WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2013; **13**:212–229
Published online 10 February 2012 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.2197

SPECIAL ISSUE PAPER

A survey on interactive games over mobile networks

M. Gerla[1], D. Maggiorini[2], C. E. Palazzi[3]* and A. Bujari[3]

[1] Computer Science Department, University of California, Los Angeles, Los Angeles, CA, U.S.A.
[2] Department of Information and Communication Science, University of Milan, Milan, Italy
[3] Department of Pure and Applied Mathematics, University of Padua, Padua, Italy

ABSTRACT

The mobile revolution has brought us the possibility to enjoy our favorite applications anywhere and anytime. In this context, interactive games over mobile networks embody a fascinating case study both for their commercial success and for their technical challenges, thus, sparking interest and development. The current state of the art of interactive games over mobile networks is captured in this article. We discuss main requirements and analyze possible combinations of existing solutions to provide better support for highly interactive game sessions with mobile players. Copyright © 2012 John Wiley & Sons, Ltd.

KEYWORDS

videogame; interactivity; mobility; networking

***Correspondence**

C. E. Palazzi, Università degli Studi di Padova, Dipartimento di Matematica, Via Trieste 63, 35131 Padova, Italy.
E-mail: cpalazzi@math.unipd.it

1. INTRODUCTION

Wireless communication has now become widely available in various forms, ensuring wireless connectivity to people in wherever building they are and even when walking around or traveling in cars. In this scenario, the number of mobile users engaged in interactive games is certainly going to grow in the near future. Indeed, interactive games are gaining attention as their users are increasing in number [1,2]. Mobile users are obviously influenced by this success, and it is becoming more and more common to see people in public spaces engaged in some game through their smartphones.

From a research point of view, network support to interactive games must overcome several challenges, especially when considering many mobile players simultaneously sharing the same virtual arena. These technical challenges are now attracting researchers and practitioners, sparking interest, and stimulating innovative developments. This survey points to open research issues and captures the state of the art in the field of interactive games over mobile networks. We focus on highly interactive mobile games, that is, games played through handheld devices and on the basis of fast action evolution on screen, requiring prompt reactions by the user. As depicted in Figure 1, mobile players can be moving inside a house, or walking in a street, or even traveling in cars or by public transport.

With this vision, in this article, we discuss main requirements, challenges, and metrics regarding the quality of gaming as perceived by users. We survey core components needed to ensure a playful experience to users and analyze through experiments the two major communication scenarios supporting a game session: infrastructure-based and infrastructure-less. In these two scenarios, we show how the combination of different solutions emerging from the current state of the art can lead to a significantly improved network support, thus, ensuring interactivity even in challenging mobile scenarios.

The remainder of the paper is organized as follows. Section 2 discusses main networking challenges related to mobile gaming. Sections 3 and 4 discuss the overview of background and related work regarding infrastructure-based and infrastructure-less mobile gaming, respectively. Section 5 discusses main metrics to evaluate the network-related quality of interactive gaming. Focusing on the infrastructure-based wireless communication scenario, Section 6 explains the possible combination of two techniques emerging from the state of the art. Section 7 reports related experimental results. Building on this, Section 8 describes the possible combination of two state-of-the-art solutions for the infrastructure-less wireless communication scenario, and Section 9 reports corresponding experimental results. Finally, Section 10 draws the conclusions.

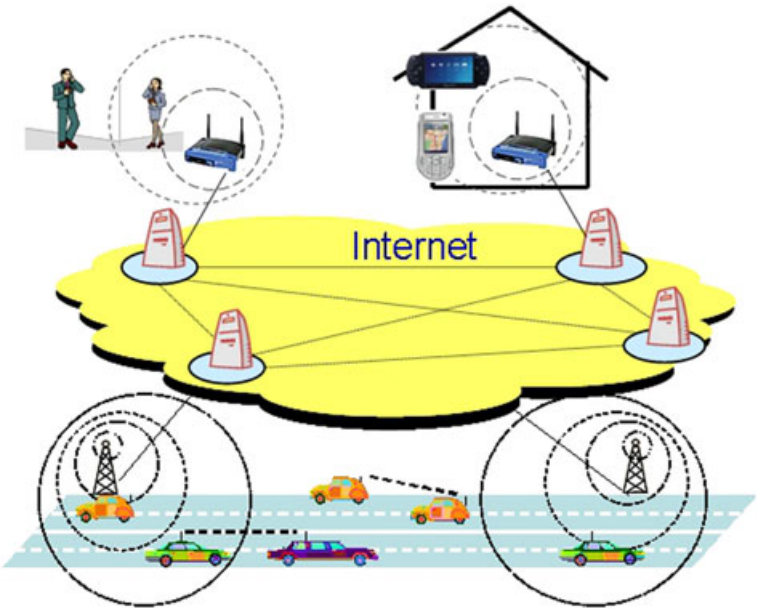

Figure 1. Different scenarios for interactive game entertainment with mobile users (pedestrian, in-home, and vehicular).

2. MOBILE GAMING: CHALLENGES BEYOND THE FUN

In this section, we discuss main networking challenges related to mobile gaming.

2.1. Main networking requirements

From a networking point of view, online games are characterized by five interrelated requirements: interactivity, consistency, fairness, scalability, and continuity.

Interactivity (or **responsiveness**) refers to the delay between the generation of a game event in a node and the time at which other nodes become aware of that event. To ensure an enjoyable game to the final user, the time elapsed from the game event generation at a certain node and its processing time at every other node participating in the same game session must be kept under a certain interactivity threshold [3,4]. Unfortunately, fluctuating congestion conditions in the Internet may suddenly slow down the game fluency on the screen. Moreover, players in the same virtual arena could be so numerous that some game servers may experience a peak of computational load and loose interactivity. These problems are obviously amplified when plunged into a wireless, mobile scenario. In fact, wireless characteristics and node mobility generate a high variability in experienced delays and network traffic. Furthermore, with high mobility, we can expect higher churn rate that worsen the aforementioned problems [5].

Consistency regards the simultaneous uniformity of the game state view in all nodes belonging to the system. As a practical example on how network delays affect consistency, Figure 2 presents a game session frame as seen by a player. The picture presents both the position of a certain participant as seen by another playing client (the full figure avatar) and the position as seen by the server (the human-shaped light boxes). Transmission lag creates a difference in the avatars' positions as perceived by the server and by each of the various clients. The easiest way to guarantee absolute consistency would be that of making the game proceed through discrete locksteps. Having a single move allowed for each player and synchronizing all agents before moving toward the next round surely grants absolute consistency but, on the other hand, impairs the system interactivity. A trade-off issue between consistency and interactivity thus needs to be solved to develop a proficient game platform.

Figure 2. Frame sequence of a game session as seen by a client: other participants are positioned in the game arena with a lag that depends on the distance between clients and server.

Fairness (or **networking fairness**) is related to providing every player with the same chances of winning the

game session, regardless of different network conditions. In this context, relative delays have to be considered as important as absolute ones. Simultaneous game evolution with identical speed should be guaranteed to all participants. To this aim, it has recently been demonstrated how increasing the interactivity degree of the game platform may also lead to improved fairness [6]. Unfortunately, the wireless environment generates peculiar delays and related unfairness problems, especially in ad hoc networks. This unfairness results from the nature of the shared wireless medium and location dependency. If we assume a node and its interfering neighbors to form a neighborhood, the aggregate of local queues at these nodes represents the distributed queue for the neighborhood. Unfortunately, this aggregate queue does not correspond to a First In, First Out queue. Flows sharing the queue have different, dynamically changing delays determined by the topology and traffic patterns. Thus, they obtain different feedback in terms of packet loss rate and packet delay when congestion occurs [7]. Clearly, capture problems in wireless networks, and hence gaming unfairness, are worse in the presence of multihop paths [8–10].

Scalability regards the capability of the system in providing efficient support to a large community of players. Indeed, it is of primary interest for game companies to have revenues generated by a very high number of customers. Besides, humans are social beings who enjoy the presence of others and the competition against real adversaries. Yet, especially in the case of fast-paced games, when additional players cause interactivity threshold violations, scalability is sometimes sacrificed by denying the access to some users depending on their experienced delays [11]. Therefore, by sustaining interactivity, we can also provide a higher scalability degree in terms of both the number and the geographic dispersion of players allowed to participate to the same virtual arena.

Regarding scalability, it is worth noting that wireless connectivity provides means to widen the set of potential players as it is no longer required to be wired to the Internet to engage online games. We are not just referring to players connected through Wi-Fi to the Internet; we also consider outdoor online gaming sessions through 3G, WiMax, vehicular IEEE 802.11p, ad hoc networks, mesh networks, and so on. Clearly, this brings up new issues such as connectivity intermittence and limited energy budget [12–15].

Continuity is concerned with having game sessions not interrupted by disconnections, handoffs, or any other wireless/mobility-related issues. Indeed, players would be very frustrated by having their game session continuously interrupted and restarted (usually after a while) with new players. This problem may also occur when trying to exploit the new wireless capabilities of popular smartphones to create proximity-based gaming sessions. Indeed, we can imagine players forming an ad hoc network to engage in an outdoor multiplayer game on the basis of the connectivity means of their smartphones. Yet, players' movements and different energy consumption rates may create detached cluster of nodes [16,17]. To this aim, proposed mobile solutions feature games with short game sessions, quick/smart handoff mechanisms, and route/server migration capabilities[18–21].

