Demonstrating the feasibility of an Autonomic Communication- Targeted Cross-Layer Architecture

#Mohammad Abdur Razzaque, Paddy Nixon and Simon Dobson
System Research Group
School of Computer Science and Informatics
UCD, Dublin, Ireland
{abdur.razzaque, paddy.nixon, simon.dobson}@ucd.ie

Abstract

Layered architectures are not sufficiently flexible to cope with the dynamics of wireless-dominated next generation communications. Cross-layer architectures may provide a better solution: allowing interactions between two or more non-adjacent layers in the protocol stack. Architectures based on purely local information will not be able to support autonomic communications like self-behaving systems. A new cross-layer architecture which provides a hybrid local and global view, using gossiping to maintain consistency has been proposed in [1]. This paper demonstrates the feasibility of that architecture through two different prototype implementations. Context-awareness is a key issue in autonomic communications and both the implementations directly or indirectly uses it. The first implementation uses contextual information to control Transmission Power of a mobile node and the second one uses it to manage traffic automatically. Results show that context-awareness is possible through our architecture and proves the potential of the proposed architecture

1. INTRODUCTION

Existing layered design is insufficiently flexible to cope with the dynamics of wireless communications. Recent studies [2] show that careful exploitation of some cross-layer protocol interactions can lead to more efficient performance of the transmission stack – and hence to better application layer performances – in different wireless networking scenarios. Cross-layer design breaks away from traditional network design, where each layer of the protocol stack operates independently and exchanges information with adjacent layers only through a narrow interface. In the cross-layer approach information is exchanged between non-adjacent layers of the protocol stack, and end-to-end performance is optimized by adapting each layer against this information.

Recent initiatives in autonomic communications [3] show that there is a need to make future networks self-behaving, in the sense that they work in an optimal way with "endogenous" management and control, with minimum human perception and intervention. To attain this self-behaving system within existing strictly-layered approaches may be possible to a certain extent, but will not leverage all the possible optimizations. Cross-layer architectures are better suited to achieving the self-optimization, self-organization and other "self-*" properties targeted by autonomic approaches. Moreover, self-organization, self-healing and network-wide self-optimization requires network-wide information (or at least direct neighbors) [1, 4].

Most existing cross-layer architectures (including GRACE [5], WIDENS [6], MobileMan [7] and ECLAIR [8]) rely on local information and views, without considering the networking context or views which may be very useful for wireless networks and autonomic communications. Only CrossTalk [9] is based on local as well as network-wide global view (partially). POEM [10] is the only architecture considering self-optimization that could be helpful for autonomic communication. Collecting and maintaining network-wide, global statistics can be expensive, while global actions are hard to co-ordinate; however, the effects of such systems can often be dramatic, and they can address problems that are difficult to detect, diagnose or solve using purely local information. To explore the impact of this idea, [1] proposes a new cross-layer architecture that is based on building and maintaining a view of the network's state and constraints, utilizing gossiping for gathering information from neighboring nodes.

A Network-wide view is a summation of summarized information regarding all the nodes in the network. To generate a network-wide view, the node-wide local views are generated first. This information is summarized and distributed around the network using gossiping algorithms. For initial prototype implementations of the architecture of [1]; this paper considers the local view-based cross-layering interactions. This paper includes an overview of that architecture with two prototype implementations and associated results. Context-awareness is one of the key

1-4244-0715-8/06/$20.00 © 2006 IEEE

properties of autonomic communications [3]. Therefore the prototype implementations are focused on demonstrating context-awareness.

An overview of the autonomic communication-targeted cross-layer architecture is presented in section 2. Section 3 presents motivating application areas of the architecture. Prototype implementations and associated results are presented in section 4 and section 5 concludes with some directions for future work.

2. AN AUTONOMIC COMMUNICATION-TRARGATED CROSS-LAYER ARCHITECTURE

Realizing the importance of cross-layering – and specifically cross-layering architectures with a network-wide view – in next-generation communications, an alternative cross-layering architecture based on a combination of node-wide (local) and network-wide (global) views has been proposed in [1]. The key distinction between this architecture and most other cross-layer architectures is that it can not only utilize a node-wide local view for optimization, but it can also use a network-wide view obtained through gossiping.

