Probabilistic Information Dissemination for MANETs: the IPAC Approach

Odysseas Sekkas[1], Damien Piguet[2], Christos Anagnostopoulos[1], Dimitrios Kotsakos[1], George Alyfantis[1], Corinne Kassapoglou-Faist[2] and Stathes Hadjiethymiades[1]

[1]Pervasive Computing Research Group Department of Informatics and Telecommunications, University of Athens Panepistimioupolis, Illissia, 15784, Athens, Greece

[2]CSEM, Centre Suisse d'Electronique et de Microtechnique S.A. Jaquet-Droz 1, CH-2002, Neuchâtel, Switzerland

[1]{sekkas, bleu, d.kotsakos, alyf, shadj}@di.uoa.gr

[2]{damien.piguet, corinne.kassapoglou-faist}@csem.ch

Abstract Efficient data dissemination in MANETs presents a significant challenge. Epidemic dissemination is introduced as a method to reliably spread information (context) across a network in which no direct path from source to destination can be secured. We propose a probabilistic broadcast scheme instead of the flooding technique, thus, reducing significantly the volume of message transmissions seen throughout the network. Simulation results prove the efficiency of such scheme which achieves full coverage of the network with disseminated context.

1 Introduction

One of the main problems in mobile ad-hoc networks (MANETs) is the efficient dissemination of data, typically from the different sources to destination nodes. Solutions that work efficient for a specific setup do not perform well on slightly different applications. Hence, careful examination and selection of dissemination algorithms is needed. Epidemic dissemination is introduced as a method to reliably spread information across a network in which no direct path from source to destination is guaranteed.

The Integrated Platform for Autonomic Computing (IPAC) project aims at delivering a middleware and service creation environment for developing embedded, intelligent, collaborative, context-aware services in mobile nodes. IPAC relies on short-range communications for the ad hoc realization of dialogs among collaborating nodes. Advanced sensing components leverage the context-awareness attributes of IPAC[1], thus rendering it capable of delivering highly innovative applications for

[1] IPAC - Integrated Platform for Autonomic Computing (ICT-224395), is funded by the European Community through FP7 ICT Program. Web site: http://ipac.di.uoa.gr

pervasive computing. IPAC networking capabilities are based on epidemic / rumour spreading techniques, a stateless and resilient approach, and information dissemination among embedded nodes.

Spreading of information is subject to certain rules (e.g., space, time). IPAC nodes may receive, store, assess and possibly relay the incoming content to other nodes. The first requirement for the IPAC dissemination scheme is to support multiple sinks for the same piece of information. Most of the constraints (energy, limited processing capability) encountered in Wireless Sensor Networks (WSNs) can be found on IPAC nodes as well. On top of that, the mobility of IPAC nodes adds more constraints and problems that must be dealt with when disseminating information. In this sense, the IPAC framework has the operating parameters of a Mobile Sensor Network (MSN), which is a WSN with moving sensors.

The rest of the paper is organized as follows: previous and related work is presented in Section 2. Section 3 describes the problem of information dissemination in the context of IPAC and presents a set of preliminary simulation results in an effort to shed some light on how information dissemination is affected by various model parameters. Based on these results, adaptive probabilistic schemes are proposed and assessed. Section 4 concludes the paper and presents directions for future work.

2 Related work

The simplest technique to spread information is flooding. Another one is that the node or process that owns a piece of information broadcasts it regularly to random subsets of its neighbors with certain probability [1], [2]. The basic parameters of such epidemic dissemination models were defined in [3] and are the number of times a message is forwarded, the buffer capacity for each node or process, the total number of nodes or processes (system size), the number of known nodes or processes (partial view size) and the size of target group of nodes. The main issues associated with information spreading are scalability and reliability of the dissemination.

Such simple models suffer from certain drawbacks: do not scale nicely; impose considerable overhead (traffic) due to deliveries to uninterested nodes, and do not take into account the limited amount of storage available on the nodes [3]. Additionally, they do not take into account the membership issue (i.e., which node knows which other). Work performed so far addresses some of the aforementioned issues. In all cases, the goals of every proposed algorithm or technique are threefold: maximize message delivery rate, minimize message latency and minimize the total resources consumed in message delivery [4].