2.2. The wireless scenario

Wireless networks can be divided into two main classes depending on whether they use fixed infrastructures or not. Infrastructure-based wireless networks extend the Internet over the wireless domain. Wireless nodes connect to an access point (AP) to have access to Internet services, and each AP offers to all engaged wireless nodes the same functionalities. A mobile node could move out of the transmission range of an AP, thus, losing its connectivity to the Internet. In this case, if another AP is available in the new location, the node might connect to the Internet again through the new AP; in the meantime, however, the wireless node has probably lost its ongoing sessions. This problem can be solved through a smooth handoff that seamlessly transfers the connectivity from the old to the new AP before disconnection. A well-known protocol to perform this task is represented by Mobile IP that transfers packets from the old AP to a new one through routing triangulation [22]. The handoff problem can be avoided altogether if the mobile can maintain multiple connections simultaneously via different APs. This is possible if the device has multiple radio interfaces, a reasonable proposition for vehicles. Across the multiple connections, Forward Error Correction Codes or Network Codes can be used for extra robustness.

Ad hoc networks are composed of several nodes with wireless connectivity capabilities that connect one another to establish communications without the need of any AP. Every node in the network is able to communicate with any other node in its transmission range. Transmissions can happen both directly between two end hosts that are close to each other or through multihop paths when a node needs to send a message to another node that is out of range. Ad hoc networks can be composed of static nodes, but the most challenging case is clearly when nodes move: they are also known as mobile ad hoc networks (MANETs) [23]. Ad hoc networks have received the attention of researchers and practitioners, thanks to their high deployment flexibility, low cost, and robustness, which make them perfectly suitable for a whole plethora of scenarios where the infrastructure is missing, for example, away from towns, in areas hit by a major disaster or just not covered by APs, and in military battlefields.

Vehicular ad hoc networks (VANETs) represent a special kind of MANETs where nodes are vehicles moving at high speed, and challenges are made tougher by the very high mobility of vehicles [24,25]. Drivers could easily be in a situation where they find no AP along the road, thus, raising the interest for ad hoc connectivity among vehicles. Indeed, several useful applications could be run over ad hoc networks composed by vehicles: safe driving, text/audio/video chats, peer-to-peer file sharing,

and even the online games that are the focus of our analysis. VANETs exacerbate mobility problems in ad hoc networks such as interference and sudden congestion. Yet, their nodes generally do not experience energy shortage.

Depending on the considered application, two possible transmission models can be considered for VANETs: the pull model and the push model [26]. With the former, vehicles explicitly ask to receive certain data from other vehicles, whereas with the latter, messages are proactively broadcast to every node in a certain area of interest. Both models can involve multihop transmissions. Because of the broadcast nature of the wireless channel, communications involving several nodes in the same VANET (e.g., online games) are clearly more efficient if performed through broadcasting (i.e., push model).

In the next two sections, we explore the two main scenarios for mobile players: infrastructure-based and infrastructure-less wireless communication.

3. INFRASTRUCTURE-BASED MOBILE GAMES: BACKGROUND

With Wi-Fi connectivity, users can gather around an AP to be engaged in an interactive gaming session or to connect to remote players or game servers. When considering a wireless home, this configuration represents a simple wireless version of regular online gaming. Yet, we may imagine similar configuration available in public areas. In any case, the AP could be employed for heterogeneous flows and applications, both elastic and real-time, thus, introducing delays that would jeopardize the performance of interactive games [12]. Furthermore, besides infrastructure-based communication, Wi-Fi also allows for ad hoc connectivity. In this configuration, Wi-Fi is able to support a higher number of players and at larger distance from each other (even 50 m) than Bluetooth. Yet, energy consumption is increased.

Mesh networks represent a way to extend the range of a wireless infrastructure, thanks to the possibility to create wireless chains among APs. A user can hence connect while being more than one hop away from the wired infrastructure. This network topology can support infrastructure-based wireless online games where players are located and moving in an area wider than one single AP's transmission range. On the other hand, users will have to deal with handoffs and with multihop communication, thus, facing the risk of high delay peaks in delivering and receiving game events.

Many researchers have focused their studies on the issues encountered in a wireless environment; we are interested here in works related with performance issues emerging when supporting online games through wireless networks.

3.1. The delivery delay issue: related work

We start with the main issue in the context of online games: the need for fast game event delivery to players, yet we focus on approaches specifically designed for a wireless scenario. Many research papers focusing on IEEE 802.11 present analysis, problems, and solutions. Nonetheless, the vast majority of them provide results that focus on a throughput/loss point of view [27–32], whereas the performance of real-time applications depends on the measured per-packet delay and jitter [33]. An insightful analysis of this issue with respect to online games is still missing, as well as efficient solutions aimed at reducing queuing delay over wireless links [34,35].

Focusing on medium access control (MAC) layer retransmissions, Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) flows have been tested on IEEE 802.11 wireless links; outcomes highlight that, without retransmissions implemented at the link layer, the loss rate becomes unacceptable for any application [36,37]. Yet, the current number of MAC layer retransmissions may not represent the optimal choice to support both TCP-based traffic and real-time applications. Indeed, a high number of repeated retransmissions could still be not enough to prevent TCP from incurring a loss. At the same time, MAC retransmissions can be wasteful and potentially harmful for time-sensitive applications, such as real-time video/audio or online games over UDP as they introduce delays [34].

The coexistence of several users running heterogeneous applications through the same AP can be a source of performance problems especially for real-time applications. Persistent TCP-based flows (e.g., downloads) are responsible for performance deterioration of concurrent UDP-based flows (e.g., interactive games). Indeed, the continuous search for more bandwidth performed by TCP's congestion control algorithm creates queues at buffers, thus, augmenting the per-packet delay of any flow sharing the same channel.

Aiming at ensuring low per-packet delays, while preserving downloading throughput, [38] proposed a solution based on a Smart AP (SAP). SAP monitors all traffic passing through and, for each TCP flow, computes the maximum transmission rate so as not to exceed the channel capacity and accumulate packets in queue. Only standard features and protocols are used to facilitate deployment: the maximum transmission rate for each flow is enforced by modifying on the fly the advertised window in TCP ACK packets.

3.2. Architectures: client–server, peer-to-peer, and hybrid

Typically, network architectures supporting online games can be distinguished on the basis of three main categories: centralized client–server, fully distributed, and mirrored game server.

The centralized client–server architecture is composed of a single authoritative point (the server) that is responsible to run the main logic of the game, execute players' commands, enforce consistency, send back to the client the new game state update, and so on. Clients have only to

receive the new game state update, render it on the screen, and forward player's commands. The server can be both a single computer or cluster of computers to increase the performance of the system [39]. The centralized client–server architecture represents the simplest solution for authentication procedures, security issues, and consistency maintenance [40–43]. Moreover, assuming to have N simultaneous players, the generated messages are in the order of $O(N)$. On the other hand, the unique bottleneck limits the efficiency and scalability of this solution.

Fully distributed architectures are well represented by the peer-to-peer paradigm. In this case, all the involved nodes share the same intelligence and are equally responsible for running the whole logic of the system. In this case, in fact, each client has to autonomously update the game state view on the basis of its player's commands and game actions received from other players. This obviously requires terminals endowed with higher computational capabilities.

The main advantage in employing a fully distributed architecture resides in spreading the traffic load among many nodes, thus, generating a more scalable and failure resilient system [43,44]. However, identical copies of the current game state need to be stored at each node. This requires some complex coordination scheme among peers; in fact, this scheme has to be distributed over the set of involved nodes and has to guarantee the coherence of all game state views. The exchanged messages could hence raise to the order of $O(N^2)$. Finally, authentication, cheating, and general consensus among all the peers are harder to be addressed than when a centralized architecture is employed. Yet, this kind of solution is generally preferred for networks such as MANETs and VANETs because of their ability to deal with high mobility of nodes that continuously change the topology of the network [19].

Mirrored game server architectures represent a hybrid solution that efficiently embraces all the positive aspects of both centralized client–server and fully distributed architectures [45]. On the basis of this approach, Game State Servers (GSSs) are interconnected in a peer-to-peer fashion over the Internet and contain replicas of the same game state view. Players communicate with their closest GSS through the client–server paradigm. Each GSS gathers all the game events of its engaged players, updates the game state, and regularly forwards it to all its players and GSS peers.

Advantages in employing mirrored game server architecture are the absence of a single point of failure, the networking complexity maintained at server side, and the possibility to easily implement authentication procedures. Even if synchronization is still required to ensure the global consistency of the game state held by the various servers, this requirement is made easier with respect to fully distributed architectures, thanks to the lower number of involved nodes. Assuming to have N players and M GSSs, for example, the generated game messages amount to $O(N + M)$, which is again $O(N)$ unless considering the unlikely case of having more servers than players.

The presence of multiple high-performance GSSs helps in distributing the traffic over the system and reduces the processing burden at each node [44]. Moreover, having each player connected to a close GSS reduces the impact of the player-dependent access technology (e.g., dial-up, cable, and *x*DSL) on the total delay experienced [46]. In this case, in fact, the communication among players results mainly deployed over links physically connecting GSSs, which can exploit the fastest available technology (e.g., optical fibers) to reduce latency. As a result, through this architecture, it becomes simpler to adopt efficient solutions for the tradeoff among interactivity, consistency, fairness, and scalability.

For instance, [47] suggested that during a game session, some events can lose their significance as time passes. When there is a rapid succession of movements performed by a single avatar in a virtual world, the event representing the last destination supersedes the previous ones. In general, certain new actions may make the previous ones irrelevant; discarding superseded events for processing fresher ones may be of great help for delay-affected GSSs, achieving high interactivity degree without compromising consistency. Furthermore, for very fast-paced games, little inconsistencies are not highly deleterious for players' fun. In these cases, even some nonsuperseded game event could be dropped when dropping all superseded ones is not yet sufficient to maintain an adequate level of responsiveness.

