A Cross-Layer based handover for TCP applications

D. Fanni, M. Luglio, C. Roseti, and F. Zampognaro

University of Rome "Tor Vergata"
Via del Politecnico, 1
00133 Roma – Italy

***Abstract* — In a mobile communication scenario handover functions must be supported in order to keep connection alive switching between different points of access. Moreover, if the infrastructure is composed of different networks or technologies, intersegment handover is necessary to roam from a component to another. The handover function is implemented either for necessity or to improve performance. Necessity means that the terminal must perform a handover because its current network connection is worsening; in the other case switching to another network can help to achieve better performance even if the current network connection is fine. Furthermore, the handover can be soft or hard, according to the possible overlap in the connection of the two segments involved. This paper illustrates the behavior of TCP during intersegment handovers in an architecture including two segments: satellite and WLAN. In particular, the goal is to carry out a thorough analysis of a cross-layer based mechanism applicable during handovers to optimize the TCP performance. Such a mechanism is validated with an ad-hoc software simulator written in C++ and the most meaningful results are shown.**

***Index Terms* — cross-layer, TCP, handover, multi-segment networks.**

I. Introduction

Nowadays fast access to telecommunication networks is often required in mobility and along wide and heterogeneous geographical areas. To meet these requirements, broadband satellite networks, which allow a potentially global coverage and then an ubiquitous access to the communication services, and WLAN systems, which offer a much higher capacity and less latency but in a limited coverage area, if jointly used, can represent an optimal solution for mobile and nomadic users to uninterruptedly access the Internet. In fact, low delay and wide band can be provided under the WLAN "Hot Spot" coverage, while continuity of service can be guaranteed thanks to satellite links which can either provide capacity directly to users or to WLANs as backbone. Of course, efficient inter-segment handovers (HO) procedures are needed to switch between such different technologies [1].

Mobile Internet Protocol and its newer realization Hierarchical Mobile IPv6 [2][3][4] has been identified as a valid approach to provide a mobility support at the IP layer, allowing mobile nodes to change network without changing their IP address to keep communication consistency.

Unfortunately, TCP/IP protocols were not designed to optimally handle mobility. HO between links with different delay and bandwidth-delay product causes a number of problems such as the generation of segment bursts and delivery of out-of-order packets [5]. Even a good management of the HO at IP level does not prevent completely TCP from experiencing unwanted effects. In fact, both transmission errors and the sudden delay change cause harmful consequences to TCP performance:

- Losses are misinterpreted as a congestion and thus TCP congestion window is dropped to a lower value [6];
- TCP computes the retransmission timeout (RTO) on the basis of measured RTT values. Then, an unexpected increase of the RTT could lead to an undesired RTO expiration with a consequent reduction of the congestion window to a minimum value [7].
- After the HO procedure is completed, the new link is established but TCP packet flow resumes with inefficient or un-initialized parameters set.

Then, when moving from a segment to another, even assuming that negotiation procedures to access the new channel are efficient and transparent to upper layers, nevertheless a certain time t^* is needed to physically switch the mobile device and notify the new channel resource manager. In this time period, no data are received (hypothesis of Hard HO [8]). Moreover when the HO is performed for necessity (i.e. WLAN link power decrease at the edge of the coverage area) BER increases as well as packet losses perceived by the unaware TCP.

In this frame, TCP could benefit of explicit notifications of handovers in order to better cope with them. To predict a handover event and to allow TCP to perform a certain number optimization tasks, we propose a cross-layer mechanism between transport layer (adopting TCP) and network layer (adopting Mobile IP protocol).

The paper is structured as follows: section II illustrates the reference scenario summarizing all the characteristics and assumptions adopted in our analysis; section III gives the details on the impact of the handover process on TCP dynamics; section IV shows a reference Cross Layer Architecture used in section V as basis for a novel TCP scheme; section VI describes the C++ simulator used to evaluate performance improvement coming from the adoption of the proposed cross-layer scheme and presents the most relevant results; section VII provides conclusions.