A. Overview of the Architecture

In conjunction with the existing layers, a knowledge plane is the key element of the architecture. Direct communication between layers and a shared knowledge plane across the layers are the two widely used cross-layer interactions policies [2]. Because of the improved separation and management possibilities we prefer to utilize the knowledge plane for the architecture. The following are the main elements of the architecture (as shown in figure 1):

Existing TCP/IP layers: These provide normal layering support when it is necessary. It also provides the different layers' information to the knowledge plane, allowing it to maintain a local view of the node. This allows full compatibility with standards and maintains the benefits of a modular architecture, as it does not modify each layer's core functionality.

Contextors for different layers: Each layer in the existing protocol stack will have a corresponding contextor, which will act as their corresponding interface between the layer and the knowledge plane. Each of these contextors will act as a "tuner" between a layer and the knowledge plane. Possible functionality for manipulating protocol data structures is built in to the contextors; no modification is required to the existing protocol stack. This facilitates incorporation of new cross layer feedback algorithms with minimum intrusion. A contextor will be responsible for reading and finally updating the protocol data structures when it is necessary.

Knowledge Plane: A common Knowledge Plane (KPlane) database is maintained to encapsulate all the layers' independent information as well as the network-wide global view, which can be accessed by all layers as needed. For modularity it maintains two entities responsible for maintaining the local and global views. Interaction between different layers and the KPlane through contextors can be reactive (responding to changing context) or proactive (anticipating changes and provisioning accordingly). Generally the interactions between different layers and the KPlane are event-oriented, which suggests a reactive scheme; on the other hand, the KPlane can maintain a model of the network and act autonomously to issue its own events. This leads to improved performance if the model leads to a correct proactive adaptation, but can be detrimental if the projection is wrong. In our architecture we are considering the database with reactive interaction policies as shown in figure 1. The KPlane consists of the database and necessary optimizing algorithms. The database is separated into local view and global view for isolation and management purposes, although it appears unified to clients.

Gossiping: Gossiping is considered as one of the most promising data-dissemination mechanisms in peer-to-peer or distributed systems. There are number of algorithms that can be classified as *reactive*, *proactive* and *periodic*. As there are few comparative performance studies amongst these algorithms, it is difficult to choose the most suitable algorithm for a particular purpose. In our case we propose a periodic gossiping approach, possibly with out-of-band "immediate" signaling for important changes. The gossiping service is built on top of existing TCP/UDP, and is responsible for gathering information from other nodes to generate the network-wide view at the host node. At each exchange the gossiping service chooses another node in the system (either randomly or with some weighted preference) and exchanges its local state with that node. In this architecture we will consider a gossiping exchange as an application-level event which will trigger the KPlane to take the necessary actions.

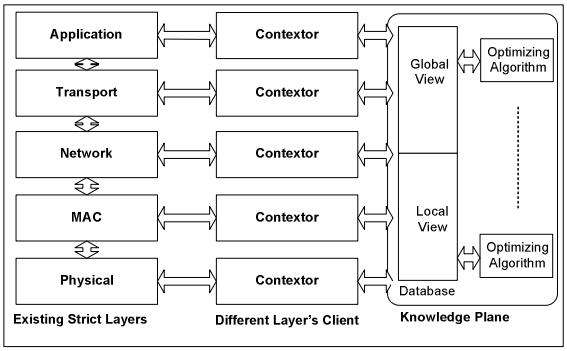

Fig.1: An Autonomic Communication-Targeted Cross-Layer Architecture

3. MOTIVATING APPLICATION AREAS

The two main broad objectives of this architecture as mentioned in [1] are to improve wireless performances and support autonomic communications. Within these two broad areas we are targeting some specific applications which could be implemented through the architecture.

Autonomic Communications: An autonomic communication is a self-behaving system with properties such as self-healing, self-configuration, self-organization, self-optimization, context-awareness, etc. Out of these self* behaviors we will restrict ourselves to self-organization, self-optimization and context-awareness. As mentioned in [1], most of the self* behaviors are inter-related and if we could justify the architecture for say self-organization then it may indirectly support self-healing and vice versa. In this paper we implement the architecture for the node-wide context-awareness. An application scenario of self-organization has been described in [1].

Wireless performance improvement: With the possible cross layering between TCP and other lower layers (Network/MAC) it is possible to improve the TCP performances and this is one of the targeted application areas of our architecture.

Alongside the above two application areas this architecture could be used for attaining QoS in wireless networking. Cross-layer architectures can capture and respond to top-down user demands or contexts which could provide better QoS.