Given the available architectural solutions, we can hence infer that when considering a (moderately) mobile network of players supported by infrastructure and APs, the hybrid solution of mirrored servers probably represents the best choice. Conversely, in case of no available infrastructure or very high mobility of players (e.g., cars' passengers), then a fully distributed solution may be the only feasible option.

4. INFRASTRUCTURE-LESS MOBILE GAMES: BACKGROUND

Focusing on an outdoor gaming perspective, we can imagine players engaging in mobile, connectivity-based and proximity-based games [48]. In this context, users' smartphones will dynamically connect to each other creating a mobile ad hoc network (MANET) dedicated to multiplayer interactive games. This combination represents a very challenging context, and three major issues can be identified.

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 [1,49]. Instead, outdoor gaming sessions may resort to ad hoc connectivity among nodes, thus, complicating the adaptation of existing successful online games for home desktops; in particular, one of the players' mobile devices will also have to act as the game server.

Second, MANETs are prone to disconnections. Because of mobility, one or more nodes may be out of range of

the original ad hoc network, becoming unable to reach the game server node and continue playing. Even worse, the server node itself may be out of range for the rest of the network, thus, interrupting the game session for everybody.

Third, although energy consumption is not an issue for home players, it certainly becomes a major concern when considering outdoor games on the basis of the connectivity of small devices because of 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 node will experience a much faster decline of its energy reserve as it will have to receive all game events from other players, compute game state updates, and transmit these updates back to the other players.

Many solutions have been proposed to ensure data transmissions over MANETs while aiming at low energy consumption [50,51]. Yet, these solutions cannot be directly applied to a MANET supporting online interactive gaming as they are generally based on having nodes alternating sleep/awake modes. Clearly, this class of solutions can be applied to sensor networks or, in general, to MANETs not supporting real-time applications, but it cannot be exploited for interactive online games.

Other solutions exist that aim at saving nodes' energy through smart routing in a MANET [15,17]. Yet, they do not consider any delivery delay options, thus, not necessarily ensuring interactivity. Moreover, the scenario with one of the nodes also embodying the game server is not considered. In this scenario, one single node (i.e., the game server) receives and generates most of the traffic. The application itself and the network architecture impose unbalanced energy consumption. Thereby, the server (and player) node will run out of energy much before the other player nodes interrupting the gaming session for every node.

To address this issue, [19] proposed to utilize three kinds of nodes in the game network: *active servers*, *backup servers*, and *players*. In essence, the active server acts as a regular game server, whereas some backup servers keep updated their game state view, 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 [52]. In this solution, all servers collect game events sent by other players and continuously 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.

Finally, a particularly interesting case study in the infrastructure-less scenario is represented by vehicular communication. In essence, the advent of vehicular communication, boosted by the new IEEE 802.11p standard, has attracted many researchers in this research field. Many applications, from safety-related to commercial ones, will soon be available from connected devices in our cars. Among them, entertainment applications such as interactive games will probably be a successful one [53]. Yet, having many connected players moving fast in their cars certainly represents one of the most challenging and interesting context for infrastructure-less mobile games. Fundamental issues for interactive games such as the quick propagation of game events among players are exacerbated in this context and require special attention [33,54,55].

4.1. The vehicular environment

The IEEE 802.11p is emerging as the standard for supporting vehicular networks [56]. In this scenario, passengers on cars or buses could engage in gaming sessions with neighbors (ad hoc mode) or through the Internet (infrastructure-based mode). Energy consumption could be a problem if players will still use handheld devices, whereas when using on-board devices, the vehicle's engine will provide all the energy needed. Yet, fast mobility represents a tough issue both for ensuring connectivity and for the high variability in concurrent network traffic [20]. Indeed, because of the topology variability of a vehicular network, vehicles around a certain player and their concurrent data traffic may continuously and quickly change, creating sudden delays in the delivery of game events and jeopardizing the interactivity of the gaming session. [53]

Sending online game messages among (many) players in a VANET amounts to propagating game events over a (multihop) wireless channel, where the recipients can be many even within the same transmission range. This clearly corresponds to the push model mentioned in Section 2.2 and is hence more efficiently resolved through (multihop) broadcast rather than resorting to many resource-consuming unicast transmissions. Therefore, one of the main problems regards fast broadcasting of a message to all cars in a given strip-shaped area of interest [57].

The vehicular communication literature reports that the main reasons behind a slow broadcast delivery lies in a non optimal in number of hops experienced by a message to cover all the involved cars and, more in general, to an excessive number of vehicles that try to simultaneously forward the message [57,59]. To tackle this problem, a theoretically optimal broadcast algorithm has been recently proposed, which propagates messages to cars making use of the notion of Minimum Connected Dominating Set [60]. This leads to great practical difficulties in the implementation of such algorithm as it would require a complete and continuously updated knowledge of the network topology. For instance, in an attempt to implement this algorithm with N cars, its authors have developed a scheme employing as many as $O(NlogN)$ control messages [61]. It goes without saying that this is not a scalable solution.

Addressing the fast-delivery broadcast problem from a more practical standpoint, various 802.11-based solutions have been proposed. For example, [57] proposed a back-off

mechanism that reduces the frequency of message retransmissions when congestion is causing collisions. In [55], instead, as soon as a car receives a broadcast message from a following vehicle along a strip, it refrains from forwarding it as the reception of this message is a clear confirmation that subsequent cars have already received it. Unfortunately, neither schemes considers a very important factor in determining the final propagation delay of a message: the number of hops a broadcast message traverses before covering its whole area of interest.

The solution presented in [62] utilizes a distributed proactive clustering scheme to dynamically create an efficient virtual backbone infrastructure in the vehicular network. The backbone formation process takes into consideration both the current distance among candidate backbone vehicles and the estimated lifetime of the wireless connection among neighbor backbone members. The attempt is that of improving the robustness and lifetime of connections among backbone members even in a highly mobile scenario as a vehicular network.

In [59], the minimization of the number of hops is achieved by individuating the farthest car within the source's backward transmission range, which has to forward the message. To this aim, jamming signals are emitted by each car with a duration that is directly proportional to the distance between the considered car and the message's source. The car with the longest jamming signal is clearly the farthest car from the source. Even if this guarantees a minimum number of hops to cover the whole area of interest, the time wasted to determine the next forwarder through jamming signals could make this scheme unsuitable for a tight time delay scenario as the one we are considering.

A scheme trying to statistically achieve a minimum number of hops when propagating a broadcasted message is discussed in [58]. In particular, different contention windows (CWs) are assigned to each car to have different waiting times before propagating the broadcast message. Nodes set their respective CWs with an inverse proportion of the distance from the sender, thus, needing less forwarders (and transmissions) to cover a certain area. Yet, this scheme assumes that there is a unique and constant transmission range for all cars in every moment; this is obviously not realistic in a VANET because of its high and fast mobility.

Other solutions have hence been devised to solve this shortcoming. In particular, similar automatic transmission range estimators are proposed in [2] and [13] to assess the actual transmission range for every car in the platoon. More in detail, the former exploits this information to have the farthest vehicle (i.e., the farthest relay) in the sender's transmission range becoming the next forwarder of the message. Instead, the latter uses this information to assign the forwarding task to the farthest spanning relay. In both cases, the computed transmission range estimation is used to support a multihop broadcasting scheme for message exchange able to dynamically adapt to the different (transmission range) conditions a vehicular network may encounter. Yet, [2] is specifically designed to support online games and includes some optimization, e.g., utilizing information included in regular game messages, further limiting the overhead.

5. QUALITY OF SERVICE VS. QUALITY OF GAMING

To achieve a better understanding of the real correlation between network delay and players' expected performance, we present here an objective assessment methodology to investigate the impact of end-to-end delay to gameplay. As already mentioned, we focus on a first-person shooter (FPS) because of the popularity and real-time requirements of this kind of game. In particular, we consider Quake III, a well-known and widely popular online multiplayer FPS. Moreover, this game has been released as open-source, with also bots available, for research purposes: IOQuake III [63].

In our methodology, we used a testbed platform. This testbed is composed of a number of clients and one server where each client runs an autonomous synthetic player with uniform and controlled skills, as available on the game's official web site. Using this testbed, we have been able to run multiple and replicable experiments where different network conditions can be taken into account while providing indexes related to player's in-game performance.

Results will be useful for a game provider to perform preemptive traffic engineering, evaluation of network infrastructures, and true-skill matching between players taking network handicap into account.

This section demonstrates with a real and objective case studio the impact of network delays on gaming performance, thus, justifying our effort in Sections 6 and 8 to individuate solutions able to ensure quick delivery of game events.

5.1. Metric validation

The final score of a game session can be considered a good metric to understand if a player is placed in the arena in a fair way or, in general, if he or she is playing in a satisfying way.

To this aim, Figure 3 outlines the complementary cumulative distributed function of the score when no player is subject to any delay. As can it be observed, the game session is fair because there is a very high probability for each player to reach a score of at least 25, but there is small evidence of an effective balance of skills in the score range from 30 to 40 because of visible oscillations. These oscillations are caused by the fact that Quake III allows only one winner for every game session: the one reaching a score of 40 points; the other players are left behind by one or two points, therefore resulting as losers even if the quality of their game session has been the same as the winner's one.