II. Reference Scenario

The reference scenario is compliant with HMIPv6 (Hierarchical Mobile IPv6) standard [4] with the key elements shown in the block diagram in fig. 1.

The mobile node, which represents the user terminal, is moving across a hybrid network composed of one Access Router (i.e. AR1) that represents the satellite access and at least one AR (i.e. AR2…ARn) which uses WLAN technology. We suppose that the source of the data received by the Mobile

Node (MN) is a remote host connected via Internet to the mobile environment ("Sender").

All the downlink traffic coming from the Sender through different radio networks is routed via a single entity, the Mobile Anchor Point (MAP), which redirects traffic to the proper radio network (satellite or WiFi AR) and manages handover signaling. Then, it is logically connected to both satellite gateway and WiFi access points management, but also to other entities in charge to perform the mobility functions (Home Agent and Corresponding Node) that will not be addressed herein.

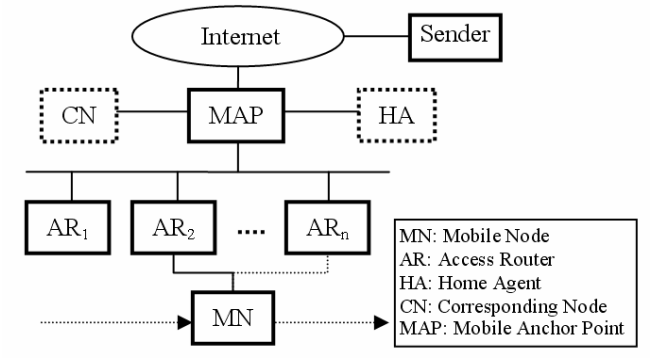

Figure 1. Mobile environment

The reference scenario used throughout this paper is composed of different overlapping segments of an access network. The technologies considered for the connections with the end user will be either:

a) two way satellite standard (DVB-RCS) over a GEO link;

b) WLAN access via WiFi "Hot Spots" (IEEE802.11b).

In general, the moving user terminal is supposed to be able to perform transparently the hard HO operations between the access technologies a) and b) to obtain a seamless data channel during its movement.

Only three different HO occurrences are identified:

1) DVB-RCS → WiFi;

2) WiFi → DVB-RCS;

3) WiFi → WiFi.

The case satellite-satellite is clearly of no practical interest, since the satellite coverage is assumed large enough with respect to user mobility needs. Case 3) is not addressed in details since it is out of the scope of this paper.

Fig. 2 pictorially shows the reference scenario, with every possible handover occurrence between segment a) and segment b) highlighted by vertical lines, with the user terminal (MN) under satellite coverage and performing HO whenever a WLAN connection becomes available. The time spent into the WLAN coverage (T_{WLAN}), not corresponding to the interval served by the WLAN, depends on the speed of the MN.

In this paper we address the problems experienced by TCP during HO. We assume that the HO procedure is fully handled by HMIP up to the network layer and that during the HO procedure a certain number of IP packets are lost. Moreover, an additional no-connection time (out of service or HO time) occurs. This happens because of limited buffers in the terminals and for the transitory in the setup of the Intermediate Frequency electronics and Phase Locked Loop that produces corrupted packets [9]. The out of service time is mainly due to the limits of hardware dedicated to the hard HO switch and to the HO decision delay [8].

