A Preliminary Evaluation of Backup Servers for Longer Gaming Sessions in MANETs

C. E. Palazzi
Dipartimento di Matematica Pura e Applicata
Università degli Studi di Padova
Via Trieste, 63 - 35131, Padova, ITALY
+39 049 827 1426
cpalazzi@math.uinipd.it

A. Kaiser, N. Achir, K. Boussetta
L2TI, University Paris 13,
99 Avenue J-B Clément,
93430 Villetaneuse, France
+33 1 49 40 40 62
nadjib.achir@univ-paris13.fr
khaled.boussetta@univ-paris13.fr

ABSTRACT
Nowadays, interactive online games and mobile devices have reached such a popularity they are obviously going to be a major success if combined. This can produce, for instance, Augmented Reality games played with real interactions among players connected as wireless nodes in an ad-hoc network. To enable this scenario, we propose a hybrid architecture that allows client-server games to be played in ad-hoc mode. In our architecture, one of the players is also the server of the game, whereas other backup server nodes are ready to become active servers in case of disconnection of the network or in case of low energy level of the currently active server. This allows player to be engaged in the game for longer time even with limited energy and continuous mobility. In this paper we discuss our idea and show preliminary experimental results that demonstrate its effectiveness

Categories and Subject Descriptors
C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *distributed networks, network communications, wireless networks.*

General Terms
Measurement, Performance, Design.

Keywords
Backup server, energy, MANET, online game.

1. INTRODUCTION
Multiplayer online games have become a major success among Internet users, generating revenues that surpass even those of movie industry. This success and the challenging technical issues related to their effective deployment, have attracted the interest of researchers and practitioners all around the world. Indeed, although this class of applications requires only a little amount of bandwidth, its performance is subject to strict per-packet delay requirements and to scalability issues that can be solved only through specific solutions and architectures.

Nowadays, the endemic diffusion of connectivity-enabled mobile devices (e.g., smart phones) fosters new gaming scenarios. For instance, think of a multiplayer game based on users' position and proximity: a game that is played outside, on the street or in courtyards. Players may gather around a certain spot and start there a gaming session based on their mobile phones' features (connectivity, GPS, gyroscope, etc.).

Another interesting and new gaming scenario is represented by the possibility to engage in Augmented Reality (AR) games through helmets and wireless connectivity so as to have players able to interact with the AR world and with each other [1, 2].

In both these simple but representative examples, players may increase their fun by engaging in an outdoor, connectivity- and proximity-based game. Indeed, players would be interested in extending their favorite games with the possibility to be played outdoor. It comes hence straightforward to imagine players exploiting ad-hoc connectivity among their devices to support game interactivity. However, this is not as simple as it may seem; in particular, we can identify the following three major issues.

First, the main gaming model today is client-server, with the client run by the player's PC and the server located remotely in the Internet [3-5]. If we want to ease the adaptation of existing successful online games, we need to project the client-server model on the scenario we are considering. Unfortunately, this is not simple since an ad-hoc network is general not connected to the Internet. Therefore, conversely from existing online games for home users, one of the players' devices has also to act as the server.

Second, mobile ad-hoc networks (MANETs) are prone to disconnections. Due to mobility, one or more nodes may get out of range of the original ad-hoc network, becoming unable to reach the game server node and continue playing. Even worse,

DISIO 2010 March 15, Torremolinos, Malaga, Spain.

the server node itself may get out of range for the rest of the network, thus interrupting the game session for everybody.

Third, whereas energy consumption is not an issue for home players, it certainly becomes a major concern when considering outdoor games based on the connectivity of small devices since the limited amount of energy stored in their batteries. This limited energy can be quickly consumed by game-related computation, visualization, and communication. Clearly, the worst energy consumption is experienced by the node that is also the server of the game session; that player's device will experience a much faster decrease of its energy reserve as it will be a central node for the whole MANET. In fact, it will have to receive all game events from other players, compute game state updates, and transmit these updates back to the other players.