Penalizing one of the players in terms of experienced network delay brings the obvious result as presented in

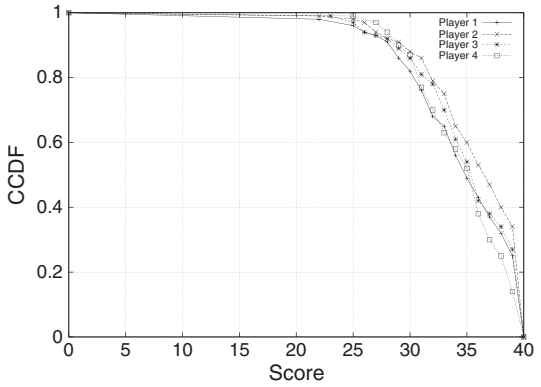

Figure 3. Final score complementary cumulative distributed function (CCDF) with no network delay.

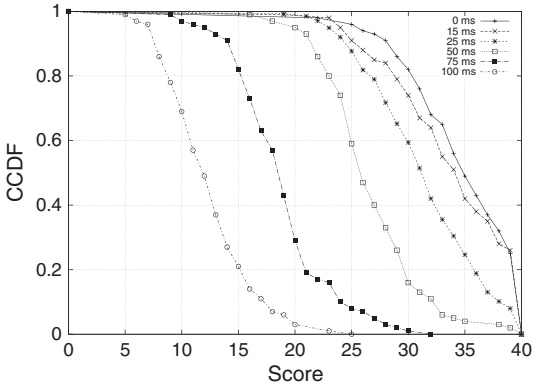

Figure 5. Score performance degradation based on player's delay.

Figure 4. Although commercial FPS vendors claim that their products are playable with up to 150-ms delay on the round-trip, we can clearly observe performance degradation for one player with just 25 ms (50-ms round-trip delay). In a real-life environment, this situation could be mistakenly classified as fair on the basis of the fact that the probability for the player to score at least 25 is still very high despite his or her decreased chances to win.

An interesting consideration comes from the observation that we are not actually able to detect small changes in behavior: a small built-in handicap may be overlooked and consequently ignored due to the aforementioned oscillations. This is not a crucial problem in the current scenario, but it might be so in other games, depending on the performed activities.

Performance degradation based on delay seems to follow an exponential behavior, as depicted in Figure 5. Starting from 50 ms of delay Ð, which is close to the threshold where an FPS starts to be less playable Ð, we can actually see a significant decrease for the player to reach a score of 25 points; hence, a human player will actually start to complain.

The outcome also shows that the rate of actions Ð or rate of frags, in this specific case Ð, is affected by network delay. In particular, Figure 6 is the equivalent of Figure 5 but using interfrag time distribution and confirms previous considerations about player's performance: degradation with delay is exponential, and even just 50 ms of network delay represents a serious handicap for players. In particular, during the experiments, the probability to wait more than 20 s to hit a player increases by 15% with 50-ms, 30% with 75-ms, and 40% with 100-ms delays.

6. CASE STUDY #1: INFRASTRUCTURE-BASED MOBILE GAME

In this section, we analyze a scenario where a mobile game is supported by an infrastructure for wireless communication. In this context, we have considered state-of-the-art

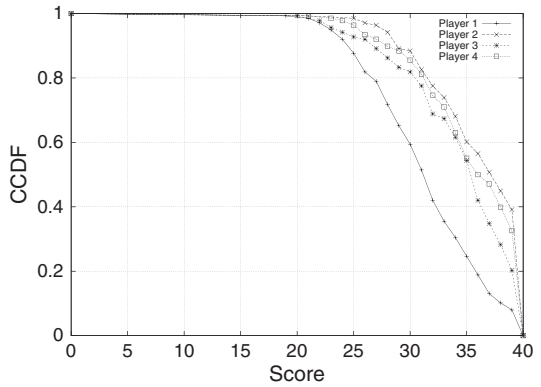

Figure 4. Final score complementary cumulative distributed function (CCDF) with one player experiencing 25 ms of network delay.

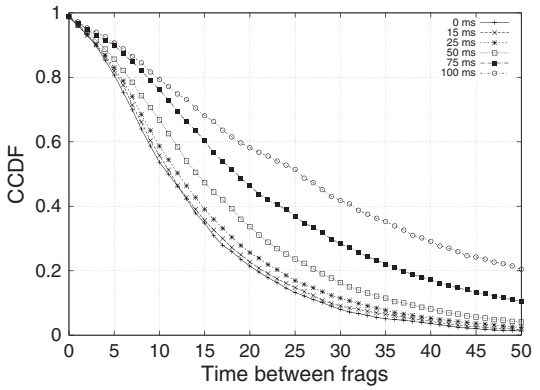

Figure 6. Fragging performance degradation based on player's experienced network delay. CCDF, complementary cumulative distributed function.

solutions to determine how the latest technology would perform. We have considered the various solutions surveyed in Section 3 and selected two of them that could produce even better results if integrated. In particular, we have combined the hybrid architecture presented in [47] for the core network with the SAP for the last mile wireless connectivity proposed in [38]. In the following sections, we provide further details on the two solutions and their possible integration into a new solution that combines their advantages.

6.1. Fast synchronization over a hybrid architecture

The hybrid architecture exploiting mirrored game servers for online games discussed in Section 5 emerges as the most promising one when considering a scenario based on infrastructure for wireless communication. Even if some synchronization burden is required to ensure the global consistency of the game state, this requirement is made easier, thanks to the lower number of involved GSSs. Moreover, as already mentioned, the synchronization mechanism among GSSs could smartly exploit the semantics of the game to discard few game packets in order to avoid interactivity loss when intense network traffic or computational load is slowing down the system [45,47].

For the sake of clarity, in the rest of the paper, we will refer to this synchronization mechanism, which is able to discard game events, as *Fast Synchronization* (FS).

With FS, each GSS continuously monitors the Game Time Delivery (GTD) of game events, which represents the time elapsed between the generation and the delivery of game events. Upon each packet arrival, each GSS determines the GTD of the arrived event, namely $sample\_GTD$, and feeds a low pass filter to compute the updated average GTD, namely avg_GTD. When avg_GTD exceeds a certain threshold in a GSS, that GSS drops superseded events with a certain probability p, without processing of forwarding them. If avg_GTD exceeds a subsequent limit, p is set equal to 1, and all superseded events waiting for being processed are discarded.

In Figure 7, two discarding functions of FS are depicted. The leftmost one is used to discard only superseded events, whereas the rightmost one can be used in case of heavy delays to discard any game event to restore interactivity, even if compromising consistency.

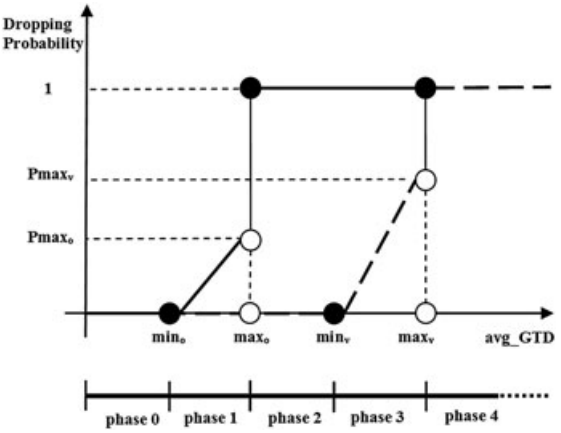

Figure 7. Discarding probability functions.

6.2. A Smart AP to save the interactivity investment

Even if FS coupled with a mirrored game server architecture is proficient in maintaining a high degree of responsiveness among GSSs, problems may still arise at the edges of the considered topology, where users in their homes or along a street may be engaged in an online game through an AP (see scenarios depicted in Figure 1). Concurrent traffic may generate queues that build up at the AP, thus, delaying the game event delivery and wasting all the interactivity investment accumulated by FS. The applications run in this context may vary, and some of these may be particularly harmful toward real-time traffic (not only online games but also video streaming, video chats, etc.). In particular, TCP-based File Transfer Protocol application for downloading files increases queuing delays to such an extent that interactivity may be completely jeopardized [34].

To this aim, we should integrate the aforementioned hybrid architecture with the SAP solution anticipated in Section 3 [38]. This solution aims at achieving best performance for both elastic and real-time applications by appropriately limiting the advertised window for TCP flows. In essence, an AP monitors all data flows passing through it; the AP is indeed in the position to know the number and bandwidth consumption of the various TCP-based and UDP-based flows that regard its local wireless nodes. SAP utilizes the advertised window, present in every TCP ACK packet, to set an upper limit to the data rate of the various TCP flows. This maximum data rate for each TCP flow is computed to leave enough space for the UDP-based (real-time) flows and avoid buffer congestion, while maintaining the throughput that would be achieved by TCP-based flows in a regular scenario.

An optimal tradeoff between throughput and low delays is achieved by maintaining the sending rate (hence, the sending window) of TCP flows high enough to efficiently utilize all available bandwidth while, at the same time, limiting its window growth so as not to overflow the buffers. As a result, the throughput would be maximized by the absence of packet loss, whereas the delay would be minimized by the absence of queuing. This can be achieved through limiting the aggregate bandwidth utilized by TCP flows just below the total capacity of the bottleneck link diminished by the portion of the channel occupied by the simultaneous UDP-based real-time traffic.

In essence, the maximum sending rate for each TCP flows at time t, namely $TCPubrate(t)$, can be represented by the following equation:

$$TCPubrate(t) = \frac{(C - UDPtraffic(t))}{\#TCPflows(t)} \quad (1)$$

where $UDPtraffic(t)$ is the amount of bandwidth occupied by UDP-based traffic at time t, $\#TCPflows(t)$ is the concurrent number of TCP flows, and C is the total capacity of the bottleneck link.

This upper bound can be enforced for all TCP flows sharing the same wireless link by having the corresponding AP exploiting the advertised window field present in every TCP ACK packet. Simply, the actual sending window of a TCP flow is determined as the minimum between the congestion window and the advertised window; thereby, having the AP appropriately modifying the advertised window of passing through TCP ACK packets would limit the factual sending rate of TCP flows under $TCPubrate(t)$.