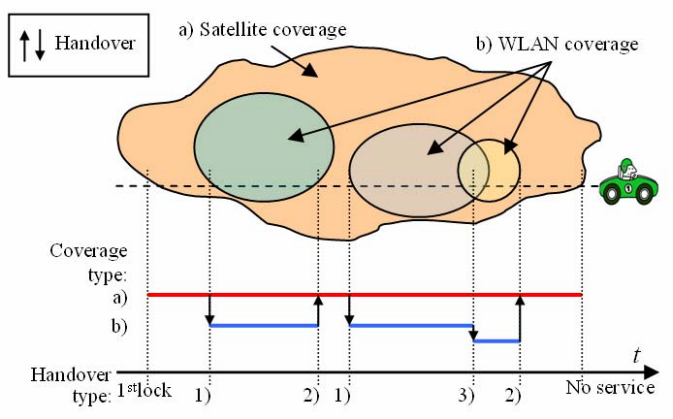

Figure 2. Reference scenario

III. TCP BEHAVIOR DURING HANDOVERS

TCP congestion control mechanism [6] interprets all the losses as implicit congestion. Then, when "holes" are detected in the sequence of the received packets/ acknowledgments, TCP generally retransmits the missing packets and decreases its transmission rate. Furthermore, TCP probes the available bandwidth and regulates its transmission rate by using the ACK reception rate as an internal clock. Unfortunately, during handover events, the corruption of packets (due to the power degradation) or quick round-trip delay (RTT) variation can cause harmful dynamics for TCP that perceives false congestion states and a not regular ACK flow. Main dynamics affecting TCP behavior in the considered scenario are briefly described.

A. Transmission Errors

Handover from WLAN Hot Spot to Satellite system is preceded by a gradual degradation of the WLAN link power that increases the Packet Error Rate (PER) perceived by TCP. TCP misinterprets such losses as a congestion notification, reduces its congestion window and triggers the congestion avoidance algorithm. Then, TCP flow will resume on the satellite link with very small congestion window and increases it slowly due to the long latency. As a consequence, satellite capacity will be wasted for long time.

B. Handover time (t^*) implications on RTO

The handover time t^* is the time needed to the mobile host to switch between different technologies. Therefore, when a handover is performed, an unexpected deep variation of the RTT could lead TCP to unnecessarily trigger its ACK-clocked recovery mechanisms [7]. In particular, Retransmission Time-Out (RTO) is automatically updated over the time on the basis of the RTT measurements.

A sudden increase of the RTT due to the handover time may lead to RTO expiration. In fact, when moving from WLAN to satellite system, the large delay difference ($Delay_{satellite} >> Delay_{WLAN}$) increases the ACK-clock break and

then the probability of the RTO expiration.

On the contrary, when the handover is exploited in the opposite direction, the RTO results to be largely overestimated causing too long wait time to restart transmission in case of network congestion.

C. Generation of segment bursts

If the new link delay is smaller than the old link delay (it is the case of handovers from satellite system to WLAN), the first ACK sent over the new link could overtake the last *k* ACKs sent over the old one. Then, due to TCP cumulative ACK scheme [7], a burst of *k* packets is sent at once. Intermediate buffers could not be dimensioned to support such a burst length causing multiple losses.

D. Bandwidth-Delay product change

TCP congestion window represents a sender's estimate of the bandwidth-delay product (BDP) or pipe size. Different effects follow a BDP change depending on whether BDP increases or decrease after a handover:

- moving from WLAN to the satellite system ($BDP_{new}>BDP_{old}$), satellite link capacity will be underutilized; in fact, the initial congestion window is small and its increment is slow if congestion avoidance is performed (increase of 1 segment per RTT);
- moving from satellite system to WLAN ($BDP_{old}>BDP_{new}$), TCP will inject on the new link more packets than it would actually fit, causing congestion and dropping packets in the bottleneck router; if all the routers through the new path have enough room to accommodate all the packets the perceived RTT will drastically increase.

IV. Cross-Layer Signaling Architecture

Among several Cross Layer (CL) signaling architectures referenced in literature, ECLAIR [10] has been considered as baseline for our work because it significantly suites the selected scenarios. In fact, it provides:

- Bi-directional CL communication channel between each pair of layers in the stack;
- Centralized control mechanism, called *Optimizing Subsystem (OSS)*, to collect more CL adaptations, handle a multiple CL interaction, and to avoid conflicts between one or more CL interactions [11];
- Several Cross-layer enhancements via an open interface called *Tuning Layer* (TL) between each layer and the OSS.