In summary, popular online games could be even more fun if introducing some real world action through the adoption of mobile devices and ad-hoc network connectivity. However, the client-server communication model has to be adapted in a smart way to meet the MANET scenario. Furthermore, disconnection and energy consumption may impede long gaming session for some or many players.

We propose here a hybrid architecture, where some popular online game is adapted to be played on the street through mobile devices and ad-hoc connectivity. To solve the aforementioned issues, we propose to have one *active server* and some *backup servers* ready to substitute the former if experiencing low energy level or disconnections. The idea takes inspiration from the fact that distributing a process may increase its efficiency [6]. In our solution all the servers collect game events sent by other players and update their game state view. However, only the active server forwards the current game state to all the players (including the backup servers); this limits the energy consumption of backup servers until they are called to become active.

In this paper we present our solution and preliminary experimental results that confirms the benefits of the idea. Our results are preliminary since they focus just on energy consumption without quantitative measurements of connectivity improvements. Moreover, our simulations consider only transmission-related energy consumption without including the energy used for game actions and updates. These limitations will be addressed in a future version of this work. However, current results already permits to intuit the advantages brought by our solution, as they can be seen as a lower bound of the actual performance benefits.

The rest of the paper is organized as follows. In Section 2 we discuss our solution within the considered multiplayer game scenario. Section 3 reports on how we modeled this scenario in NS-2, whereas Section 4 specifies all the parameters chosen. We present experimental results in Section 5. Finally, in Section 6, we conclude this paper and suggest future extensions.

2. BACKUP SERVERS IN MANETS

We are considering a multiplayer, interactive game played by wireless nodes in an outdoor environment and connected through ad-hoc connectivity. Nodes could be smart phones, PDAs, tiny laptops, or other portable devices customized for this kind of application (e.g., helmets for AR).

Exploiting ad-hoc connectivity frees the users for the need of Access Points (APs), thus facilitating the emergence of a gaming session whenever a group of players gather around a certain location. This kind of connectivity could seem to be directly linked with having a peer-to-peer (P2P) game architecture. Yet, current online games generally adopt a client-server architecture. Therefore, to ease a direct porting of existing games of success it is better to adopt a client-server paradigm even in our ad-hoc network.

A client-server architecture is a straightforward solution with a fixed infrastructure and dedicated game servers. Unfortunately, in the considered MANET scenario, one of the nodes must also act as server and, by doing so, it will drain battery more quickly than the other players: it receives and sends more packets than any client and also need to compute the updated game state. This will result in shorter gaming sessions: once the server is out of energy, the game will be interrupted. Furthermore, due to mobility, the network may become disconnected, splitting in two or more groups and having just the group of players still connected to the server-player able to continue the gaming session.

In order to avoid this, we have more than one server: at each time only one is active, while the others are kept synchronized. The active server is periodically selected among all servers in a round robin way, thus distributing the battery consumption over all servers. Note that a node that acts as a server is also a client and thus participate in the game.

Because of node movements, most of the time a node will not able to reach all other the nodes but only a few ones. We can see each node as a member of a *partition* - a subset of nodes that reach each other. A partition could change at any time: for example a node could go out of its partition and enter another partition; or an isolated client can join a partition and begin to play. We note that we allow at most one active server in each partition.

Communications between nodes are made in *unicast*, like actual games; in particular, when the active server has to send a packet to other nodes, it sends a copy for each destination, without relying on broadcast or multicast. Clearly, broadcast/multicast solutions can be added later; their integration with our solution can generate even better performance.

For the sake of clarity, now we summarize node types in the simulation:

- **active server**: a node that holds game status, communicates with all nodes in its partition and keeps synchronized backup servers;
- **backup server**: a server that plays and can be chosen as active server in the future;
- **client**: a node that only plays the game.