4. IMPLEMENTATION AND RESULTS

In this section we apply the above architecture to two example scenarios: transmission power control and context-driven traffic management by profile. For the implementation, the discrete event network simulator ns-2 [14] is used. The main objective of these implementations is to show that integration of context-awareness (node-wide) in wireless environments through the proposed architecture is possible and study its impacts. The reason for selecting context-awareness for the initial implementations is that it directly or indirectly supports both of our broad application areas (mentioned earlier). The first implementation uses contextual information to control Transmission Power of a mobile node and the second one uses user level profile information to configure the internet connections and manage traffic automatically. In both of implementations simple algorithms are used as the goal of this work is to justify the architecture not to deal specific algorithms like [11, 12, and 13]. The concerned wireless environment is a Wireless LAN with a fixed Access Point (AP) and a number of mobile nodes (MNs) moving around within the coverage range of AP.

A. Context-based Transmission Power Control

For the first implementation a naive Transmission Power Control (TPC) algorithm for Wireless LAN (WLAN) is considered. The main principle of TPC is to transmit a packet with a power which could be sufficient enough for the receiver to receive it properly depending on the context/situation instead of transmitting all the time with maximum power. For TPC one can utilize one or more parameters such as distance between the nodes (e.g. distance between AP and Mobile Node), user density, connectivity, packet dropping frequency, neighbor node's power level, etc. Based on these parameters TPC selects power from a discrete set of powers. This work is considering almost the same assumptions used in [11, 13] five power levels (Plevel) of 4.8, 10.6, 36.6, 115.4, and 281.8mW to support corresponding transmission ranges of 90, 110, 150, 200 and 250m, respectively.

Two different scenarios are considered for TPC. The first scenario considers distance between the nodes (positional context) and packet dropping probability (a derived context from packet retransmission tries) to update transmission power. The second scenario considers only packet dropping frequencies, which could indirectly represent location/situation context. Figure 2 shows a snapshot of the implementation of scenario one (power-up) with step-by-step cross-layering interactions: (1) KPlane periodically checks the retransmission tries (Retrns_Tries) of a packet through the MAC layer's contextor (2) if the Retrns_Tries>=3, KPlane gets the positional information of nodes from the Application Layer (with the support of GPS: Global Positioning System) through the contextor and calculates the distances. (3) KPlane gets the current power level and current power (Pt_) from the MAC and Physical Layers through their contextors respectively. (4) if the current Plevel is lower then the calculated distance's corresponding Plevel then KPlane increases the Plevel by one step and sets the corresponding level's power. Finally, it updates the MAC and Physical layers with new settings and transmission continues with new settings.

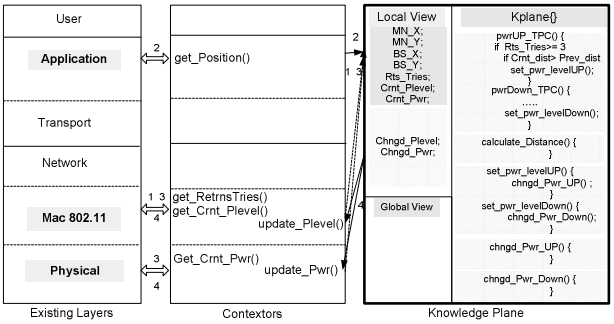

Fig.2: A snapshot of the implementation of Context-based Transmission Power Control

The results of two scenarios and no TPC are shown in figure 3 & 4, which represents graphs between distances vs. throughput and distances vs. Power respectively. In the simulation, mobile nodes are moved from 1m to 300m during a simulation period of 40s. Without TPC a MN is transmitting with a constant power of 281.8mW to cover a range of 250m. As shown in figure 3 (No_TPC) after 250m all packets are dropped. On the other hand in the case of TPC it transmits with variable (5 levels) power depending on the contexts (positional) and situations. Within two TPCs, the one with the application layer feedback regarding positional context (Dist_Retrns_Tries) is showing little better performance then its counterpart. This is obvious, as first TPC is using direct positional contextual and retransmission information whereas

the second one only uses indirect positional information derived from packet dropping frequencies to set the power up. Different treatment is needed for lowering the power levels in TPC when a node is transmitting with a power higher than is required. Even two scenarios needs different policies to do so. For the first scenario a periodic check of the distance can be used. For the second scenario only a trial and error method can be used. In the case of the second scenario, as we are allowing few (3) packet drops so there are ripples in the throughputs as shown in the graph (figure 3: Drp_frq_bsd_). This is logical because just before every power level increase there are few packet drops and it regains its normal throughput levels after a while. Figure 4 shows the corresponding power used in the figure 4 to support different distances. It is clearly shows the benefit of TPC. These results justify two points: (i) with the help of more useful information from different layers better optimization is possible and (ii) with the help of multiple items of direct contextual information more reliable and realistic performances are possible than single item of derived contextual information [15]. But interdependencies between the layers could be a problem.

