Context Awareness through Cross-Layer Network Architecture

Murthy D.H.R
Assistant Professor, Dept. of Information Technology
Shree Rayeshwar Institute of Engineering and Information Technology
Shiroda, Goa, India

Ramesh M. Badiger
Assistant Professor, Dept. of Computer Science and Engineering
Tontadarya College of Engineering,
Gadag, Karnataka, India

Abstract
Layered architectures are not sufficiently flexible to cope with the dynamics of wireless-dominated next generation communications. Cross-layer approaches may provide a better solution: allowing interactions between two or more nonadjacent layers in the protocol stack. Cross-layer architectures based on purely local information will not be able to support system-wide cross-layer performance optimization, context awareness, etc. A new cross-layer architecture which provides a hybrid local and global view, using gossiping to maintain consistency. This paper includes the possibilities of context-awareness in communications through this architecture by two examples. The first example uses user centric context to control the available link-bandwidth and satisfy user accordingly. The second uses contextual information to control the transmission power of a mobile node.

Index Terms
Layered architecture; Cross-Layer Architecture; Context awareness.

1. Introduction

Wireless network like Ad hoc Networks are characterized by a dynamic topology, a limited bandwidth, and energy consumption constraints. Wireless link quality changes dynamically through time and space. Existing layered design is insufficiently flexible to cope with these dynamics of wireless communications. Some innovating techniques have to be developed to improve the performance of those networks.

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 nonadjacent layers of the protocol stack, and end-to-end performance is optimized by adapting each layer against this information. Use of contextual information in computing to improve performance, adaptability, user satisfactions, etc is well known, on the other hand in communications it is not that much explored.

Some of the early initiatives in this area are autonomic communications, Ambient Networking, CATNIP etc. Context information can be classified based on factors such as (volatile, on-volatile), (real time, non-real time), (private, public), (network-centric, user-centric), etc.

Network-centric context information is the one about changes in the network; at the same time it can be used to cause desired changes, it can be used to make networks more receptive to users' needs and enhance the users experience by making the communication easier and richer. Exploiting network-related context information can make wireless networks like ad hoc simpler, more efficient and more powerful thus simplifying the management of the networking infrastructure for network operators while providing end users with value-added services and an enhanced communication experience.

Cross-layer architecture could be a used as an infrastructure to support network level context-awareness. Most existing cross-layer architectures (including GRACE, WIDENS, MobileMan and ECLAIR) rely on local information and views, without considering the networking context or views which may be very useful for wireless networks. Only Crosstalk is based on local as well as network-wide global view (partially). 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 know this new cross-layer architecture that is based on building and maintaining a view of the network's state (context) and constraints, utilizing gossiping for gathering information from neighboring nodes.

Manuscript received December 5, 2013
Manuscript revised December 20, 2013

Objective of this paper is to show that cross-layering approach with appropriate architecture can be used for Context-awareness in communications. To do that we considered two different implementations of the proposed cross-layer architecture.
First implementation we consider the possible integration of node-wise user level context-awareness for bandwidth management to support QoS.
In second implementation we consider context awareness to control mobile nodes transmission power using a simple algorithm. The results show the possibility of context-awareness through cross-layering. An overview of the proposed cross-layer architecture is presented in section 2. Section 3 presents context-awareness through cross-layering. Implementations and associated results are presented in section 4 and section 5 concludes with some directions for future work.

2. Proposed Cross-Layer Architecture

If each protocol of the layered model is designed independently, the end of the execution of a low level protocol, consuming data at the destination node, should not influence the behavior of high level protocols. In contrast, the cross–layer concept adapts the protocols to the wireless context by sharing information between layers and by an overall optimization instead of multiple optimizations at different levels. Cross-layer design has been proven to be effective in wired and fixed network.

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 implemented. The key distinction between this architecture and most other crosslayer 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.

1.1. 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. 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):

1.2. Existing TCP/IP layers

These provide normal layering support when it is necessary. It also provides the different layers' context 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.

1.3. 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.

1.4. 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.

1.5. 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.