3. SIMULATION ASSESSMENT

During the development of this work we have utilized the following software:

- NS-2 version 2.33 with modified DSDV code [7, 8];
- BonnMotion version 1.3a with some fixes [9];

3.1 Partition Management

The management of the connectivity in the simulated MANET is done through a specific Tcl class we implemented to contain active and backup server identifiers, scheduled time for active server change, identifiers of clients that are reachable by the active server. Furthermore, this class also creates a server CBR flow for each active server UDP agent and a client CBR flow for each of the other nodes.

Backup servers can become active servers in case of:

- *active server rotation*: another active server must be chosen because the current one has been active for its quantum;
- *topology variation*: some nodes are not reachable anymore by the active server thus needing to elect their own active server.

Among these two cases, the second one is more complex. If a group of clients is not reachable anymore by the current active server, they could generate a MANET on their own and continue playing. Even better, if within this new MANET there is also a backup server, that seamlessly becomes the new active server, thus avoiding the need to interrupt the ongoing game session (clearly, the number of players will be reduced with respect to pre-partition, unless introducing artificial players).

For backup servers the situation is the same as previous one, but we have also to understand how to create new partitions when backup servers cannot reach their active servers. The solution we come up with is reported in the following.

1. A *reachability matrix* is created. In this matrix, component *[i, j]* is *1* if node i can communicate with node j and *0* otherwise. To compute this matrix we exploit the fact that, if G is an adjacency matrix for a graph and we consider each node adjacent to itself, G^r is a matrix in which entry $g^r[i, j]$ is greater than *0* if there is at least a path from i to j of length at most r. In our case $g[i, j] = 1$ if j is in the transmission range of i and the reachability matrix is simply G^r, where r is the number of nodes minus 1. Indeed, if a node wants to communicate with another it will hop through almost all other nodes. Note that the matrix is symmetric.
2. Periodically, the system checks whether the backup servers can communicate with their active server; if not, a backup server can become an active server of a new partition.
3. In case of two backup servers that no more reach the active one but reach each other, only one becomes the active server for that partition. Similar, if two active servers can reach each other then the partitions they belong to are merged.
4. For each active server we create a new partition, using the reachability matrix. Note that this way some nodes can change partition and others can become isolated. In case an active server was in a old partition, we carry on the time in which it ceases to be active and the order of backup servers (if they are still reachable). We can have partitions with only the active server: we have to create them for the correctness of the algorithms that manage the partitions.
5. At this point old partitions are deallocated from memory.

3.2 Movement Generation

We have considered two main movement scenarios for our MANET:

- **Grid** model: a simple model in which nodes do not move but stand still forming a grid.
- **Reference Point Group Mobility (RPGMM)** model: nodes are distributed in groups; the motion of the group center completely characterizes the movement of its components.

The first one is a grid of nodes, as depicted in Fig. 1, with no movements at all; this configuration allows us to clearly understand properties of tested solutions without having them affected by mobility and routing algorithms. Instead, the second one is a mobility model that is known to be more realistic in the real world with respect to a pure random mobility model. With RPGM, nodes are distributed in groups and the motion of the group center influences the movement of its nodes as would happen in a team-based game or in a game where certain players have to catch other players. To add the RPGM to NS-2 we have utilized a tool called BonnMotion [9].

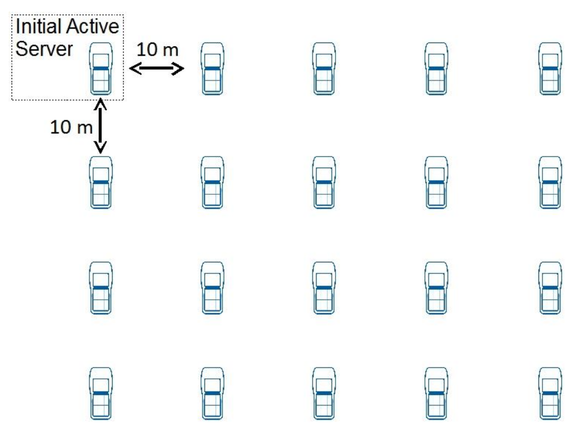

Figure 1. Nodes' position in Grid model.