Fig. 3 shows the general architecture of ECLAIR adopted. Using this architecture TLs are a sort of middleware to mask to the Optimization Subsystem the platform dependent TCP/IP stack.

The ECLAIR architecture has been already proposed in [10] to better handle HO addressing only theoretical analysis (no simulation results). Moreover, they used a different approach: for instance on HO recovery TCP window is restored and not set back to 1 segment as it is proposed in this paper.

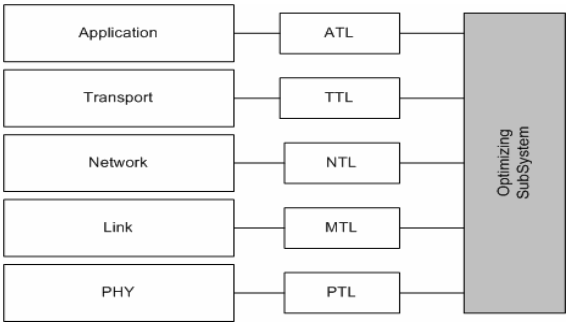

Figure 3. Cross-Layer Architecture Adopted (ECLAIR)

V. Cross-Layer Interactions Design

This section describes how the ECLAIR architecture introduced in section IV has been adopted to avoid the harmful dynamics that arise during handovers (section III). We have introduced three CL adaptations to the standard TCP implementation:

A. Freezing TCP flow

To avoid the burst overflow (III.C) and an unnecessary overgrowth of the internal queues, it is useful to freeze TCP data flow, while MN is changing AR. Although this enhancement is the most intuitive, it implies a few problems, which are addressed and solved with the next two enhancements.

B. Resetting of the sampled RTT

TCP standard implementations attempt to predict future round-trip times by sampling and averaging the RTT of each packet according to the Karn and Partridge relation (1) [12]:

$$sRTT_{i+1} = \alpha * sRTT_i + (1-\alpha) * S_i, \quad (1)$$

where sRTT is the smoothed RTT, α is a constant between 0 and 1, and S_i is the sampled RTT of the Packet i^{th}. The goal is to compute the RTO, usually equal to *2*sRTT*. To avoid the harmful behavior described in section III.B, it is useful to reset the *sRTT* after each change of interface, avoiding a misleading estimation of sRTT with the above mentioned consequences.

C. Optimized handling of cwnd and sstresh

TCP standard implementations handle the congestion control through a combined use of *cwnd* and *ssthresh* [5]. After MN has switched to a different AR, as explained in section III.A, the increase of Packet Error Rate may lead to packet losses, with the side effect that TCP reduces *cwnd*, and *sstresh*. To avoid this behavior, it is useful to handle these parameters in a smarter way. Thus, since *ssthresh* represents the threshold above which TCP performs a linear increase of the *cwnd* instead of exponential, it should be useful to set *ssthresh* close to the real capacity of the new channel that is related to the BDP (section III.D).

Moreover, after a handover from satellite to WLAN, *cwnd* may be greater than BDP. This causes harmful dynamics of TCP with possibility of channel congestion. However, it is possible that, due to some external circumstances (e.g. high traffic rate), leaving the *cwnd* unchanged might lead to a further congestion. A safe way to prevent this is to set, after a

handover, the *cwnd* value to the Maximum Segment Size (MSS).

VI. SIMULATIONS

A. Description

In order to evaluate performance of the proposed enhancements in comparison with the standard protocol stack, a custom simulator in C++ has been developed. It re-implements the communication stack keeping the data flow in a stand-alone context. The major reason to use a custom simulator is that any other existing simulator does not provide explicit methods to enable CL communication channels.