7. CASE STUDY #1: EXPERIMENTAL EVALUATION

In this section, we evaluate through simulations the interactivity improvement ensured by the holistic solution discussed in the previous section. For the sake of clarity, we refer to it with the name of *Smart Architecture* (SMA).

7.1. Simulation assessment

To evaluate the discussed solution, realistic game traffic was generated to have a lognormal distribution of the GTD values as suggested by [64]. This traffic was utilized to feed the well-known NS-2 simulator [65]. In particular, the simulated scenario includes seven interconnected GSSs with various network latencies between any two. To perform measurements, we focus on one GSS that is connected to all other GSSs. The network latency between the considered GSS and the others spans uniformly between 40 ms and 90 ms. The simulation environment is configured to let movement game events being superseded by successive ones.

Results were gathered collecting the total latency experienced by game events reaching one of the clients connected through an IEEE802.11g AP to one of the GSS. The same AP was also in charge of handling traffic coming from other applications run on different devices that were simultaneously sharing that wireless link: a UDP-based video stream, a UDP-based live video chat, and a TCP-based downloading session. The video stream and video chat applications were simulated by injecting in NS-2 real traces corresponding to high quality MPEG4 Star Wars IV and VBR H.263 Lecture Room-Cam, respectively, as available in [66].

Interactivity represents the main feature determining the perceived quality of an online game service, even when players employ personal digital assistants or cellphones. Therefore, in the next section, we report results related to the GTD experienced by game events delivered to the considered mobile gaming device: the smaller the GTD, the higher the perceived interactivity.

In particular, in the following charts, REG represents the case where a regular synchronization scheme is adopted by GSSs, whereas SMA represents the case employing the discussed SMA. Finally, GIT stands for *Game Interactivity Threshold* and represents the maximum delay that a game event can experience from its generation to its delivery to the player's device to still consider the ongoing gaming session as interactive. As a reference, a GIT of 150 ms was considered as suggested by related scientific literature for interactive online games [3].

7.2. Results

Focusing on the interactivity benefits provided by the first component of SMA, that is the FS coupled with a mirrored game server architecture (see Section 6.2), we show in Figure 8 the maximum, the average, and the standard deviation of the delivery time that game events experience to reach the GSS supporting the considered player. Basically, the chart reports statistical values about the time elapsed from the generation of a game event and its delivery to the GSS that will then forward it to the considered player.

Clearly, SMA outperforms REG, which demonstrates the effectiveness of FS in quickly synchronizing GSSs.

Instead, Figure 9 regards the impact of the wireless last hop on the game interactivity. In particular, the chart

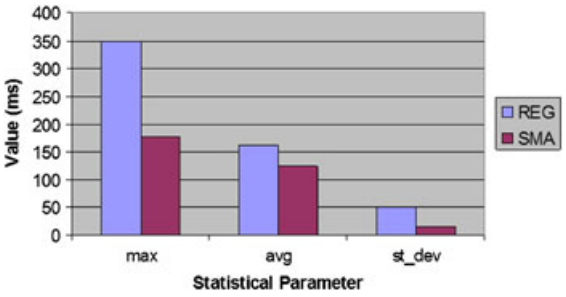

Figure 8. Fast Synchronization's evaluation: statistical values for the delivery delay of game events (packets) from their generation to the Game State Server engaging the considered player.

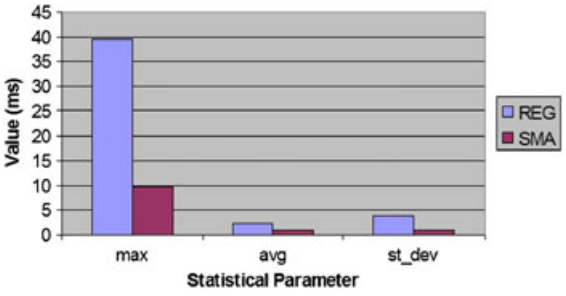

Figure 9. Smart Access Point's evaluation: statistical values for the delivery delay of game events (packets) from the access point to the considered player.

reports statistical values related to time elapsed from the moment when the AP receives the game update from its GSS to the moment when the considered player actually receives it on her/his mobile device. Basically, this figure highlights the effectiveness of employing the SAP solution explained in Section 6.2 versus a traditional one.

Combining the two components of SMA (FS and SAP) can sum the positive benefits seen in Figures 8 and 9. Indeed, Figure 10 confirms this expectation. In particular, Figure 10 reports the delivery time of 100 consecutive game events that have to be delivered to the considered player through the whole gaming platform. The outcome clearly demonstrates how SMA outperforms REG. Moreover, SMA is able to keep the game event delivery time almost always under the interactivity threshold (GIT), exceeding that threshold only rarely and in those case by just a slight amount. Conversely, with REG, delivery delays frequently and significantly exceed the GIT.

This demonstrates the ability of the SMA in factually ensuring a high interactivity degree to mobile players in an infrastructure-based wireless environment.

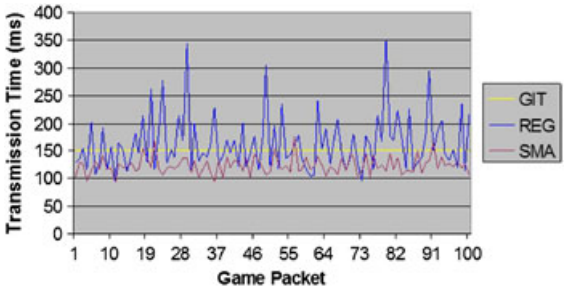

Figure 10. Smart Architecture's comprehensive evaluation: instantaneous delivery latency of game events (packets) over the whole game platform.

8. CASE STUDY #2: INFRASTRUCTURE-LESS MOBILE GAME

In this section, we analyze the scenario where a wireless communication infrastructure may not be available or not usable because of the very high mobility of nodes. A challenging example is embodied by a vehicular network with passengers engaged in an interactive game session. In this case, vehicles could experience a more stable connectivity by staying connected directly with each other rather than resorting to the Internet through sporadic and range-limited APs along the road. Indeed, having games played by passengers in a vehicular network represents the most challenging version of ad hoc networks for mobile players.

In this situation, the combination of the two approaches proposed in [2] and [58], respectively, and discussed in Section 4 can embody an effective solution. Indeed, the combined solution would factually speed up the propagation of game events in the vehicular network by having as few relays (and network traffic) as possible to cover the whole set of players. For the sake of clarity, in the rest of the paper, we refer to this combined solution as *Fast Multi-Broadcast Protocol* (FMBP).

The FMBP is run by all vehicles whose passengers are engaged in the online game session. As said, part of FMBP is inspired by [58], whose rationale is clearer with the help of Figure 11. The picture shows a group of cars belonging to the same vehicular network and moving right to left. For simplicity, we assume a linear topology and a game event generated from car A to be broadcast backward toward all following vehicles till car K. This is just a representative example, but the scenario and the solution could easily be extended to work even when the game event has to be propagated in multiple directions. For instance, messages could be propagated backward and forward in case of linear topology or could be propagated backward/forward and perpendicularly in case of more complex topologies.

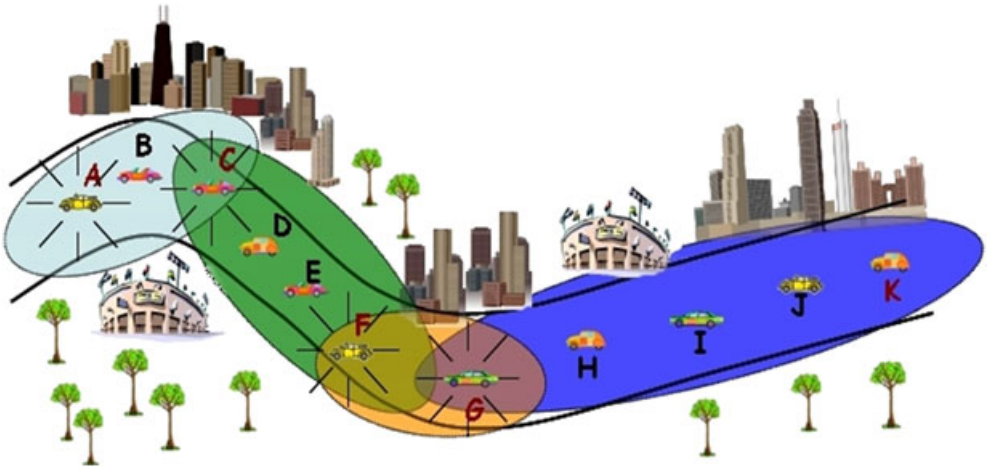

Figure 11. Transmission ranges in a gaming car platoon: an example.

Cars in Figure 11 are located 200 m apart, and the transmission range along the road is variable because of environmental conditions. Each circled area represents the backward transmission range of the leftmost vehicle in that area. Therefore, car A has a transmission range of about 400 m, thus, being able to reach cars B and C within a single transmission hop; then, car C has a transmission range of about 600 m, thus, being able to be heard directly by cars D, E, and F; and so forth. In this situation, if we pretend that car A sends out a game message that has to reach all vehicles in the gaming car platoon, then the optimal solution is represented by having only cars C, F, and G forwarding it.

However, this optimal solution can be generated only with cars aware of their position within the sender's transmission range. Therefore, the efficiency of this solution can be significantly improved by integrating the transmission range estimator suggested by [2]. By including this estimation in game messages, cars C, F, and G can realize that they probably are the farthest cars in the transmission range of the previous sender and decide to be the next forwarder.