Table 1. Simulative parameters

Initial MANET size	100×50 m^2
Number of nodes	20
Number of servers	8
Movement models	Grid, RPGM
Antenna height	1.5 m
MAC protocol	IEEE 802.11g
Routing algorithms	AODV, DSDV
Transmission Range	20 m
Server-generated flow	200 bytes every 50 ms
Client-generated flow	40 bytes every 300 ms
Server hold time	5 min, 10 min, no hold time
Simulation time	2 hours

4. SIMULATIVE PARAMETERS

We summarize in Table I the most significant parameters and variables that we have used in our simulations; for the sake of clarity, we comment some of them.

The MANET is composed of 20 nodes that represents participants to the gaming session. Among them, 8 embodies also potential game servers: in any connected group, 1 will be the active server and the others are backup server. Initially the MANET topology can be represented as a regular 5×4 grid with adjacent nodes at 10 m from each other.

Gaming traffic is composed of game actions generated by clients and game updates transmitted by servers. As inspired by existing literature of online games, server CBR flows are composed by packets of 200 bytes (i.e., game state updates) sent every 50 ms, whereas client CBR flows are composed by packets of 40 bytes (i.e., game events generated by the user) sent every 300 ms [10].

Mobility is achieved by moving the nodes at walking speeds in the range from 1 to 2 m/s. As said, two different movement models have been considered : Grid model and RPGM model.

The Grid model is a very simple scenario useful to clearly understand the energy and networking dynamics.

Instead, the RPGM model is more realistic for representing the mobility of players with the considered outdoor game [11]. It can represent a gaming paradigm where a player is chased by others or where players are grouped in teams and move together. In our simulations we created 3 groups, one with 6 components (with 2 potential servers) and two with 7 components each (with 3 potential servers each); every group component resides in a circle of 30 m of radius and centered at the group center. Each group stops for at most 60 s in a place and moves again toward a new attraction point.

We have placed the antenna height of each node at 1.5 m above the ground, i.e., a smart phone held in the players' hands, and a transmission range oscillating around 20 m, which represents a realistic value for an IEEE 802.11g Wi-Fi handheld device running on battery [12, 13].

Routing algorithms have an important role in MANETs as they can have an impact on transmission performance and, in particular, on energy consumption. Therefore, our experiments were run exploiting both DSDV and AODV, which are the main routing algorithms for ad-hoc networks. Before running the experiments, we have corrected the default DSDV module as suggested in [8].

Our solution promises to ensure longer gaming sessions in MANETs thanks to the use of backup servers, which allow a more fair distribution of energy consumption, rather than being the burden of just one server. Therefore, we focus on the study of server energy consumption with respect to the time a server remains active before passing the task on a backup server. From here on, we refer to this time as the *hold time*; in this paper we present the outcomes when adopting our solution with a hold time of 5 or 10 minutes, comparing them with a default solution of having a fixed game server.

5. EXPERIMENTAL RESULTS

We evaluate the benefits of our solution in ensuring longer gaming session through the evaluation of energy consumed by nodes in the MANET.

First, we start with the simple Grid model and compare a traditional, single server solution with our solution exploiting backup servers. In the latter case, the hold time of the active server has been set equal to 5 and 10 minutes in two different experiments.

Results corresponding to the use of AODV and DSDV as routing algorithms are shown in Fig. 2 and Fig. 3, respectively. In both figures, the starting active server is represented by node *0*. With the traditional scheme (i.e., *No hold time* in the figure), node *0* remains the game server for the whole duration of the gaming session. As it is evident, this results in a much higher energy consumption for that node with respect to the others. The server node will hence run out of energy sooner than the other nodes interrupting the game session. This outcome is similar regardless of employing AODV or DSDV.