The SPIN protocols, [5], [6] were introduced to improve the situation. They are based on meta-data (i.e., semantic awareness). They can be bound to the needs of each application, and they impose selective retransmissions of information, thus minimising retransmission overhead. Nodes transmit after ensuring that the information to be transmitted is useful and after probing their own resource manager. In this manner, the problems of redundant information transmission and resource-blind transmission are addressed. Publish/subscribe models have also been proposed [7] as

directed diffusion-based schemes. Probabilistic broadcast and multicast schemes [8], [9] have been proposed in order to address the reliability issue raised as one moves away from the flooding concept. Gossip-based broadcast algorithms trade reliability guarantees against "scalability" properties, which is a known pattern in this field. The concept of awareness can apply to network-related properties too, such as the received signal strength indication or RSSI [10].

More limited awareness may be used in random phone call-type algorithms to push rumors to one random partner each time. This does not use awareness beyond immediate neighbor awareness, but can impose considerable overhead traffic in order to reach good reliability [11]. Dissemination schemes that avoid flooding the network without the need to maintain subscription information have also been proposed [12]. A multi-epidemic approach that relies on semantic dependencies, which are modeled through a hierarchical representation scheme, is investigated in [13].

3 IPAC information dissemination

In IPAC, information dissemination actually boils down to the forwarding of messages by each node, according to specific rules. Messages are broadcast by each node to its neighborhood, i.e., are not explicitly destined to specific neighbors. Nodes receive every message they listen to and process it according to some rules/policies. It is also possible that nodes subscribe for information that they are interested in. In case that relevant information is received through an IPAC message, it is delivered to the appropriate application, and, then, such message is considered again for forwarding. In this paper we only focus on the dissemination of messages throughout the network, leaving outside details regarding the application layer.

At each time instant, nodes might be in vicinity and connected to each other thus forming a graph. Note that, due to mobility, such graph is not static and might change quite rapidly. A node, at any time, might generate a message which will have several attributes, typically set by the application that generated it (e.g. time validity – TTL or criticality). A node is a reconfigurable entity that has several parameters tunable to adapt to a specific situation. The concept of IPAC is to let the node itself observe its situation and decide how to optimally tune its operation. A situation can be decomposed into a set of primitive elements. The proposed scheme represents a typical cognitive networking model where each node is capable of sensing/observing its situation/context, and using this information to intervene with diverse, cross-layer parameters within the node, thus, optimizing its operation.

3.1 Non-adaptive probabilistic broadcast algorithm – L0

In this section, we present the simulation results of a non-adaptive broadcast algorithm. Such simulations assist in the design of an adaptive algorithm. In this setup, the number of broadcast attempts (f_m) and the probability of broadcast by a certain

node (β) are the same for every node and do not vary during the simulation. All simulations have been performed using the Omnet++ tool [14].

The basic setup involves $N = 25$ nodes which are placed on a 5×5 squared grid Nodes are fixed (zero mobility). While the number and the relative positions of the nodes remain static, the side size w varies in order to change the network density. For the lower layers of the network we employed IEEE 802.15.4 non-beacon enabled and CSMA MAC layer. Table 1 summarizes the simulation parameters along with the corresponding domains adopted for the simulations.

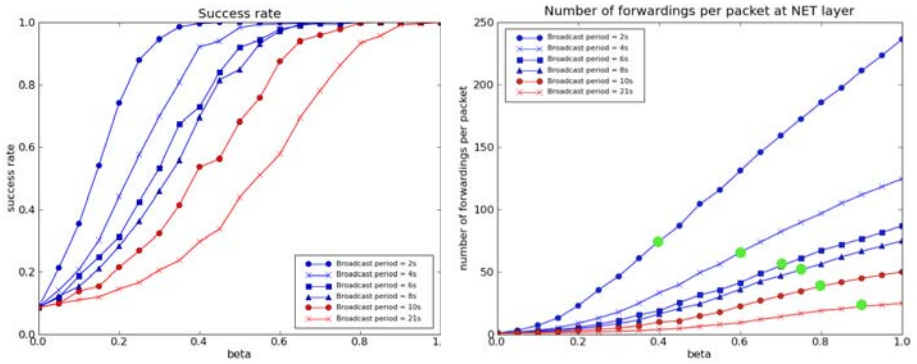
Table 1. Simulation parameters.