In the following sections, we explain in more detail the two main components of FMBP: the transmission range estimator and the game event propagation scheme [2,58].

8.1. Transmission range estimation

To compute transmission range estimations, vehicles can rely on regular game event transmissions. In particular, each vehicle includes information about the range of transmissions that it has been able to hear and, at the same time, it collects data included in game messages sent by other vehicles. Consequently, a vehicle can update its transmission range every time a game message is received from some other vehicle (note: we assume all cars use the same transmit power).

More in detail, every game message generated by a vehicle includes its own position. Moreover, for any considered direction of propagation, two other parameters are present: the distance from which other vehicles have been sensed and the current transmission range estimation, both regarding the considered direction. In particular, if we consider for simplicity (and with no loss of generality) only the case where game events have to be transmitted backward, then we should consider its *forward maximum distance* (FMD) parameter and its backward maximum range (BMR) estimation.

Vehicles exploit game messages' fields about position and FMD to compute their BMR. Both the longest distance from which another vehicle has been sensed sending a game message and the longest maximum distance advertised by received game messages are employed. BMR is obtained by considering only game messages coming from following vehicles; its value is computed as the largest among all their included FMD values and all distances from vehicles that generated them.

Because we are considering only the case where game messages are always sent backward, we have the following purpose and semantics for the information included by FMBP in each broadcast message:

(i) Game messages received from the front allow the receiver to compute FMD; its value will then be declared by the receiver in its game messages to claim: '*This FMD value represents the farthest distance from which I have been able to sense another car in front of me*'.
(ii) Game messages received from the back includes the sender's FMD and position. They hence provide the receiver with information about the hearing capabilities of following cars. This is what the receiver needs to know to compute its BMR, which will then be sent along with game messages to declare: '*This BMR value is the maximum backward distance at which some car would be able to sense me*'.

8.2. Game event propagation

Upon players' actions, game messages are broadcast along the vehicular network. Each of these messages includes not only information about the game evolution but also the sender's position and its current transmission ranges in all considered directions of propagation. To avoid cyclic back and forth transmissions of the same game event, each message also includes its propagation direction and a unique identifier.

The transmission range values are used by vehicles on the message's path to determine which one among them will have to take upon itself the task of forwarding it onto the next hop in the considered direction. Because the aim is that of minimizing the number of hops to reduce the propagation delay, the farthest possible vehicle from the sending one should become the next forwarder. Therefore, the longer the relative distance of the considered vehicle from the sender with respect to the transmission range estimation, the higher the priority of the considered vehicle in becoming the next forwarder.

Vehicles' priorities to forward a game message are determined by assigning different waiting times from the reception of the message to the time at which they will try to forward it. This waiting time is randomly computed on the basis of a CW value, as inspired by classical back-off mechanisms in IEEE 802.11.

If, while waiting, some farther vehicle with respect to the direction of propagation had already forwarded the game message, all vehicles between the sender and the forwarder abort their countdowns to transmission as the message has already passed past them. Instead, all vehicles after the forwarder in the direction of propagation will participate to a new forwarding contest for the next hop. Obviously, the larger the CW utilized by a vehicle, the more likely are some other vehicles to be quicker in forwarding the message. Transmission range values advertised in the game message are updated at each hop with the value computed

by the forwarder; thus, on each hop, proper transmission range estimation is computed through information related to that specific area.

The CW of each vehicle is measured in slots and varied between a minimum value ($CW_{\min}$) and a maximum one ($CW_{\max}$). More in detail, it depends on the distance from the sending/forwarding vehicle (d) and on the advertised estimated transmission range (Tx). To this aim, Equation 2 can be used where Tx changes depending on the considered direction of propagation (for instance, it corresponds to the aforementioned BMR when transmitting backward).

$$CW = \left| \left(\frac{Tx - d}{Tx} \times (CW_{\max} - CW_{\min}) \right) + CW_{\min} \right| \quad (2)$$

This scheme ensures that the farthest vehicle within the transmission range of the sender/forwarder will be statistically privileged in becoming the new forwarder. For instance, considering the setting in Figure 11, if G forwards the triggering message advertising a correct BMR of 800 m, then the CWs computed by H, I, J, and K on the basis of Equation 2 will be 776, 528, 280, and 32 slots, respectively. Consequently, K is more likely to become the next forwarder of the game message, and with a high probability, the final forwarder chain will coincide with the aforementioned optimal solution presented in Figure 11, that is, A, C, F, and G.

9. CASE STUDY #2: EXPERIMENTAL EVALUATION

In this section, we evaluate through simulations the interactivity improvement ensured by the FMBP solution discussed in the previous section.

9.1. Simulation assessment

The experiments reported in this section consider a vehicular network with a linear topology and a span of 8 km. Vehicles with communication capabilities are placed, on average, every 20 m, thus, having 400 vehicles that can transmit and forward game events; these vehicles may not all be engaged in the same game session. Note that this does not imply the absence of other vehicles on the road with no communication capabilities or refusing to act as relays for other vehicles' transmissions; indeed, considering the case of a highway with multiple lanes, several vehicle densities are possible.

Among vehicles with communication capabilities, 50 of them are playing among each other. This means that game events are periodically generated in these nodes and broadcast, even through multihop, to all other players in the network. Actually, different sending rates are considered to test how the system behaves in the presence of different kinds of games, from frenetic fast-paced games to slower strategic games, that is, 100 ms, 300 ms, and 500 ms, although the size of each game event is 200 B [67]. The factual transmission range in the experiments varies from 300 m to 1000 m to test extreme values that have been declared by the IEEE 802.11p/DSRC-developing committee [56].

Focusing on FMBP's parameters, we have set the size of a single slot in the CW equal to 100 ms, whereas $CW_{\min}$ and $CW_{\max}$ are equal to 32 and 1024 slots, respectively, as inspired by the standard IEEE 802.11 protocol [68]. In particular, in the latter case, we expect to witness a reduced number of collisions among game events although at the cost of an increased total delivery time.

We have compared our combined FMBP solution with the simple scheme proposed in [58]; the main difference is that the latter does not include the transmission range estimator proposed in [2] and simply assumes to have the transmission range parameters constantly set to a fixed value. Specifically, we name this scheme *Static300* if it considers 300 m as the fixed transmission range parameter and *Static1000* if it uses 1000 m.

Needless to say, Static300 and Static1000 represent the ideal scheme when the factual transmission range is indeed 300 m and 1000 m, respectively. In any other situations, the utilization of a wrong transmission range parameter could result in performance degradation.

Focusing on the chosen metrics, the ability to quickly deliver online game events to players under various conditions is analyzed through the transmission time required by game-related messages to cover the whole vehicular network. In particular, we consider a car platoon of players traveling along a highway and focus on game events belonging to the worst case, those sent by the player within the car leading the car platoon. These messages experience the longest transmissions both in terms of hops and delivery time, as they have to cover the whole car platoon to reach all participants.

9.2. Results

As anticipated, FMBP is compared with Static300 and Static1000. Different generation rates for game events at each player have been considered. Specifically, game events were generated at each vehicle in the gaming car platoon every 100 ms, 300 ms, or 500 ms.

Outcomes for the considered metrics in a scenario with 300 m of factual transmission range are presented in Figure 12. The first property that emerges is represented by the fact that achieved results seem to be independent of the considered message rates. Only at the highest message-sending rate (i.e., 100 ms of interdeparture time), we can observe a little degradation of the performance because of the increased congestion in the network.

The second evident property is that FMBP always achieves better results than the other two schemes, even better than Static300 that is supposed to be the ideal scheme in a scenario with 300 m of transmission range. This is not because of any mistake; rather, even if we have set the transmission range to be 300 m, the adopted

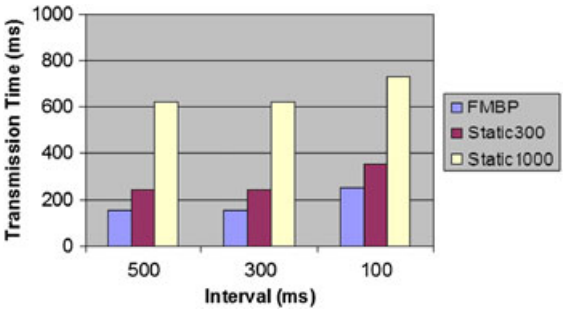

Figure 12. Average transmission time required to cover the whole gaming car platoon, 300 m of factual transmission range.

wireless model realistically generates interferences as it would happen in real world; these interferences make the factual transmission range oscillate around 300 m. Whereas FMBP dynamically adapts to the changing transmission range conditions to maximize its performance, Static300 cannot.

FMBP, Static300, and Static1000 are compared even in a scenario with 1000 m of factual transmission range. The time required to cover the whole vehicular networks is reported in Figure 13. As expected, in this case, Static1000 performs much better than Static300 because with 1000 m of factual transmission range, Static1000 corresponds to the ideal scheme. Yet, without any predetermined knowledge, FMBP succeeds in properly estimating the transmission range for each vehicle and performs as Static1000.

The chart also shows that if the game application generates a message on each vehicle every 100 ms, the total transmission time is considerably higher with respect to the other two considered rates for all the tested schemes. This is clearly because of an excessive increment in the traffic on the wireless channel that causes collisions and time-consuming retransmissions. Indeed, with a wider factual transmission range, more transmitting vehicles can be simultaneously present in the same transmission range area, thus, interfering with each other, increasing congestion and, hence, network delay.