Moreover, we considered the worst case scenario, i.e., the configuration where the server is located at one corner of the MANET grid, thus having packets traversing the whole MANET to reach the opposite corner. This higher number of transmissions involves higher energy consumption. If, thanks to our solution, after the hold time the server is changed, it may happen that the new active server is closer to the center of the MANET thus reducing the average number of hops/transmissions required to propagate messages over the whole network. This is confirmed if we look again at Fig. 2 and Fig. 3. In fact, not only is the energy consumption more fairly distributed among the nodes when employing our backup server solution, but also the total energy consumed by each node is reduced. This happens because the selection of a new active servers creates a network where the active server is closer to the center of the MANET; the nodes of the network will hence be requested to perform less packet forwarding thus reducing their energy consumption.

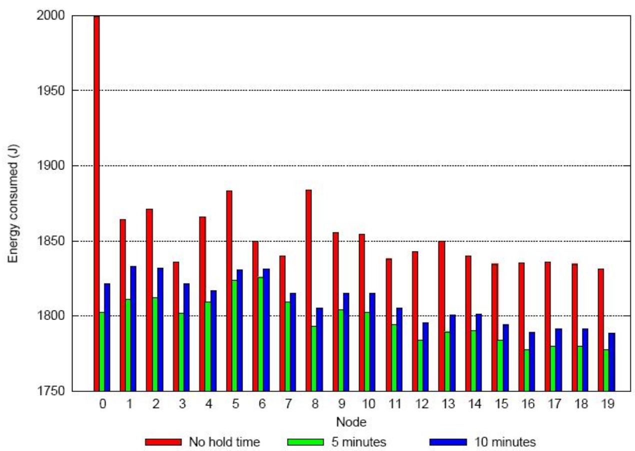

Figure 2. Energy consumption with AODV and Grid model.

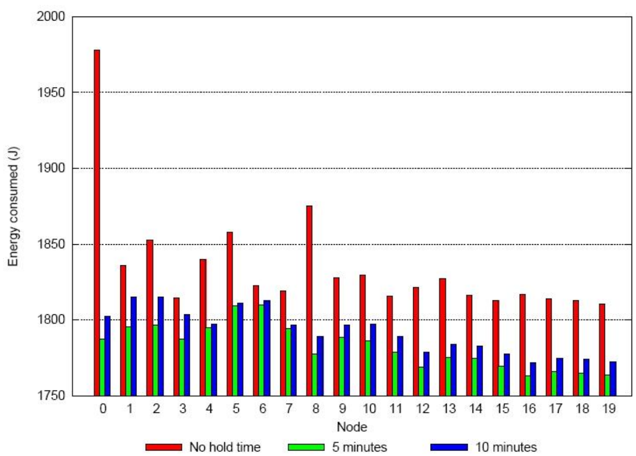

Figure 3. Energy consumption with DSDV and Grid model.

When nodes move, the consumption due to routing algorithms and hop routing becomes significant, even much higher than the one caused by transmission/reception of game packets. It is hence interesting to evaluate the benefits of our solution considering realistic mobility by including the RPGM model in our simulations and random initial position of the game server(s).

To this aim, Fig. 4 and Fig. 5 reports on the energy saved when utilizing our backup server solution in place of a traditional scheme. As can be seen, our solution improves the battery usage in all situations, saving from about 5 to 10 % of energy. The energy efficiency seems better when considering 5 minutes as a hold time instead of 10 minutes since having smaller hold times foster a more fair utilization of backup server resources.

Presented tests embody only a preliminary evaluation of our solution and achieved results are actually a lower bound for the factual improvement that can be achieved through backup servers. Indeed, with the traditional scheme, it frequently happens that some group of nodes gets disconnected from the server, thus interrupting their gaming session and packet transmissions. Instead, with our scheme this almost never happens thanks to the presence of backup servers that can become active servers in case a group of nodes disconnects from the MANET. However, since disconnected nodes stop transmitting and receiving packets, they also stop consuming energy; therefore, a side effect of these disconnections is that they significantly lower the total amount of energy consumed by the traditional scheme. Therefore, the energy improvement showed by Fig. 4 and Fig. 5 is currently affected by the energy spared by these disconnected nodes. By considering only nodes that are able to continue playing for the whole game session and taking their average energy consumption, our solution should show a better, even much better, energy efficiency with respect to a traditional scheme.