Through the Object Oriented modeling (OOD) [12] shown in fig. 4 and adopted for this simulator, it is possible to map classes to elements of ECLAIR, so that the stack layering concept can be deployed in OOD using classes hierarchy. This will help in the future to apply the proposed changes to a real implementation of the stack.

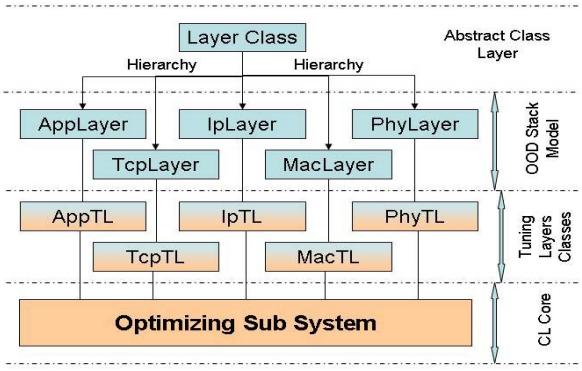

Figure 4. Simulator Object Oriented Simplified Model

The CL simulator core has been setup to simulate a data transfer according to the scenario described in section II, assuming a SAT channel of 1 Mbit/s with RTT fixed to 550 ms and a "Hot Spot" area as a WLAN with a bandwidth of 11 Mbit/s and a RTT of 25 ms. A PER of 10^{-4} is assumed in both SAT and WLAN channel. Data packet dimension is fixed to 1024 bytes. We measured the Data Transfer Time (DTT) and the instantaneous throughput of a heavy data transfer application, such as a FTP, with data file of 10 MBytes. During the transfer, at T_0 it is assumed that the MN, currently served by a satellite system, enters in a Hot Spot area, and thus a handover to the WLAN network occurs. After that, it leaves the Hot Spot and comes back to the SAT network. This simulation has been repeated varying two parameters: T_{WLAN}, that is the overall time during which the MN is inside the WLAN coverage, and *t**, that is the handover execution time, assumed constant independently on the start and the target network to simplify the model.

B. Performance evaluation

Fig. 5 shows instantaneous throughput variation as a function of time when a handover occurs during a data transfer with t*=2.5 s, T_0= 5.5 s and T_{WLAN},= 5 s.

When at T_0 the MN performs the HO to the "Hot Spot" using the modified stack, the throughput increases thanks to the effect of the proposed technique, which in particular avoids an overflow of the internal queues. A reset of the sRTT is also forced, causing a faster adaptation of TCP at T_0+t* to new channel conditions.

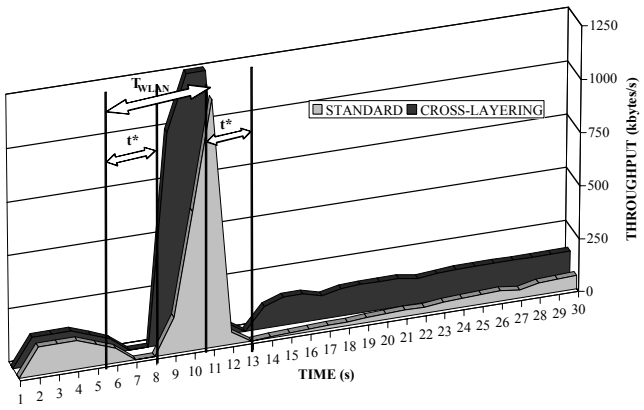

Figure 5. Throughput with t*=2.5 s and T_{WLAN}= 5 s

At T_0+T_{WLAN}, when the MN moves back to the SAT channel, the standard stack shows a worse response due to the high RTT that causes a slower growth of *cwnd*. With the proposed cross-layer handling of *cwnd* and *ssthresh* values it is possible to force TCP to re-apply the Slow Start algorithm and obtain a faster growth of data rate up to the BDP of the new link. This, combined with a reset of sRTT, prevents unnecessary timeouts and unwanted *ssthresh* decreases. Finally, by freezing TCP data flow, it is also possible to avoid the initial delay caused by the data stored in the internal queues during the handover that must be sent when the new channel becomes available.