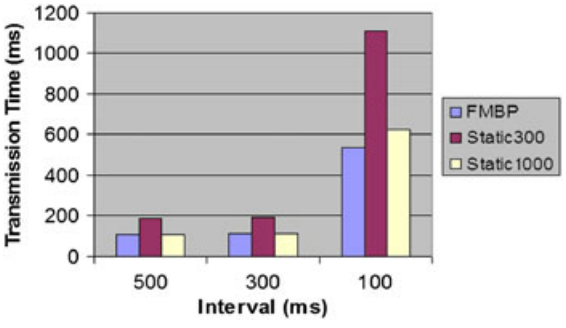

Figure 13. Average transmission time required to cover the whole gaming car platoon, 1000 m of factual transmission range.

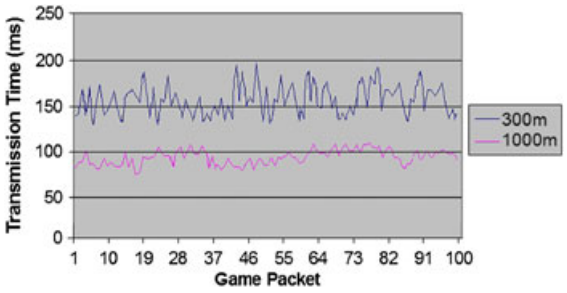

Figure 14. Transmission time trend of messages that have to cover the whole gaming car platoon; FMBP employed 300 ms of message generation interval for each player.

The positive effects of the FMBP combined solution are visible even in Figure 14, which considers 100 successive game event transmissions and, for each of them, reports the transmission time from the first to the last vehicle in the car platoon. The chart considers a packet interleaving time of 300 ms at the source and two possible factual transmission ranges: 300 m and 1000 m. The outcome demonstrates the ability of FMBP in keeping a low transmission time for game events, thus, ensuring a high interactivity degree to mobile players in an infrastructure-less wireless environment. Clearly, the transmission time is affected also by the factual transmission range as the number of transmission hops needed to cover the whole car platoon depends on this parameter.

10. CONCLUSION

Mobile online games are increasing their popularity thanks to the proliferation of personal devices with wireless connectivity. With this vision, we have overviewed the current state of the art in terms of interactive games over mobile networks. We have surveyed main components in ensuring a playful experience to users, discussed network-related metrics for measuring the quality of gaming as perceived by users, and analyzed through experiments the two main communication scenarios supporting a game session: infrastructure-based and infrastructure-less. In particular, we have shown that as combining different solutions taken from the current state of the art, it is possible to significantly improve network support to ensure interactivity even in challenging mobile scenarios.

ACKNOWLEDGEMENTS

The authors of this manuscript wish to express their deepest gratitude to Dr. Nadjib Achir, Dr. Khaled Boussetta, Dr. Stefano Cacciaguerra, Dr. Stefano Ferretti, Dr. Arnaud Kaiser, Dr. Giovanni Pau, and Prof. Marco Roccetti. Part of this manuscript is based on data gathered and/or ideas emerged while collaborating with them at different times. This work has been partially supported by the UniPD Web Squared and MIUR/PRIN ALTER_NET projects.

REFERENCES

1. Cai W, Xavier P, Turner SJ, Lee B. A scalable architecture for supporting interactive games on the Internet, In *Proceedings of the 16th Workshop on Parallel and Distributed Simulation*, Washington, DC, USA, 2002.
2. Palazzi CE, Roccetti M, Ferretti S, Pau G, Gerla M. Online games on wheels: fast game event delivery in vehicular ad-hoc networks, In *Proceedings of 3rd IEEE V2VCOM 2007, IEEE Intelligent Vehicles Symposium 2007*, Istanbul, Turkey, 2007.
3. Pantel L, Wolf LC. On the impact of delay on real-time multiplayer games, In *Proceedings of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Miami, FL, 2002.
4. Armitage G. An experimental estimation of latency sensitivity in multiplayer Quake 3, In *Proceedings of ICON*, Sydney, Australia, 2003; 137–141.
5. Nandan A, Das S, Pau G, Sanadidi MY, Gerla M. Cooperative downloading in vehicular ad hoc wireless networks, In *Proceedings of IEEE/IFIP International Conference on Wireless On demand Network Systems and Services (WONS 2005)*, St. Moritz, Switzerland, 2005.
6. Ferretti S, Palazzi CE, Roccetti M, Pau G, Gerla M. FILA, a holistic approach to massive online gaming: algorithm comparison and performance analysis, In *Proceedings of GDTW 2005*, Liverpool, UK, 2005; 68-76.
7. Xu K, Gerla M, Qi L, Shu Y. TCP unfairness in ad hoc wireless networks and a neighborhood RED solution. *Springer Wireless Networks* 2005; **11**(4): 383–399.
8. Gerla M, Tang K, Bagrodia R. TCP Performance in wireless multi-hop networks, In *Proceedings of the 2nd IEEE Workshop on Mobile Computer Systems and Applications (WMCSA '99)*, Washington, DC, USA; 1999.
9. Xu K, Bae S, Lee S, Gerla M. TCP behavior across multihop wireless networks and the wired Internet, In *Proceedings of the 5th ACM International Workshop on Wireless Mobile Multimedia (WoWMoM 2005)*, Atlanta, GA, USA, 2002.
10. Hamidian A, Palazzi CE, Chong TY, Navarro JM, Körner U, Gerla M. Deployment and evaluation of a wireless mesh network, In *Proceedings of the 2nd IARIA/IEEE International Conference on Advances in Mesh Networks (MESH 2009)*, Athens, Greece, 2009.
11. Rakion. (Online). Available: http://www.rakion.com/.
12. Palazzi CE. Interactive mobile gaming over heterogeneous networks, In *Proceedings of the 5th IEEE/ITI International Conference on Information and Communications Technology (ICICT 2007)*, Cairo, Egypt, 2007.
13. Roccetti M, Marfia G, Amoroso A. An optimal 1D vehicular accident warning algorithm for realistic scenarios, In *Proceedings IEEE Symposium on Computers and Communications (ISCC'10)*, Riccione, Italy; 2010.
14. Marfia G, Roccetti M. Dealing with wireless links in the era of bandwidth demanding wireless home entertainment, In *Proceedings 6th IEEE International Workshop on Networking Issues in Multimedia Entertainment (NIME'10) - 2010 IEEE International Conference on Multimedia and Expo (ICME'10)*, Singapore; 2010.
15. Havinga PJM, Smit GJM. Energy-efficient wireless networking for multimedia applications. *Wiley Wireless Communications and Mobile Computing* 2001; **1**(2): 165–184.
16. Kokkinos P, Papageorgiou C, Varvarigos E. Multi-cost routing for energy and capacity constrained wireless mesh networks. *Wiley Wireless Communications and Mobile Computing* 2011. DOI: 10.1002/wcm.1112
17. Yu C, Lee B, Youn HY. Energy efficient routing protocols for mobile ad hoc networks. *Wiley Wireless Communications and Mobile Computing* 2003; **3**(8): 959–973.
18. Chen L-J, Sun T, Yang G, Gerla M. USHA: a simple and practical seamless vertical handoff solution, In *Proceedings of the 2006 IEEE Consumer Communications and Networking Conference (CCNC 2006)*, Las Vegas, NV, USA, 2006.
19. Palazzi CE, Kaiser A, Achir N, Boussetta K. A preliminary evaluation of backup servers for longer gaming sessions in MANETs, In *Proceedings of the International Workshop on DIstributed SImulation & Online gaming (DISIO 2010) - ICST SIMUTools, 2010*, Torremolinos, Spain, 2010.
20. Zhu K, Niyato D, Wang P, Hossain E, Kim DI. Mobility and handoff management in vehicular networks: a survey. *Wiley Wireless Communications and Mobile Computing* 2011; **11**(4): 459–476.
21. Zhang B, Mouftah HT, Zhao Z, Ma J. Localized power-aware alternate routing for wireless ad hoc networks. *Wiley Wireless Communications and Mobile Computing* 2009; **9**(7): 882–893.
22. Perkins C. *IP Mobility Support for IPv4*, IETF RFC 3344, 2002.
23. Corson S, Macker J. *Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations*, IETF RFC 2501, 1999.
24. Casteigts A, Nayak A, Stojmenovic I. Communication protocols for vehicular ad hoc networks. *Wiley Wireless Communications and Mobile Computing* 2011; **11**(5): 567–582.