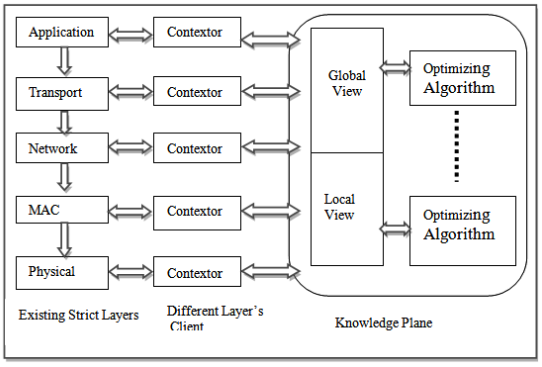

Fig.1: Proposed Cross-Layer Architecture

3. Context-Awareness Through Cross-Layering

Historically, context has been adapted from linguistics, referring to the meaning that must be inferred from the adjacent text. In respect to computing world definitions of context varies with computing environment (available processors, devices accessible for user input and display, network capacity, connectivity, and costs of computing) user environment (location, collection of nearby people, and social situation) and physical environment (lighting, noise level etc). Context is "any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context can also be defined as "Context is not simply the state of a predefined environment with a fixed set of interaction resources. It is part of a process of interacting with an ever-changing environment composed of reconfigurable, migratory, distributed, and multi scale resources."

A definition of context-awareness is given as: a system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task. So context-aware communication is a system which uses user's/device's/ protocol layer's contextual information to provide relevant information and/or services to the user. Even it can use one layer context information to provide services to other layer. For example, Transport layer (TCP) can utilize Network Layer's node mobility information for the congestion control in wireless network. Again context-sensitive communication is a type of communication, where the communication channel is established between devices based on some specific contexts. So importance of contextual information in next generation communications specially in wireless ad-hoc network more than necessary.

As context can be seen everything around and/or within an entity, it is impractical to make a concrete list of all the possible contextual information. But classification of context and context-awareness can help application designer and developer to uncover the possible context and simplify the context manipulation. Context information can be categorized into several classes based on factors such as (volatile, nonvolatile), (real time, non-real time), (private, public), (network-centric, user-centric), etc. Network element related context information such as queue length, processing power, etc varies dynamically to the change of network traffic and other network dynamics. Context information in networking also could be viewed as node-wide local and network-wide non-local information. Node-wide local information mainly includes the user's preferences/requirements, protocol layer's information, node's capacity, etc. Table 1 shows the possible contextual information (node-wide) related to different protocol-layers. Non-local information includes neighbor node's processing capacity, processor utilization, movement pattern, remaining power, etc. Existing strict layering approach allows interaction only between adjacent layers and this restricts the possibility of context-awareness within different layers and user. Cross-layering approach support interactions between one or more non-adjacent layers which opens the door for- context-awareness throughout the protocol layer stack. For example; with the cross-layering application layer can interact with network or MAC layer and set some parameters there according to user's requirements and/or user contexts but in strict layer it is not possible.

Table 1: Possible different layer's contextual information

Layer	Information (Context)
Physical	Node's location, Movement information, Transaction range, Battery power, SNR
MAC/Link	Link bandwidth, Link quality, Packet delay
Network	Neighbors' list, Routing affinity, Routing lifetime, Multiple Routing
Transport	Congestion/Receiver window, Timeout clock, Packet loss rate
Application	User preferences, QoS, Topology Control, Algorithm, Network Map

In strict layering, other than the Application Layer no layer does 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. Users' preferences or priorities can be defined through a profile, based on different situations or contexts. But with the existing strict layering it is not possible to deliver this context-based priority information to relative layers; cross-layering could be helpful. Consider a user in a location at the edge of WLAN's coverage with his laptop and downloading two files one is important FTP file and the other one is a less important music file. 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.

4. Implementation And Results

In this section we apply the above architecture to two example scenarios to show the possibility of context awareness in networking through cross-layering. In both implementations simple algorithms are used as the goal of this work is to justify the architecture for context-awareness not to deal specific algorithms. 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. User-centric Context-awareness to support QoS

Although "User" is not a layer in existing strict layering system, every layer is working directly or indirectly to provide services to user. Therefore user level contextual information is very important in providing services with QoS. User-centric context-awareness deals with user level different contextual information like user position, situation, location, requirement, etc. To support this context-awareness user level contextual information has to be delivered to relevant layer which requires cross-layering approach. Consider a user is using his laptop in the coverage area of a WLAN. He is running a video streaming application and also downloading a file. Video streaming is real time traffic and the other one is non-real time traffic. For real time traffic QoS is a key issue. For example for video streaming a minimum bandwidth has to be maintained all the time. For an important task during the application running, user has moved to place where link quality has gone down as well as the bandwidth. In this context if the user continue running both the applications then bandwidth of video streaming goes down less than the minimum requirement for QoS. Moreover he has to continue the video streaming considering the importance. One can solve this problem by closing the other connection manually but using cross-layering with the help of context-awareness it can be done automatically.

