-MAC, the definition of a politecast transmission is the same as for sender-initiated protocols: politecasts are transmitted as link-layer broadcasts and potential receivers decide to listen for them simply by keeping their radios on. However, politecasts can also be piggybacked on the periodic beacons that are transmitted as part of the normal operation of the receiver-initiated duty cycling mechanism.

Under synchronous duty cycling mechanisms, the implementation of politecast differs from that of asynchronous
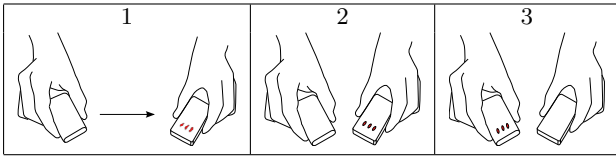
Figure 3: Our Steal the Light toy: when two devices are moved within the vicinity of each other, the LED signature is stolen.

Figure 4: For a low beacon rate, politecast has a slightly higher duty cycle than broadcast, but for higher beacon rates politecast has a much lower duty cycle.

Figure 5: With politecast, the duty cycle decreases with less frequent movement. With broadcast, the duty cycle is nearly constant.

mechanisms because in a synchronous mechanism, all transmissions and receptions are explicitly scheduled. Politecasts in a synchronous protocol are transmitted only during one or more dedicated time slots. Receivers that wish to listen for politecasts schedule a listen slot during these time slots, while the others keep their radios off. Thus those nodes that listen during the politecast slots will receive the politecast transmissions, but the others will not.

5. EVALUATION

We evaluate politecast with two applications that differ both in behavior and in scale: one mobile toy and one full-scale low-power network deployment. For both applications, we have implemented two versions of the software on the devices: one that is built on top of politecast and one that is built on top of broadcast. Our aim is to provide initial insight into the trade-offs in application performance and power consumption, to guide further research into the behavior and run-time properties of the politecast primitive.

We have implemented politecast in the Contiki operating system. As our hardware platform we use a set of Tmote Sky motes, a widely used sensor network platform, and a later version of the same mote called the Sentilla JCreate. Both are equipped with a CC2420 IEEE 802.15.4 low-power radio transceiver, an MSP430 microcontroller, a set of sensors, and LEDs. The Sentilla JCreate is also equipped with an accelerometer, which we use to measure physical mobility of the device. The power consumption of the device is approximately 60 mW when the radio transceiver is on, regardless if it is transmitting, receiving, or listening. When the transceiver is off, the power consumption is less than 0.1 mW. We use the transceiver on-time as a proxy for the total power consumption and the built-in power profiling module in Contiki [4] to measure the transceiver on-time.

5.1 A Toy Example: Steal the Light

Steal the Light is a small toy system where hand-held devices can "steal" the LEDs from other devices when they are in the immediate vicinity of each other. Steal the Light uses a set of hand-held Sentilla JCreates. One device has their LEDs switched on in a specific signature. When a device is moved to the vicinity of another device, the device "steals" the LED signature from the other device and displays it with its own LEDs, as shown in Figure 3.

The devices use periodic beacons, sent as broadcast or politecast, to announce themselves to nearby nodes. When a nearby node has been detected, the nodes exchange the LED signature using unicast. It does not matter which node detects the other.

Politecasts are less expensive to transmit than broadcasts, but politecasts require the receiver's radio to be explicitly
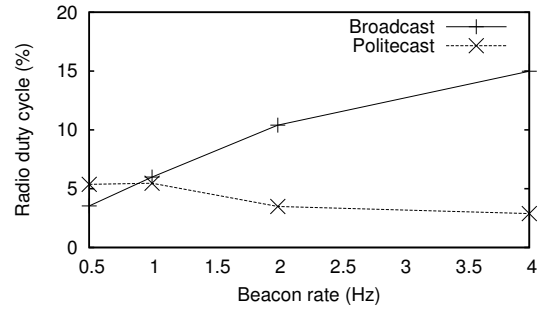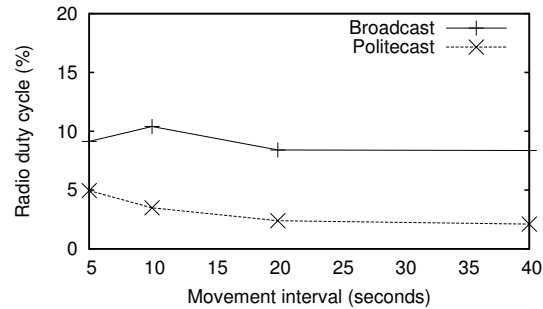
switched on, which also costs energy. The politecast-based version of Steal the Light starts to listen for policast transmissions when physical movement has been detected and keeps the radio on for a beacon interval.

The beacon rate affects both application performance and system lifetime. With a high rate, the devices detect each others' presence quicker than with a low rate, but also requires them to spend more energy on transmissions.