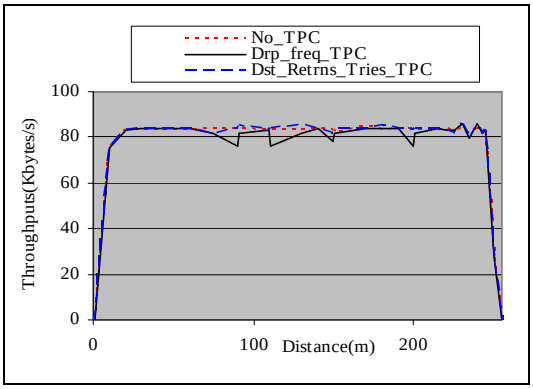

Fig.3: Comparison between no-TPC, TPC with dropping frequencies only And TPC with distance & retransmission tries

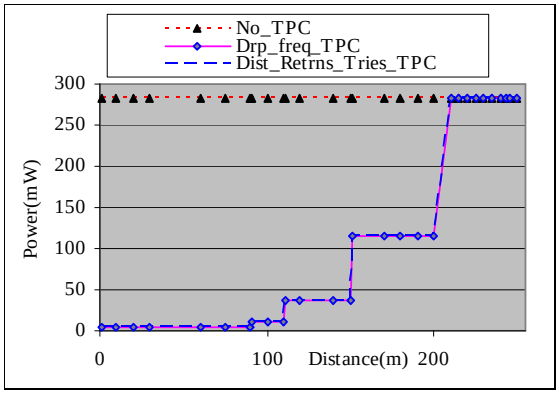

Fig.4: Powers used in figure 4 to cover different coverage distances

B. Context-driven traffic management by profile

Although "User" is not a layer in existing strict layering system, every layer is working directly or indirectly to provide services to user. Other than the Application Layer the rest of the layers do not understand or get the users requirements directly and react accordingly to satisfy their demands. User requirements and corresponding context should be taken into account to enhance the user perceived QoS. The motivation for this is that the user decision could be contrary to the system decision but it could lead to improved user satisfaction and thus we are adding User layer on top of all the layers (figure 2). Users' preferences or priorities can be defined through a profile, based on different situations. But with the existing strict layering it is not possible to deliver this priority information to relative layers; cross-layering could be helpful and in this implementation proposed architecture will be used to do so.

Consider a user in a location at the edge of WLAN's coverage with his laptop and he is downloading two files: one is important FTP file (7.7Mbytes) and the other one is a less important music file (8Mbytes). At the middle of downloading there is a warning about the battery and unfortunately there is no power connection nearby to charge it. In this critical situation if he continues downloading both files he might be end up with nothing finished. One can solve the problem by closing the less important connection. But with the help of cross-layer architecture like ours and integration of context-awareness it could be done automatically.

Figure 5 shows the implementation snapshot of a context-driven traffic management by profile and solve the earlier mentioned problem. Stepwise interactions for this implementation are: (1) KPlane periodically gets the battery power level from Physical Layer (2) if power level is lower than a critical value then it gets the priority levels for the different internet connections from the user layer (3) KPlane gets the current receiver window sizes for the connections from the Transport layer (TCP) (4) finally using priority levels KPlane calculates the new receiver window sizes for the connections and updates the Transport layer accordingly. Figure 6 shows the problem when there is no priority setup, as single file download is an atomic action so less than 100% finished means finished nothing. It shows that after 180s all the packets are dropped due to no power and resulting zero download even with 92% and 86% finished for two connections respectively. Figure 7 & 8 show the results for context-driven traffic or connections management by profile. In figure 8 Connection_1 is the higher priority FTP download whereas Connection_2 is the lower priority music file download. As shown in the figure 7 & 8, approximately at 5s simulation time there is a power warning. There is little difference between the two solutions. For the solution of figure 7: after the warning KPlane sets the new receiver window sizes based on user level's priority. Here connection_1 has the higher priority level than connection_2 and connection_1 successfully finished downloading before (128s) power failure and after that connection_2 gets full bandwidth to download but it failed to finish before the power failure around 180s. For the solution of figure 8: after the