Application's bandwidth share, on a *receiving* device can be done through manipulation of the *receiver window* of its TCP connection i.e. *Receiver Window Control* [18].The receiver reflects its receive buffer status by the *advertised window* field in the acknowledgments to the sender. If the advertised window decreases, the sender also reduces its send rate. This TCP behavior can be exploited to reduce the throughput of some applications and consequently increase throughput of rest of the applications, on the receiver. For the above problem we can use this *Receiver Window Control.* When the link quality goes down then advertizing the very small or even zero receiver windows for the file download connection will help to increase the bandwidth for the streaming.

Figure 2 shows a snapshot of the implementation of the cross-layer architecture to solve this problem. Stepwise interactions for this implementation are: (1) KPlane periodically checks the link quality from MAC Layer (2) if the link quality is lower than a critical value then it gets the priority levels from the user context 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 3 shows the problem when there is no context-awareness and both the applications get the almost same bandwidth. Most importantly video streaming the important task doesn't get the minimum bandwidth to support QoS. Figure 4 is the shows the solution to this problem.

Figure shows when the link quality as well the link bandwidth goes down instead of sharing the current bandwidth between the applications equally it utilizes user context (priority) to distribute bandwidth. As video streaming is real time traffic and it can't tolerate delay but the other one can. Therefore real time traffic gets the higher priority than non-real time.

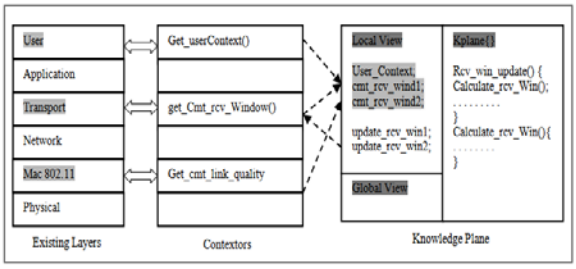

Fig.2: A snapshot of the implementation of User-centric Context-awareness to support QoS

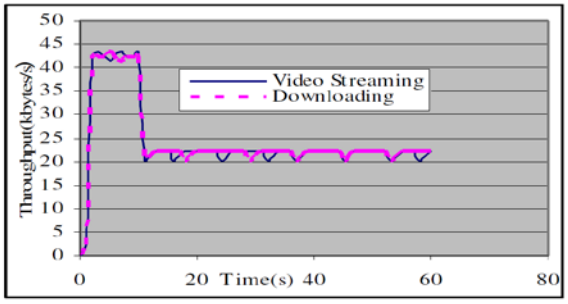

Fig. 3: Without user-centric Context-awareness

For this 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.

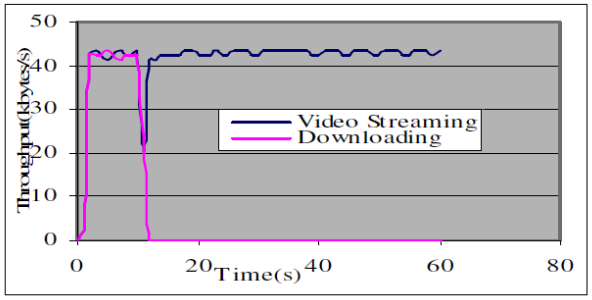

Fig 4: User-centric Context-awareness to support QoS

B. Context-aware Transmission Power Control

This work is considering almost the same assumptions used in [19] 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 5 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.

The results of two scenarios and no TPC are shown in figure 6 & 7, 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 6 (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 6: 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 7 shows them corresponding power used in the figure 6 to support different distances. It is clearly shows the benefit of TPC. These results justify two points: (i) with the help of more useful contextual 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 [6]. But interdependencies between the layers could be a problem.