To study the politecast power consumption trade-off, we ran a set of experiments where we measured the power consumption of the devices while we moved the devices with a fixed interval, once every ten seconds. The system used ContikiMAC as the duty cycling protocol with a channel check rate of 16 Hz. We varied the beacon rate between experiments. Figure 4 shows the result: at a low beacon rate, politecast results in a shorter lifetime because of longer listening times, but at a higher rate, the broadcast transmission costs begin to dominate and politecast yields a significantly longer lifetime.

To study the effect of the physical movement, we varied the interval between physical movements with a fixed beacon rate (2 Hz). Figure 5 shows the result. The broadcast-based version is not affected by the activity, but the politecast-based version has an increased lifetime with less movement.

Our results show that politecast can give significant savings in power consumption over broadcast.
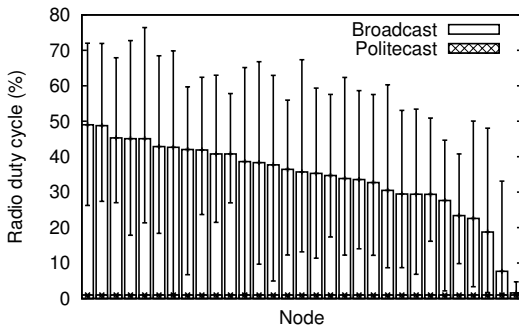
Figure 7: The radio on-time is significantly higher and has a higher variance for the broadcast-based version than for the politecast-based version, which is so low that it is hardly visible.
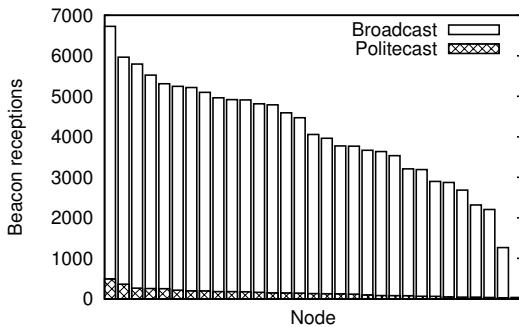
Figure 8: Redundant receptions are much more prevalent in the broadcast-based version than in the politecast-based version.

5.2 A Full-scale Deployment: The Art Gallery Network

We deployed a low-power wireless network in an art gallery as part of a research project where we studied visitors and their interactions with a digital environment in the gallery. Visitors were given a hand-held device with which they could leave traces of their visit to other visitors by using buttons on the device. Other visitors get physical feedback from their mobile device, with on-device LEDs and vibration motors, of the traces left by other visitors. The deployment consisted of 31 Tmote Sky infrastructure nodes, located as shown in Figure 6, five custom-built handheld nodes, and eight additional mobile Sentilla JCreates used for utility functions.

The deployment was temporary, by design, but each deployment could be deployed for months. The infrastructure therefore needed to consist of wireless low-power nodes. Since the nodes were concealed and mounted between 3 and 4 meters up on the walls, batteries could not be easily replaced. The infrastructure nodes had a dual purpose: both to act as a traffic backbone for data generated by the mobile nodes and to act as a reference point for approximate localization of the mobile nodes.

To determine the effect of politecast, we ran the broadcast-based version and the politecast-based version of the system for 60 minutes each, with no mobile nodes, and collected power consumption measurements and congestion estimation data from the infrastructure. Both networks used X-MAC with a channel check rate of 2 Hz. The resulting radio duty cycle is shown in Figure 7. We see that the duty cycle is significantly higher for the broadcast-based version, and also more varying, than the politecast-based version. This is due in part to the higher transmission cost for broadcasts, but also to that the nodes overhear transmissions from other nodes. The map of the deployment, as shown in Figure 6, shows that many nodes are in transmission range of each other. Figure 8 shows the amount of beacon receptions from other nodes, for each of the infrastructure nodes. Since the infrastructure nodes are not interested in beacons from other infrastructure nodes, these receptions are redundant and therefore constitute needless energy expenditure. Not surprisingly, the broadcast-based version has a significantly higher redundant reception rate than the politecast-based version.

Seeing that politecast made a significant difference in power consumption, we used the politecast version for the two months long network deployment.

6. RELATED WORK

Broadcasts have a history of problems in wireless multi-hop networks but these problems have been for multi-hop broadcast forwarding (flooding). If nodes blindly re-broadcast received broadcast packets, the network can enter a broadcast storm. Several mechanisms have been devised to tackle the broadcast storm problem in wireless multi-hop networks, such as Trickle [9] and RPB [12]. These protocols solve a different problem than politecast: politecast addresses power consumption and congestion in single-hop broadcast transmission and not the problems in multi-hop network flooding.

Many mechanisms try to reduce the number of broadcast transmission. Examples include adaptive beaconing [7] and beacon coordination [3]. Unlike those mechanisms, politecast does not try to reduce the number of transmissions, but instead make each transmission cheaper.