N: number of nodes	25
P0: Number of packets sent by node 0 (for other nodes: none)	50
Traffic type	Exponential(10s)
TTL [s]	20
Backoff [s]	Uniform in [0, 1]
Number of Runs per β	5
w: square size [m]	50, 100, 200, 300, 400
β: probability of broadcast	2, 4, 6, 8, 10, 21
T_m: broadcast period [s]	from 0 to 1 with steps of 0.05

In this simulation setup only node with identifier 0 sends packets. The back off parameter is the probabilistic delay upon which every node waits before sending the first copy of a message to mitigate the risk of systematic collisions. From T_m, max of back off and *TTL*, it is effortless to calculate how many times a message will be sent at most. In our case: f_m = 10, 5, 4, 3, 2 and 1 times. The node, which creates a message, sends it definitely (β=1.0) at the first attempt, independent of the current value of β. Each message *m,* which arrives at the network layer, is forwarded only once to the application. Other copies of that message are discarded. The metrics that were monitored through the simulations are:

- *Success rate (good-put):* (total number of received packets) / (total number of expected packets). Total refers to all the nodes. The total number of expected packets is defined as $(N\text{-}1)\cdot P_0$.
- *Number of forwardings:* average number of packet transmissions within the network per message.

In Fig. 1a we provide plots of the success rates (good-put) for w = 300m. We observe percolation phenomena such that full network coverage is obtained after a certain value of β. Once this threshold is reached, it is meaningless to further increase β. This percolation value decreases with network density and increases with the broadcast period (i.e., decreases with the number of transmissions), as expected. Once the percolation conditions are known, we can consider the cost factor. In our case we estimate the cost from the number of transmissions needed within the whole network in order to obtain a given coverage.



(a) (b)

Fig. 1. Side size w= 300m. (a) Success rates (good-put). (b) Number of packet transmissions per message in the network.

In Fig. 1b we have added enlarged dots to indicate percolation probabilities for w=300 m. We observe that to reach full coverage, it is cheaper, in terms of number of transmissions, to send a message once with high probability than to forward it several times with lower probability. From our observations, we derive hints for the design of the adaptive algorithm:

Hint 1. The broadcasting probability must decrease with the network density.

Hint 2. To reach a desired coverage at the lowest energy cost, one should adopt higher probabilities along with lower number of broadcasts (or longer broadcast periods).

In another scenario we simulate conditions of network congestion. All nodes send 50 packets and we varied the exponential traffic parameter from a mean of 10 seconds to 1 second. We also introduce an additional metric which is the number of packets dropped by the MAC layer due to congestion. The derived hint from this simulation experiment is:

Hint 3. The dissemination protocol must observe or be notified of MAC status. When packets are dropped at MAC layer, probability of broadcast must be decreased until the number of dropped packets becomes negligible.

Scenarios involving random mobility have also been simulated. Good-put results showed that sometimes mobility helps achieving full dissemination with lower probabilities while, under certain conditions, mobility undermines good-put. Hence we could not find a clear rule about how the mobility should influence the probability.

3.2 Adaptive Probabilistic Broadcast algorithm – L1

In this section, we propose an adaptive probabilistic broadcast algorithm. The difference between the adaptive and the non-adaptive probabilistic broadcast is that the probabilistic parameter β is dynamically adjusted by each node according to Hint 1, section 3.1. Specifically, the network density is inferred from the estimation of the

number of neighbors. Each time a packet is received, the address of the sender is registered in a "neighbors" table. At that time, a timer related to the new entry is set to T_L. When the timer expires, the entry is removed from the table. When the node receives a packet from a neighbor that is already registered, it restarts the corresponding timer to T_L. Each T_c seconds, the node counts how many nodes are in its table. After that it adjusts the value of β according to the number of neighbors estimation with the values presented in Table 2, determined experimentally:

Table 2. Adaptive probabilistic broadcast: values of beta (β) for different neighbourhood sizes

Number of neighbours	0, 1, 2, 3	4, 5	6, 7	8, 9	10, 11	12, 13	14, 15	16, 17	18, 19	≥20
B	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1

We adopt the same simulation setup regarding the grid layout, the number of nodes and other settings (TTL, etc.) in section 3.1. The timer parameters of the adaptive scheme are T_L = 10s and T_c = 5s. The broadcast period is set to TTL + 1 to ensure only one broadcast. Therefore, there is only one value per grid size. We examine the behavior of β derived from the network nodes. As we can conclude from Table 3, the adaptive algorithm allows full or, almost full, coverage even when the number of broadcasts is only one.