Fig. 6 shows the variation of the DTT by varying t* in the interval [0.5 s, 6.5 s] with T_{WLAN} fixed to 7 s, using the standard and the modified stack. The horizontal line represents the transfer time if the MN keeps staying connected through the SAT channel without performing HO.

The value of t*, defined as t^*_{MAX}, at the intersection between the DTT curves and the SAT constant transfer line represents the trade-off HO execution time. If the HO procedure takes less than t^*_{MAX}, in case a WLAN is available, it is advisable to perform a HO to reduce DTT. If the HO procedure takes more than t^*_{MAX} there are no advantages in terms of DTT with respect to the SAT constant transfer rate and thus it is better that the MN does not perform the HO, unless it is strictly necessary.

In fig. 6, it is also evident a meaningful improvement in terms of DTT implementing the modified stack with respect to the standard (about 30s). Also t^*_{MAX} value results improved: while t^*_{MAX} is about 2.5 s for the standard stack, it is 5.5 s for the modified stack. As a consequence, less severe requirements in terms of HO execution time are allowed.

Fig. 7 shows how DTT varies as a function of T_{WLAN}, considering a t* of 2.5 s. Values for T_{WLAN} ranges between 2 s and 15 s.

In the whole range of T_{WLAN}, fig. 7 shows that the proposed mechanisms always help to greatly alleviate the TCP problems

which increase the DTT when HO is performed.

When T_{WLAN} is small (less than 3 s), the improvement introduced by the modified stack is relatively small because $T_{WLAN} \approx t^*$ and there is no effective usage of the Hot Spot coverage. Increasing the T_{WLAN} beyond 3 s, the DTT of the modified stack rapidly decreases, while DTT of standard stack keeps constant, due to the slow growth of throughput when the session switches back to the SAT channel at (T_0+T_{WLAN}). Then, the gap between the curves remains almost the same (about 30 s) until T_{WLAN} is big enough to complete the 10 Mbytes file transfer before the communication comes back to the SAT channel. In these circumstances, the gain is limited since there is no transfer when the MN switches back to the SAT channel.

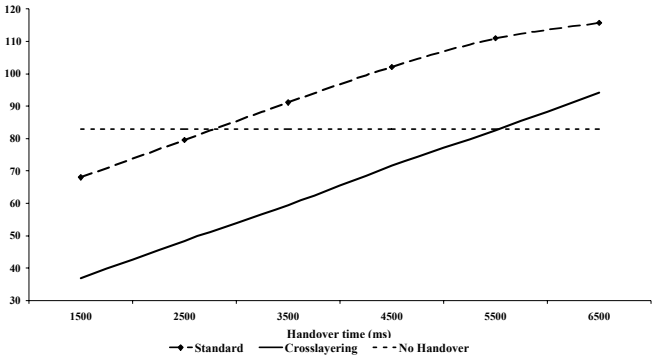

Figure 6. DTT @ T_{WLAN} constant (7 s)

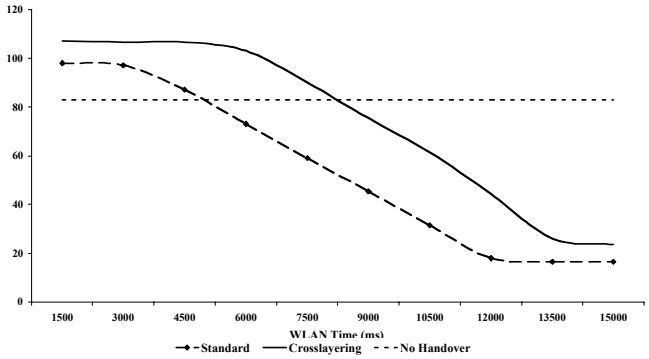

Figure 7. DTT @ t* constant (2.5 s)