warning KPlane assigns the full bandwidth to the connection_1 and close the connection_2 and connection_1 successfully finished downloading around 96s which is well before the power failure. As second solution is closing the connection_2 just after the warning and no scope for unsuccessful finish of it, which is saving little power compared to the first solution. If we look at the graphs before and after the priority settings, the sum of the two connection's throughputs is almost same, which signifies that proportionality of the connection bandwidth share is working properly based on profile-based priority settings. Similar to previous section result these also show that cross layering could be used to utilize for context-awareness and automatically manage internet configurations and traffic.

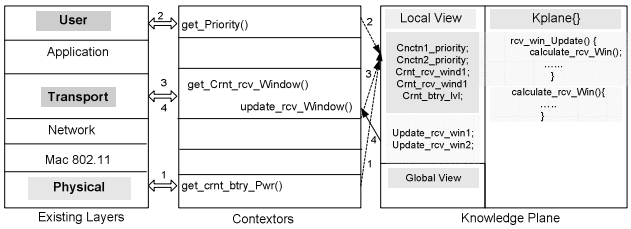

Fig.5: A snapshot of the implementation of context-driven traffic management by profile

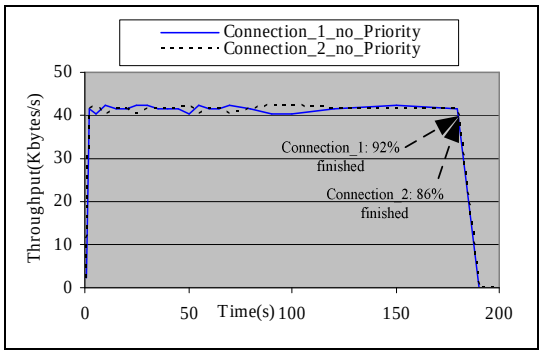

Fig.6: Without Context-driven connections management

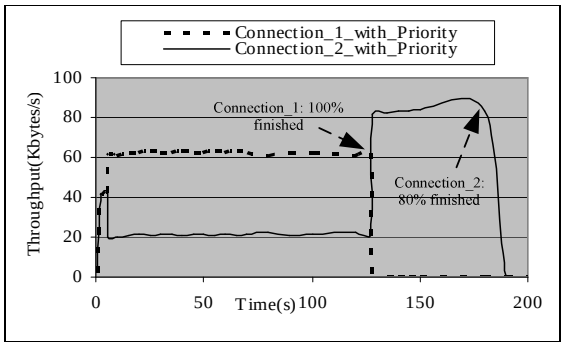

Fig.7: Context-driven connections management by profile (solution one)

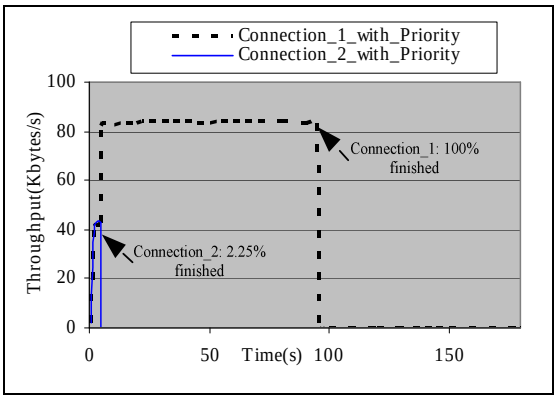

Fig.8: Context-driven connections management by profile (solution Two)

5. CONCLUSIONS AND FUTURE WORK

Existing layered architectures are inefficient when deployed in wireless networks. Hence, cross-layer architecture is essential for next generation communications. Therefore, we have proposed a new cross-layer architecture for autonomic communications in [1]. Alongside an overview of the proposed architecture we have presented two different prototype implementations of it with some initial results in this paper. For the first implementation we considered the possible integration of node-wise context-awareness to control mobile nodes transmission power using a simple algorithm. The result proves the possibility of context-based TPC using cross-layering. We implemented context-driven traffic management by profile as a second prototype of our architecture. As user layer's profile indirectly presents user's context. We could conclude that results of two different implementations show that context-awareness is possible through our architecture. This ultimately demonstrates the feasibility and potential of our proposed architecture.