Table 3**.** Coverage and number of forwardings for adaptive probabilistic broadcast. Grid layout, 25 nodes. Only node 0 (top, left) sends 50 packets. Mean inter-packet interval: 10s. TTL = 20s. Broadcast period = 21. Initial value of β when the node is turned-on: 0.9.

Grid width [m]	50	100	200	300	400
Good-put	1.000	0.999	0.996	0.980	0.055
Number of forwardings[2]	14.9	15.7	20.7	22.0	2.2

Once the network density is high, we notice that the average number of transmissions per message is higher than necessary. If traffic is too low, the values of T_L and T_c may be too low to allow nodes to find out all their neighbors. To verify this assumption, the simulation is re-executed with all 25 nodes set to send 50 packets. The results are provided in Table 4 and plotted in Fig. 2.

Table 4: Good-put and average number of transmissions per message when all nodes send 50 packets.

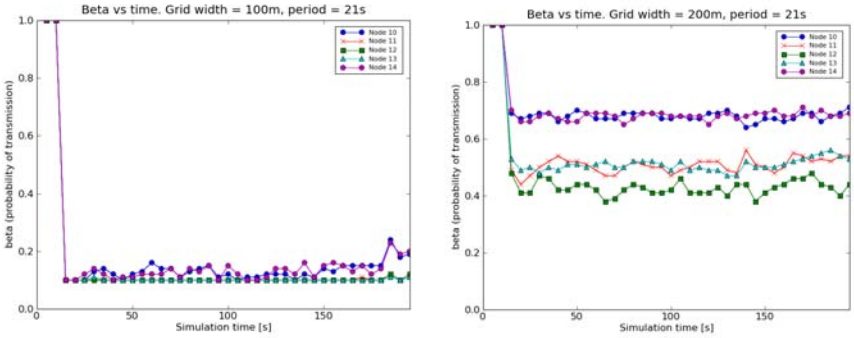
Grid width [m]	50	100	200	300	400
Good-put	0.996	0.962	0.952	0.964	0.111
Number of forwardings	3.9	5.0	15.9	20.0	3.6

As expected, the algorithm is more efficient in presence of enough traffic. We observed also a border effect: nodes which are near the border of the playground (nodes 10 and 14) detect fewer neighbors and infer a lower density.

We have also run a mobility scenario with the adaptive algorithm. Evidently, we obtain that increasing the number of broadcast attempts (that is decreasing the

[2] Average number of times a particular message is forwarded in the network, including the first emission when it is created. If β = 1.0 and broadcast period > TTL (one broadcast at each node), average number of forwarding per message = number of nodes.

broadcast period) definitely achieves better network coverage. This further supports our strategy of tuning the probability according to the density and not on the mobility. Using mobility to control the number of broadcast attempts helps obtaining full, or almost full coverage while keeping the algorithm simple.



(a) (b)

Fig. 2. Probability β vs. time for playground sizes of (a) 100m and (b) 200m. All 25 nodes generate and transmit 50 messages. Initial value of β=1.0.

3.3 Adaptive probabilistic broadcast algorithm - L2

In this section, we describe an alternative approach to achieve information dissemination in IPAC. We describe a second adaptive probabilistic broadcast algorithm (L2). We provide a preliminary discussion, introducing useful notations and ideas, as well as a detailed description of the proposed algorithm.

Let m be a message received by a node. We define t_m (reception time for message m), TTL_m (the TTL of message m). Upon receiving message m, the node should consider transmitting the message within time frame $[t_m, t_m + TTL_m]$. The message may be transmitted multiple times within this time frame, i.e., f_m times. The number of transmissions f_m must increase with TTL_m. The transmission period is $T_m = TTL_m/f_m$. The node, at the beginning of each transmission period, decides stochastically whether to transmit or not with probability β. The probability is calculated according to the number of neighbors, as well as changes in the node's neighbor list. Specifically, β starts from an initial value of β_0 and is adjusted dynamically according to network density. Another factor that influences β is node mobility and the number of messages retransmitted by neighbors. A minimum value of β_{min} is assumed to avoid having all nodes seizing transmitting.