Consequently, we are now working on eliminating this bias. Furthermore, we are adding computational energy cost to our simulation as this element is an important factor affecting server nodes. We are also quantitatively evaluating the connectivity degree of the MANET expressed in terms of nodes able to seamlessly play to the same game session regardless of mobility. Finally, we plan to perform a rigorous evaluation of backup server election and hold time setting. All these new results require additions to our simulative code and will be shown in a future extension of this work.

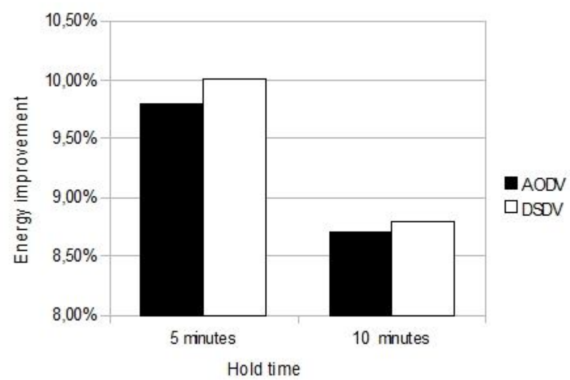

Figure 4. Energy improvement of backup server solution with respect to a traditional scheme; Grid model.

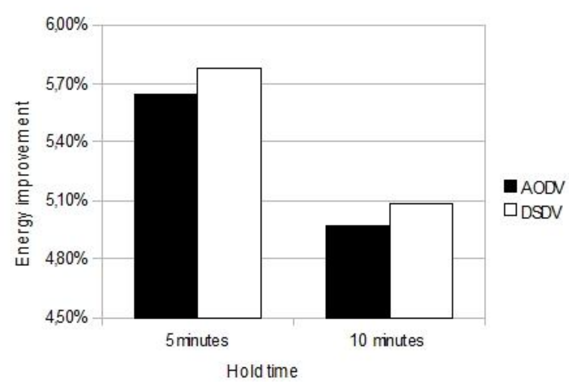

Figure 5. Energy improvement of backup server solution with respect to a traditional scheme; RPGM model.

6. CONCLUSION AND FUTURE WORK

The combination between interactive online games and mobile devices represents a match made in heaven that fosters a new generation of games played through real interactions among players connected as wireless nodes in a MANET. To enable this scenario, we have proposed a hybrid architecture that allows client-server games to be played in a MANET. In each group of players, one node is also the server of the game, whereas other backup server nodes are ready to become active servers in case of disconnection of the group. Furthermore, to avoid unfair energy consumption, a backup server substitute the active server after a certain quantum of time. Our solution allows player to be engaged in the game for longer time, even with limited energy and continuous mobility.

In this paper, we have discussed our idea and shown experimental outcomes that demonstrate its effectiveness. As preliminary results, we have focused on energy consumption achieving encouraging feedback; we are currently performing quantitative evaluation of connectivity improvement and interactivity degree. Other possible extensions of our work include the combination with new routing and MAC layer protocols, multicast/broadcast solutions, vehicular ad-hoc network (VANET) scenarios, and different applications such as participative learning and digital adventures brought on the streets [14-17]. It would be particularly interesting to compare our solution with others borrowed from sensor network literature related to energy consumption [18]. Moreover, we also plan to refine our simulations by varying the number of clients and servers, evaluating the impact of servers' position with respect of the MANET topology.

7. ACKNOWLEDGMENTS

Our sincere gratitude to Stefano Gazzola, Eric Miotto, and Roberto Pordon for their technical help in developing the experimental testbed.