Although with the cross-layering wireless performance improvements and autonomic communications are possible, unbridled cross-layering could raise loops between the layers and could deliver opposite results. Therefore we should take note of it during the design as well in the implementation stages. Network-wide view (global) is the contribution of the different node's local views. A sound node-wide (local) view formation (as we did here) is the foundation to the network-wide view formation. In future work we will try to integrate network-wide (global) view to deal with more complex situations.

ACKNOWLEDGEMENT

This work is partially supported by Science Foundation Ireland (SFI) under grant number 04/RPI/1544, "Secure and Predictable Pervasive Computing."

REFERENCES

[1] M.A. Razzaque, Simon Dobson and Paddy Nixon, "A cross-layer architecture for autonomic communications", In Autonomic Networking, Dominique Gaïti, Guy Pujolle, Ehab Al-Shaer, Ken Calvert, Simon Dobson, Guy Leduc and Olli Martikainen (ed). Volume 4195 of LNCS. Springer-Verlag. Paris, FR. 2006, pp. 25–35.

[2] Srivastava, V. and Motani, M, "Cross-Layer Design: A Survey and The Road Ahead", IEEE Communications Magazine, Volume 43, Issue 12, Dec. 2005, pp. 2 – 119.

[3] Simon Dobson, Spyros Denazis, Antonio Fernández, Dominique Gaïti, Erol Gelenbe, Fabio Massacci, Paddy Nixon, Fabrice Saffre, Nikita Schmidt and Franco Zambonelli. A survey of autonomic communications. ACM Transactions on Autonomous and Adaptive Systems 1(2). December 2006. To appear.

[4] Prehofer, C.; Bettstetter, C.**, "**Self-organization in communication networks: principles and design paradigms", IEEE Communications Magazine, July 2005 Page(s): 78 – 85.

[5] Daniel G.Sachs et al, "GRACE: A Hierarchical Adaptation Framework for Saving Energy", ACEED 2005.

[6] Dzmitry Kliazovich and Fabrizio Granelli, "A Cross-layer Scheme for TCP Performance Improvement in Wireless LANs", Globecom 2004, IEEE Communications Society, pp. 841-844.

[7] Marco Conti, Gaia Maselli, Giovanni Turi, Sylvia Giordano "Cross layering in mobile Ad Hoc Network Design", IEEE Computer Society, February 2004, pp. 48-51.

[8] V. T. Raisinghani and Sridhar Iyer, "ECLAIR: An Efficient Cross Layer Architecture for Wireless Protocol Stacks", WWC2004.

[9] Winter, R.; Schiller, S.; Nikaein, N.; Bonnet C., "CrossTalk: A Data Dissemination-based Cross-layer Architecture for Mobile Ad-hoc Networks", IEEE Workshop on Applications and Services in Wireless Networks, Paris, June 2005.

[10] X. Gu, X. Fu.H. Tshofenig, and L. Wolf, "Towards Self-optimizing Protocol Stack for Autonomic Communication: Initial Experience", Proceedings of the 2nd IFIP International Workshop on Autonomic Communication (WAC'05), Springer Lecture Notes in Computer Science Vol. 3854 (LNCS), October, 2005, pp: 186-201.

[11] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar, "Power control in ad-hoc networks: theory, architecture, algorithm and implementation of the COMPOW protocol," in Proc. Euro. Wireless, 2002, pp. 156–162.

[12] Daji Qiao, Sunghyun Choi, Amit Jain, and Kang G. Shin, "Adaptive Transmit Power Control in IEEE 802.11a Wireless LANs", IEEE Vehicular Technology Conference 2003.

[13] Chansu Yu, Kang G. Shin, and Ben Lee, "Power-Stepped Protocol: Enhancing Spatial Utilization in a Clustered Mobile Ad Hoc Network", IEEE journal on selected areas in communications, vol. 22, no. 7, September 2004, Page(s): 1322 – 1334.

[14] NS-2: http://www.isi.edu/nsnam/ns/ns-build.html

[15] M.A. Razzaque, Simon Dobson and Paddy Nixon. Classification and modeling of the quality of contextual information. Journal of Autonomic and Trusted Computing. 2006. To appear.