Moreover, a relative mobility factor M_n is defined. Each node keeps a table of the nodes seen in its neighborhood and the table is updated when a node receives or overhears a packet. When a timer expires, the corresponding entry is removed from the table. When an entry is removed or inserted in the table, a change counter is incremented. The node periodically observes the change counter with period T_c. The counter is set to zero at the beginning of each period. At the end of each period, the

node decides to alter the mobility factor M_n depending on the observed number of changes N_c in the neighborhood:

$$M_n = (N_c - N_{mean})/N_{mean,}$$

where N_{mean} is the mean of the observed values of N_c at each round. In other words, M_n represents the deviation from the expected number of changes.

A network density factor D_n is also defined. The network density factor D_n can be altered periodically by observing the total number of nodes K_n present in the neighborhood table used to compute the mobility factor. Specifically:

$$D_n = (K_n - K_{mean})/K_{mean}$$

where K_{mean} is the mean of the observed values of K_n at each round. In other words, D_n represents the deviation from the expected number of neighbours.

Simulation layout

The basic setup involves $N = 25$ moving nodes which are initially randomly placed on a varying width squared playground. Nodes are moving randomly according to the following mass mobility scenario:

- At the beginning, nodes are randomly placed on the playground field.
- Each node changes its speed and direction a number of times. The interval of time between changes is normally distributed (~N(10, 0.5), figures in seconds.)
- At each change, every node turns by a certain angle (~N(0, 30), figures in degrees.)
- At each change, every node selects its new speed (~N(0.1, 0.1), figures in m/s.)
- When a node reaches an edge of the play ground, it bounces on it.

The goal of mass mobility is to take into account the inertia of moving objects. Table 5 describes the simulation parameters

Table 5**.**Simulation parameters.

N: number of nodes	25
P0: Number of packets sent by node 0 (for other nodes: none)	50
Traffic type	Exponential(10s)
TTL [s]	20
Backoff [s]	Uniform in [0, 1]
Transmission Periods	10,21
w: square size [m]	200, 300, 400
B_0	0.5, 0.7, 0.9
Node (average) velocity (m/s)	5,10,15, 20

Only node 0 sends packets. The setup regarding the messages *m* that are generated and arriving in networking level, as well as the metrics value that are monitored (*Success rate* and *Number of forwardings*) are similar to those of the section 3.1. We notice that for low density, the average number of transmissions per message as well as good-put gets lower. We would like to examine the values of β that the nodes derive. In Table 6, we summarize our findings. Results are shown for broadcast periods 10s and 21s (one and two broadcasts at most) and β_0=0.5 and indicate increased network coverage as node mobility (node speed) increases. This observation becomes more evident when the network is sparse in the case of w=400m. This means that mobility helps the dissemination process.

Table 6. Good-put and number of forwardings vs. mean speed and playground size. $\beta_0 = 0.5$. In each cell, the first value corresponds to a broadcast period of 10s and the second one to a broadcast period of 21s.

	Field side [m]	200	300	400
Average speed: 5 m/s	Good-put	0.993583, 0.901000	0.771667, 0.554000	0.417667, 0.294167
	Number of forwardings	23.312, 11.326	15.932, 7.422	8.572, 4.408
Average speed: 10 m/s	Good-put	0.991500, 0.897250	0.823500, 0.561167	0.487250, 0.287500
	Number of forwardings	22.566, 11.264	16.81, 7.51	9.708, 4.306
Average speed: 15 m/s	Good-put	0.996583, 0.921167	0.891333, 0.595583	0.505583, 0.292250
	Number of forwardings	22.86, 11.64	18.276, 8.034	9.504, 4.566
Average speed: 20 m/s	Good-put	0.993000, 0.937417	0.893083, 0.599833	0.550333, 0.305917
	Number of forwardings	22.498, 11.72	18.146, 8.324	18.146, 8.324

The main difference of algorithm L2 with the algorithm L1 is that it tries to infer mobility and computes the broadcast probability β accordingly. Moreover, while L1 relies on the detected number of neighbors to infer density and tunes the probability accordingly, L2 starts from a reference value β_0 and increases or decreases it according to the variation of density. The simulations show that the derived values of β heavily depend on β_0.