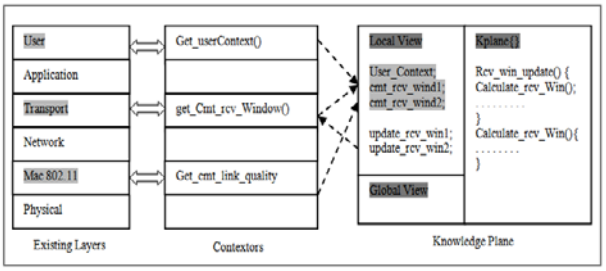

Fig.5: A snapshot of the implementation of Context-aware Transmission

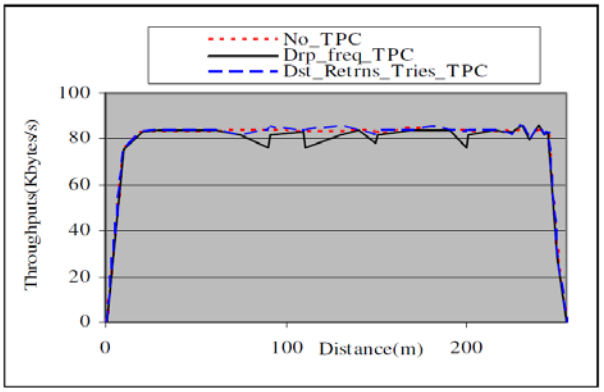

Fig. 6: Comparison between no-TPC, TPC with dropping frequencies only and TPC with distance & retransmission tries

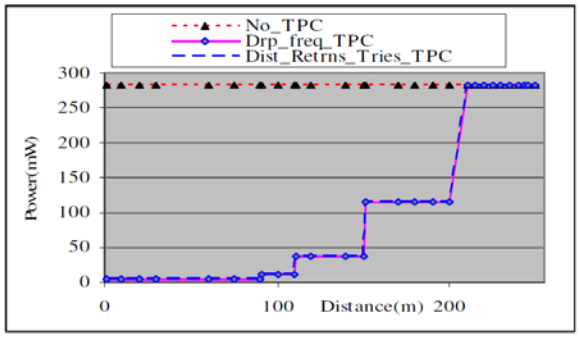

Fig.7: Powers used in figure 6 to cover different coverage distances

5. Conclusions And Future Work

Existing layered architectures are inefficient when deployed in wireless networks. Hence, protocol layer architecture with cross-layering facilities is essential for next generation communications. Alongside an overview of the proposed architecture we showed through two simple implementations that context awareness through cross-layer is possible. For the first implementation we considered the possible integration of user-centric context-awareness for bandwidth management to support QoS. The result proves the possibility of user-centric Context-awareness using cross-layering. Context-awareness to control mobile nodes transmission power using a simple algorithm considered in second implementation. The result proves the possibility of context-based TPC using cross layering.

We could conclude from results of two different implementations that context-awareness is possible through our architecture. This ultimately proves the feasibility of context-awareness through cross-layer approach.

Although with the cross-layering context-awareness, wireless performance improvements 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 contextual view is the contribution of the different node's local views. A sound node-wide local contextual 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 contextual view to deal with more complex situations.

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.

[4] Raffaele Giaffreda et al, "Context-aware Communication in Ambient Networks", Wireless World Research Forum.

[5] Williamson and Q. Wu, A Case for Context-Aware TCP/IP, Performance Evaluation Review, 29(4),pp.11-23, March 2002.

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

[7] A.K. Dey, G.D. Abowd. "Towards a Better Understanding of Context and Context-Awareness," CHI2000 Workshop, 2000

[8] J. Coutaz, J. Crowley, S. Dobson, and D. Garlan. "Context is key." Communications of the ACM 48(3), March 2005

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

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

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

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

[13] 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.

[14] D. Cark and D. Tennenhouse, "Architectural Considerations for a New Generation of Protocols", in Proc. of ACM SIGCOMM'90, September 1990.

[15] Brown, P.J., Bovey, J.D., Chen, X., "Context-aware applications: from the laboratory to the Marketplace", IEEE Personal Communications, Volume: 4, Issue: 5, pp. 58 – 64, October 1997.

[16] S. S. Yau and F. Karim, "An Adaptive Middleware for Context-Sensitive Communications for Real-Time Applications in Ubiquitous Computing Environments", Real-Time Systems, The International Journal of Time-Critical Computing Systems, Kluwer Academic Publishers Dordrecht, The Netherlands, vol. 26, no. 1, pp. 29-61, Jan 2004.

[17] http://www.isi.edu/nsnam/ns/ns-build.html

[18] P. Mehra, A. Zakhor, and C. Vleeschouwer. Receiver-Driven Bandwidth Sharing for TCP. In IEEE INFOCOM, SF, USA, April 2003.

[19] S. Narayanaswamy et al "Power control in ad-hoc networks: theory, architecture, algorithm and implementation of the COMPOW protocol," in Proc. Euro. Wireless, 2002, pp. 156–162.