8. REFERENCES

[1] S. Singhal, M. Zyda. 1999. Networked Virtual Environments: Design and Implementation, Addison Wesley.

[2] A. D. Cheok, K. H. Goh, W. Liu, F. Farbiz, S. L. Teo, H. S. Teo, S. P. Lee, Y. Li, S. W. Fong, X. Yang. Human Pacman: a Mobile Wide-Area Entertainment System Based on Physical, Social, and Ubiquitous Computing. In *Proc. of the 2004 ACM SIGCHI Intl. Conference on Advances in Computer Entertainment Technology*, Singapore, Jun 2004.

[3] S. Wright, S. Tischer. Architectural Considerations in Online Game Services over DSL Networks. In *Proc. IEEE International Conference on Communications - (ICC'04)*, Paris, France, Jun 2004.

[4] E. Cronin, A. R. Kurc, B. Filstrup, S. Jamin. An Efficient Synchronization Mechanism for Mirrored Game Architectures. *Multimedia Tools and Applications* 23, 1, Jan 2004, 7 – 30.

[5] A. Ploss, S. Wichmann, F. Glinka, S. Gorlatch. 2008. From a Single- to Multi-Server Online Game: A Quake 3 Case Study Using RTF. In *Proc. of ACM ACE 2008*, Yokohama, Japan, Dec 2008.

[6] F. Corradini, R. Gorrieri, M. Roccetti. Performance Preorder: Ordering Processes with Respect to Speed. In *Proc. of Mathematical Foundations of Computer Science*, LNCS 969, p 444-453, Springer Berlin, 1995.

[7] The Network Simulator - NS-2, [Online]. Available: http://www.isi.edu/nsnam/ns

[8] DSDV in ns-2, [Online]. Available: http://www.winlab.rutgers.edu/\~zhibinwu/html/DSDV_ns2.html

[9] BonnMotion, [Online]. Available: http://tinyurl.com/a9y8jh

[10] J. Farber. Traffic Modelling for Fast Action Network Games. *Multimedia Tools and Applications* 23, 1, Jan 2004, 31 - 46.

[11] T. Camp , J. Boleng, V. Davies. A Survey of Mobility Models for Ad Hoc Network Research, *Wireless Communication & Mobile Computing* 2, 5, Feb 2002, 483 - 502.

[12] IEEE, "802.11-2007, IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks- Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", 2007.

[13] P. J. Vincent, M. Tummnala, J. Mceachen. A New Method for Distributing Power Usage across a Sensor Network. In *Proc. of the IEEE SECON 2006*, Reston, VA, Sep 2006.

[14] S. Ferretti, S. Mirri, L. A. Muratori, M. Roccetti, P. Salomoni. Cooperative Multimedia Management for Participative Learning: A Case Study. *New Review of Hypermedia and Multimedia*, Taylor & Francis 14, 2, Dec 2008, 177 - 197.

[15] A. Amoroso, M. Roccetti, M. Nanni, L. Prati. VANETS without Limitations: An Optimal Distributed Algorithm for Multi-Hop Communications and Some Preliminary Results. In *Proc. of the IEEE NIME'09/CCNC 2009*, Las Vegas, NV, USA, Jan 2009.

[16] M. A. Alberti, D. Maggiorini, P. Trapani. A Semiotic Approach to Narrative Manipulation. In *Proc. of the 29th International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2002)*, San Antonio, TX, USA, Jul 2002.

[17] G. Marfia, G. Pau, M. Roccetti. On Developing Smart Applications for VANETs: Where are we now? Some Insights on Technical Issues and Open Problems. *In Proc. of the IEEE UMSA'09/ICUMT 2009*, St Petersburg, Russia, Oct 2009.

[18] H. Zeghilet. Performance Improvement of Passive Clustering Algorithm in Wireless Sensor Networks. In *Proc. of 4th International Conference on Networked Sensing Systems (INSS 2007)*, Braunschweig, Germany, Jun 2007.