4 Conclusions and future work

Disseminating information within a wireless ad-hoc network calls for the use of a flooding technique. In IPAC, by proposing a probabilistic broadcast technique, we aim to reduce the amount of message transmissions. Simulation results show that for a given network density, there is a minimal probability of broadcast β_{min}, (percolation probability), which achieves full coverage. Trying different broadcast periods, we found out that the percolation probability decreases with the number of broadcast attempts. However, in terms of efficiency, using a low number of broadcast attempts and higher probabilities involves fewer transmissions than the opposite.

The simulations also showed the importance of implementing a congestion control mechanism. If the network creates too many messages, the mechanism will not allow full coverage but will help reaching the maximal possible coverage given the actual generated traffic. Mobility scenarios were simulated as well. Due to their significant randomness, we cannot conclude whether probability should be increased or decreased depending on the mobility.

We also propose an adaptive probabilistic broadcast algorithm (L1) based on network density inferred by the estimation of the number of neighbors. Simulation results prove that a good coverage can be obtained in very different conditions of network density, size and mobility. Moreover, the algorithm is able to detect high density situations and divide the amount of transmitted packets by more than 5 for a network of 25 nodes. Another advantage of the scheme is that it does not require control signaling. Hence the protocol overhead is minimal.

A variant of the adaptive algorithm (L2) is also introduced. The main difference with the first algorithm is that it tries to infer mobility and takes it into account when computing the probability of broadcast. Also, it starts from a reference value β_0 and increases or decreases it according to the variation of density. The simulations show that the derived values of β heavily depend on β_0. Our findings show that the first scheme is more likely to provide a good coverage irrespective of network density.

As future work we are planning to run simulations in order to optimize certain parts of the presented algorithms, such as congestion control (which could depend on the criticality of messages) and number of broadcasts according to mobility.

References

1. Demers, A.J., Greene, D.H., Hauser, C., Irish, C. W., Larson, J.: Epidemic algorithms for replicated database maintenance. (PODC 1987), pages 1–12, Vancouver, British Columbia, Canada (1987)
2. Franklin, M.J., Zdonik, S.B.: Dissemination-based information systems. Data Engineering Bulletin, 19(3), 20–30 (1996)
3. Eugster, P. Th., Guerraoui, R., Kermarrec, A. M., Massoulie, L.: From epidemics to distributed computing. IEEE Computer 37 (5), 60-67 (2004)
4. Vahdat, A., Becker, D.: Epidemic Routing for Partially-Connected Ad Hoc Networks. Duke University Technical Report CS-200006 (2000)
5. Rabiner, W., Kulik, J., Balakrishnan, H.: Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. MOBICOM 1999: 174-185 (1999)
6. Kulik, J., Rabiner, W., Balakrishnan, H.: Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks.Wireless Networks 8(2-3):169-185 (2002)
7. Estrin, D., Govindan, R., Heidemann, J., Kumar, S.: Next century challenges: scalable coordination in sensor networks. MOBICOM 1999: 263-270 (1999)
8. Eugster, P. Th., Guerraoui, R., Handurukande, S. B., Kouznetsov, P., Kermarrec, A. M.: Lightweight probabilistic broadcast. ACM TOCS 21 (4), 341-374 (2003)
9. Luo, J., Eugster, P., Hubaux, J.P.: Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks. In: Proceedings of the 22nd IEEE Conference on Computer Communications, pp 2229--2239, San Francisco, California, USA (2003)
10. Miranda, H., Leggio, S., Rodrigues, L., Raatikainen, K.: A power-aware broadcasting algorithm. In: Proc. of the 17th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pages 1–5, Helsinki, Finland, (2006)
11. Karp, R., Schindelhauer, C., Shenker, S., Vöcking, B.: Randomized Rumor Spreading. In: Proc. of the 41st FOCS, pp. 565-574. Redondo Beach, CA (2000)
12. Datta, A., Quarteroni, S., Aberer, K.: Autonomous Gossiping: A Self-Organizing Epidemic Algorithm for Selective Information Dissemination in Wireless Mobile Ad-Hoc Networks. ICSNW 2004: 126-143 (2004)
13. Anagnostopoulos, C., Hadjiefthymiades, S.: On the Application of Epidemical Spreading in Collaborative Context Aware Computing", ACM SIGMOBILE MC2R, (2008)
14. Varga, A.: OMNET++ Discrete Event Simulation System User Manual. (2006)