Vertically integrated systems such as Dozer [2] achieve a low power consumption, but does so at the price of modularity and layering. Dozer consists of a routing protocol that is conflated with a synchronous radio duty cycling mechanism. In such vertically integrated systems, there is little need to define layers or abstractions between layers inside the system. In contrast, the purpose of politecast is to find a modular element that can be used to build systems that have similarly low power consumption as a vertically integrated system, but with a clean separation of concerns in terms of layering. Politcast can be used below any network layer or protocol and on top of any underlying duty cycling mechanism.

7. CONCLUSIONS

In low-power wireless networks, radio transceivers are duty cycled to save power and nodes need to be woken up to receive transmissions. With duty cycling, the traditional communication primitives unicast and broadcast do not always provide a favorable power trade-off and many low-power protocols have sidestepped these primitives by allowing messages to be sent without waking nodes up. We generalize this communication pattern into a new communication primitive that we call politecast. Unlike unicast and broadcast trans-
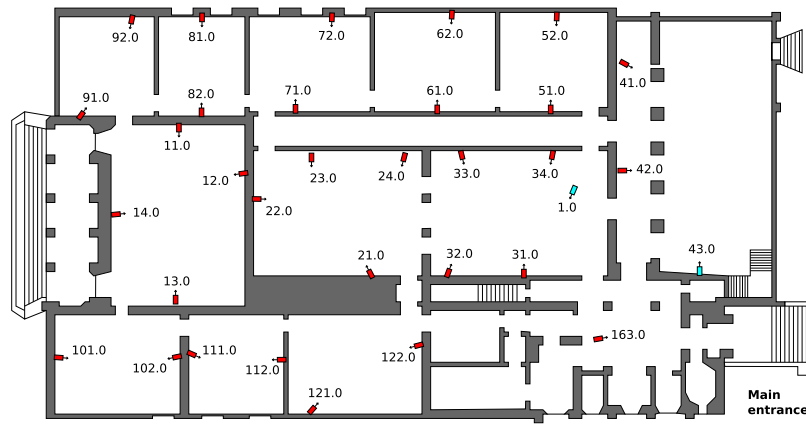
Figure 6: The art gallery infrastructure nodes were used both as a traffic backbone and for approximate localization.

missions, a politecast transmission does not wake neighbors up, but reaches only those that explicitly are awake. We argue that politecast can be efficiently used for periodic redundant transmissions; transmissions that contain information that the neighbors already know. New or mobile nodes, who need to quickly gather information about the network, will explicitly listen for the periodic politecast transmissions from its neighbors. We experimentally evaluate how well politecast can save reduce power consumption for periodic redundant transmissions in a toy example and a full-scale network deployment. Our results show that politecast can save significant amounts of both power and bandwidth, thereby allowing increased network lifetime and application performance.

Acknowledgements

8. REFERENCES

[1] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, pages 307–320, Boulder, Colorado, USA, 2006.

[2] N. Burri, P. von Rickenbach, and R. Wattenhofer. Dozer: ultra-low power data gathering in sensor networks. In *Proceedings of the International Conference on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, Cambridge, Massachusetts, USA, 2007.

[3] A. Dunkels, L. Mottola, N. Tsiftes, F. Österlind, J. Eriksson, and N. Finne. The announcement layer: Beacon coordination for the sensornet stack. In *Proceedings of the European Conference on Wireless Sensor Networks (EWSN)*, 2011.

[4] A. Dunkels, F. Österlind, N. Tsiftes, and Z. He. Software-based on-line energy estimation for sensor nodes. In *Proceedings of the IEEE Workshop on Embedded Networked Sensor Systems (IEEE Emnets)*, Cork, Ireland, June 2007.

[5] P. Dutta and D. Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Raleigh, North Carolina, USA, 2008.

[6] P. Dutta and D. Culler. Mobility changes everything in low-power wireless sensornets. In *Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS)*, Monte Verita, Switzerland, May 2009.

[7] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Berkeley, CA, USA, 2009.

[8] A. Kandhalu, K. Lakshmanan, and R. Rajkumar. U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol. In *Proceedings of the International Conference on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, pages 350–361, Stockholm, Sweden, 2010.

[9] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the USENIX Symposium on Networked Systems Design & Implementation (NSDI)*, March 2004.

[10] M. Malinowski, M. Moskwa, M. Feldmeier, M. Laibowitz, and J. Paradiso. Cargonet: a low-cost micropower sensor node exploiting quasi-passive wakeup for adaptive asychronous monitoring of exceptional events. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, pages 145–159, Sydney, Australia, 2007.

[11] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, pages 95–107, Baltimore, MD, USA, 2004. ACM Press.

[12] F. Stann, J. Heidemann, R. Shroff, and M. Z. Murtaza. RBP: robust broadcast propagation in wireless networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, pages 85–98, Boulder, Colorado, USA, 2006. ACM.

[13] Y. Sun, O. Gurewitz, and D. Johnson. RI-MAC: A Receiver-Initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Raleigh, NC, USA, 2008.

[14] J.P. Vasseur and A. Dunkels. *Interconnecting Smart Objects with IP: The Next Internet*. Morgan Kaufmann, 2010.