25. Leng S, Fu H, Wang Q, Zhang Y. Medium access control in vehicular ad hoc networks. *Wiley Wireless Communications and Mobile Computing* 2009. published online in Wiley InterScience, DOI: 10.1002/wcm.869
26. Nandan A, Das S, Zhou B, Pau G, Gerla M. AdTorrent: digital billboards for vehicular networks, In *Proceedings of IEEE/ACM International Workshop on Vehicle-to-Vehicle Communications (V2VCOM), vol. 1*, San Diego, CA, USA, 2005; 286-294.
27. Heusse M, Rousseau F, Berger-Sabbatel G, Duda A. Performance anomaly of 802.11b, In *Proceedings of IEEE INFOCOM 2003*, San Francisco, CA, USA, 2003.
28. Bottigliengo M, Casetti C, Chiasserini C, Meo M. Short-term fairness for TCP flows in 802.11b WLANs, In *Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, 2004.
29. Garg S, Kappes M. An experimental study of throughput for UDP and VoIP traffic in IEEE 802.11b networks, In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC 2003)*, New Orleans, LA, USA, 2003; 1748–1753.
30. Wijesinha AL, Song Y, Krishnan M, Mathur V, Ahn J, Shyamasundar V. Throughput measurement for UDP traffic in an IEEE 802.11g WLAN, In *Proceedings of 6th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)*, Towson, MD, USA; 2005.
31. Bianchi G. IEEE 802.11-saturation throughput analysis. *IEEE Communications Letters* 1998; **2**(12): 318–320.
32. Bianchi G. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications* 2000; **18**(3): 535–547.
33. Beigbeder T, Coughlan R, Lusher C, Plunkett J, Agu E, Claypool M. The effects of loss and latency on user performance in unreal tournament 2003, In *Proceedings of ACM Network and System Support for Games Workshop (NetGames 2004)*, Portland, OG, USA, 2004; 144–151.
34. Palazzi CE, Pau G, Roccetti M, Gerla M. In-home online entertainment: analyzing the impact of the wireless MAC-transport protocols interference, In *IEEE WIRELESSCOM2005*, Maui, HI, USA, 2005.
35. Hole DP, Tobagi FA. Capacity of an IEEE 802.11b wireless LAN supporting VoIP, In *Proceedings IEEE International Conference on Communications (ICC 2004)*, Paris, France, 2004.
36. Nam CH, Soung CL, Fu CP. An experimental study of ARQ protocol in 802.11b wireless LAN, In *Proceedings of IEEE Vehicular Technology Conference (VTC 2003)*, Orlando, FL, USA; 2003.
37. Xylomenos G, Polyzos GC. TCP and UDP performance over a wireless LAN, In *Proceedings of IEEE INFOCOM '99*, New York, NY, USA; 1999.
38. Palazzi CE, Ferretti S, Roccetti M, Pau G, Gerla M. What's in that magic box? the home entertainment center's special protocol potion, revealed. *IEEE Transactions on Consumer Electronics* 2006; **52**(4): 1280–1288.
39. Butterfly grid solution for online games. http://www.butterfly.net.
40. Quake Forge Project. http://www.quakeforge.org/.
41. Everquest. http://www.everquest.com.
42. Ultima online. http://www.uo.com.
43. Gautier L, Diot C. Design and evaluation of MiMaze, a multi-player game on the Internet, In *Proceedings of IEEE International Conference on Multimedia Computing and Systems (ICMCS'98)*, Austin, TX, USA, 1998.
44. Safaei F, Boustead P, Nguyen CD, Brun, J, Dowlatshahi M. Latency driven distribution: infrastructure needs of participatory entertainment applications, In *IEEE Communications Magazine, Special Issue on Entertainment Everywhere: System and Networking Issues in Emerging Network-Centric Entertainments Systems, Part I*, 2005.
45. Palazzi CE, Ferretti S, Cacciaguerra S, Roccetti M. Interactivity-loss avoidance in event delivery synchronization for mirrored game architectures. *IEEE Transactions on Multimedia* 2006; **8**(4): 847–879.
46. Jehaes T, De Vleeschauwer D, Coppens T, Van Doorselaer B, Deckers E, Naudts W, Spruyt K, Smets R. Access network delay in networked games, In *Proceedings of ACM NetGames 2003*, Redwood City, CA, USA, 2003.
47. Palazzi CE, Ferretti S, Cacciaguerra S, Roccetti M. A RIO-like technique for interactivity loss avoidance in fast-paced multiplayer online games. *ACM Computers in Entertainment* 2005; **3**(2).
48. Palazzi CE, Maggiorini D. From playgrounds to smartphones: mobile evolution of a kids game, In *Proceedings of the 8th IEEE Communications and Networking Conference (CCNC 2011)*, Las Vegas, NV, USA, 2011.
49. Cronin E, Kurc AR, Filstrup B, Jamin S. An efficient synchronization mechanism for mirrored game architectures. *Multimedia Tools and Applications* 2004; **23**(1): 7–30.
50. Yu C, Lee B, Youn HY. Energy efficient routing protocols for mobile ad hoc networks. *Wiley Wireless*

Communications and Mobile Computing 2003; **3**(8): 959–973.
51. Yahya B, Ben-Othman J. Towards a classification of energy aware MAC protocols for wireless sensor networks. *Wiley Wireless Communications and Mobile Computing* 2009; **9**(12): 1572–1607.
52. Corradini F, Gorrieri R, Roccetti M. Performance preorder: ordering processes with respect to speed. In *Proceedings of Mathematical Foundations of Computer Science*, LNCS 969. Springer: Berlin, 1995; 444–453.
53. Palazzi CE, Roccetti M, Ferretti S. An inter-vehicular communication architecture for safety and entertainment. *IEEE Transactions on Intelligent Transportation Systems* 2010; **11**(1): 90–99.
54. Aoki M, Fujii H. Inter-vehicle communication: technical issues on vehicle control application. *IEEE Communications Magazine* 1996; **34**(10): 90–93.
55. Biswas S, Tatchikou R, Dion F. Vehicle -to-vehicle wireless communication protocols for enhancing highway traffic safety. *IEEE Communication Magazine* 2006; **44**(1): 74–82.
56. ASTME 2213-03, *Standard Specification for Telecommunications and Information Exchange Between Road-side and Vehicle Systems 5.9GHz Band Dedicated Short-Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ASTM International, 2003.
57. Yang X, Liu J, Zhao F, Vaidya N. A Vehicle-to-vehicle communication protocol for cooperative collision warning, In *Proceedings of 1st Annual International Conf. on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04)*, Boston, MA, USA, 2004.
58. Fasolo E, Furiato R, Zanella A. Smart broadcast algorithm fo inter-vehicular communication, In *Proceedings of Wireless Personal Multimedia Communication (WPMC'05), IWS 2005*, Aalborg, DK, 2005.
59. Korkmax G, Ekici E, Ozguner F, Ozguner U. Urban multi-hop bradcast protocol for inter-vehicle communication sustems, In *Proceedings of the 1st ACM Workshop on Vehicular Ad-hoc Networks (VANET 2004)*, Philadelphia, PA, USA, 2004.
60. Zanella A, Pierobon G, Merlin S. On the limiting performance of broadcast algorithms over unidimensional ad-hoc radio networks, In *Proceedings of WPMC04*, Abano Terme, Italy, 2004.
61. Wan PJ, Alzoubi K, Frieder O. Distributed construction of connected dominating set in wireless ad hoc networks, In *Proceedings of IEEE INFOCOM 2002*, New York, NY, USA; 2002.
62. Bononi L, Di Felice M. A cross-layered MAC and clustering scheme for efficient broadcast in VANETs, In *Proceedings of 2-th IEEE International Workshop on Mobile Vehicular Networks (MOVENET 2006)*, Pisa, Italy, 2007.
63. IOQuake 3, Ioquake 3 open source project, http://ioquake3.org/.
64. Park K, Willinger W. *Self-similar Network Traffic and Performance Evaluation*, 1st Edition. Wiley-Interscience: New York (NY), USA, 2000.
65. The Network Simulator, NS-2. [Online]. Available: http://www.isi.edu/nsnam/ns/.
66. Movie Trace Files. [Online]. Available: http://www-tkn.ee.tu-berlin.de/research/trace/ltvt.html.
67. Färber J. Network game traffic modelling, In *Proceedings of ACM NetGames2002*, Braunschweig, Germany, 2002.
68. IEEE, *Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (Phy) Specifications*, Specifications, ISO/IEC, 8802-11:1999(E), 1999.

AUTHORS' BIOGRAPHIES

Mario Gerla is a professor in the Computer Science Department at the University of California, Los Angeles (UCLA). He holds an Engineering degree from Politecnico di Milano, Italy and a PhD degree from UCLA. He became IEEE fellow in 2002. At UCLA, he was part of the team that developed the early ARPANET protocols under the guidance of Professor Leonard Kleinrock. He joined the UCLA faculty in 1976. At UCLA, he has designed network protocols including ad hoc wireless clustering, multicast (ODMRP and CODECast), and Internet transport (TCP Westwood). He has lead the ONR MINUTEMAN project, designing the next generation scalable airborne Internet for tactical and homeland defense scenarios. He is now leading two advanced wireless network projects under ARMY and IBM funding. His team is developing a Vehicular Testbed for safe navigation, content distribution, urban sensing, and intelligent transport. Vehicular content distribution protocols such as Car Torrent and Code Torrent are being tested, as well as urban surveillance techniques based on epidemic P2P dissemination. A parallel research activity explores smartphone centric wireless medical monitoring.

Dario Maggiorini is an assistant professor at the University of Milan, Italy, where he received his master degree and PhD in computer science in 1997 and 2002, respectively. He joined as a faculty member of the Department of Informatics and Communication in 2003, where his

teaching activity is typically related to operating systems and network protocols and architectures. In the past, he has been working on Quality of Service for IP-based networks, multimedia content delivery, application-level networking, and software architectures for service provisioning. Currently, his research interests focus mainly on software and network architecture for entertainment applications and content/service provisioning in distributed environments.

Claudio Enrico Palazzi is an associate professor of the Department of Pure and Applied Mathematics, University of Padua, Italy. He was the first student enrolled in the joint PhD program in Computer Science organized by the University of Bologna and UCLA. Through this program, he received his MS degree in Computer Science from UCLA in 2005, his PhD degree in Computer Science from University of Bologna in 2006, and his PhD degree in Computer Science from UCLA in 2007. From 2007 to 2010, he was an assistant professor at the Department of Pure and Applied Mathematics of the University of Padua. His research is primarily focused on protocol design and analysis for wired/wireless networks, with emphasis on network-centric multimedia entertainment and vehicular networks. On these topics, he is activein various technical program committees in prominent international conferences and is a co-author of more than 80 papers, published in international conference proceedings, books, and journals.

Armir Bujari is a PhD student in Computer Science at the University of Padua, Italy and is completing the first year of his program under the supervision of Professor Claudio E. Palazzi. Armir's research is focused on wireless networking and delay-tolerant communication for mobile networks. Armir previously completed a Master's degree in Computer Science, Summa Cum Laude, at the University of Padua.