T_{WLAN_MIN} is the minimum T_{WLAN} value that leads to a benefit, in terms of DTT decrease, while performing a HO: it can be identified as the value of T_{WLAN} at the intersection of DTT curves with the SAT constant transfer rate in Figure 7. In practice this means that if the MN is under the Hot Spot coverage less than T_{WLAN_MIN} there is no improvement in terms of DTT and then it is better not performing the HO, unless it is strictly necessary. It is possible to note that T_{WLAN_MIN} is lower for the modified stack implying that we can accept shorter permanence time in Hot Spot coverage areas.

VII. Conclusion

In a scenario characterized by heterogeneous networks composed of a satellite segment and a terrestrial wireless segment (WLAN) the behavior of TCP during handovers between links with different Bandwidth Delay Product has been addressed, taking into account the real hardware implementation that can lead to packet loss and delays during the HO procedure itself. We proposed some cross-layer enhancements targeted to the mitigation of such problems, introducing some adaptations in parameters and algorithms of TCP during the HO phase.

This preliminary work has been the starting point for defining a complete cross-layer model, implemented into a TCP/IP emulated stack written from scratch in C++.

We proved the feasibility of a cross-layer approach which truly offers better performances compared to a standard stack. In particular, outcomes regard both the handover parameters (T_{WLAN_MIN}, t^*_{MAX}) and the Data Transfer Time, which result improved when adopting the proposed modified stack.

References

[1] M. Leo, M. Luglio, "Intersegment Handover between Terrestrial and Satellite segments: Analysis and Performance Evaluations through simulation", IEEE Transactions on Vehicular Technology, vol. 50, n. 3, May 2001, pp. 750-766.

[2] C. Perkins, "IP Mobility Support", RFC 2002 , Oct. 1996.

[3] C. Perkins, "IP Mobility Support for IPv4", RFC 3344, Aug. 2002.

[4] H. Soliman et al., "Hierarchical Mobile IPv6 Mobility Management (HMIPv6)", RFC 4140, Nov. 2005.

[5] W. Hansmann, M. Frank, "On Things to Happen During a TCP Handover," 28th Annual IEEE International Conference on Local Computer Networks (LCN'03), Oct. 2003, Bonn/Königswinter, Germany, pp. 109-118.

[6] W. Stevens, "TCP Congestion Control", IETF RFC 2581, Apr. 1999.

[7] W. Stevens. "TCP/IP Illustrated. Vol. 1", Ed. Addison Wesley, Reading, UK, 1994.

[8] M. Zonoozi, P. Dassanayake, M. Faulkner, "Optimum Hysteresis, Signal Averaging Time and Handover Delay", IEEE Vehicular Technology Conference, Mar. 1997, Phoenix, AZ, USA , vol. 1 pp 310-313.

[9] J-O. Vatn "An experimental study of IEEE 802.11b handover performance and its effect on voice traffic", technical paper, July 2003 `citeseer.ist.psu.edu/vatn03experimental.html`

[10] V. T. Raisinghani and S. Iyer, "ECLAIR: An Efficient Cross Layer Architecture for wireless protocol stacks", in Proc. of World Wireless Congress (WWC04) , May 2004, San Francisco, USA.

[11] V. Kawadia, P. R. Kumar, "A cautionary perspective on cross layer design", IEEE Wireless Communications, Feb. 2005, Vol. 12, Issue 1, pp. 3-11.

[12] P. Karn and C. Partridge, "Improving Round-Trip Time Estimates in Reliable Transport Protocols", ACM Transactions on Computer Systems, Vol. 9, N. 4, Nov. 1991, pp. 364-373.

[13] I. Sommerville, "Software Engineering", Addison-Wesley, 5th ed., Harlow, England, 1996.