Rear-seat Multiplayer Gaming using Peer-to-Peer Car Communication Networks

Piyao Liu

A dissertation submitted to the University of Dublin, Trinity College

in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

September 2007

DECLARATION

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Name: Piyao Liu

Date: 13th September, 2007

PERMISSION TO LEND AND/OR COPY

I agree that Trinity College Library may lend or copy this dissertation upon request.

Name: Piyao Liu

Date: 13th September, 2007

ACKNOWLEDGEMENTS

I would especially like to thank my supervisor, Professor Vinny Cahill, for his enthusiasm, support and guidance throughout the course of this project.

I would also like to thank Shane Brennan, Anthony Harrington and Stefan Weber for their help of supplying me with the huge amount of hardware I needed. Special thanks to Shane Brennan for the help of providing helpful suggestion during the project.

I would like to take the opportunity to thank Erich Barnstedt, Ian Toal and Sébastien Molines for their help and insights throughout the project.

Finally, I would like to thank my father and elder brother for their perpetual and unconditional support.

ABSTRACT

The inter-vehicle communication has been becoming a new favour of research area during these years. A lot of algorithms and protocols for vehicular ad hoc networks (VANETs) are proposed, mainly focusing on the on the traffic information propagation among vehicles but paying little attention on the prediction of connection duration and issues of latency.

Multi-player vehicular games require relatively long connection duration and low latency. We design and implement a game platform, which predicts the connection duration and the signal strength, running over Windows CE operating system in a VANET by studying the requirements of online games and the scenarios of vehicular communication for multi-player games. The game platform consists of a reactive ad hoc routing protocol, Ad Hoc On-demand Distance Vector routing protocol (AODV) [1], which is fixed and improved from a semi-finished implementation discussed in [11], and the Simple Game platform (SGP), which is built on top of .Net Compact Framework [2] [3] [4] and Garmin GPS system [6] in a peer-to-peer structure, fulfills the functionality of registering, launching, updating game services and predicting the connection duration and signal strength by using the Connection Duration Prediction (CDP) algorithm.

The whole system has been successfully deployed on PDAs. The tests of the game platform measure the performance of AODV routing protocol and SGP. Evaluation of the test shows the amelioration of AODV routing protocol before it deployed for VANETs is recommended because of its neglects of the physical position of vehicles and routing strategies.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

1 Introduction

This dissertation presents a design, implementation and evaluation of the rear-set multiplayer gaming using peer-to-peer car communication networks.

While the ubiquitous technology exploits in gadgets, the inter-vehicle communication remains a conservative technological island. The requirement of traffic information around one specific vehicle becomes significance, especially in foggy day and when a vehicle is around the corner. The adoption of wireless communication technology to support computer-based inter-vehicle communication can make the exchange of data fast. Based on the inter-vehicle communication protocols and vehicular operating systems, we can implement other software to extend the functionalities of inter-vehicular networks, such as: games, multimedia services, email services, location services, and online shopping, etc. Unlike devices in other wireless networks, vehicles in vehicular networks tend to move in an organized fashion. The interactions with roadside equipment can likewise be characterized fairly accurately. Further more, most vehicles are restrained in their range of motion, for example by being constrained to follow a paved highway. In this dissertation project we design and implement a game platform running over Windows CE operating system based on a reactive ad hoc routing protocol, AODV routing protocol, .Net Compact Framework and Garmin GPS system by adopting the peer-to-peer structure.

The AODV routing protocol is a routing protocol to discover route across wireless mesh network. It is capable of both unicast and multicast routing. AODV offers quick adaptation to dynamic changing network topology with low processing and memory head. It is a reactive routing protocol, meaning that it establishes the route to a destination only on demand.

The networking part of the present operating systems are designed with the assumption of static networks. New features of wireless Mobile Ad hoc Networks (MANETs) are not considered, such as dynamically changing topologies of wireless networks and unstable signal strengths of wireless network connections. Windows CE operating system provides the Network Device Interface Specification (NDIS) which governs the communication between interface device drivers controlling hardware adapters, and the upper-level protocols [2]. The AODV routing protocol for Windows CE in this project is a fixed version from a semi-finished implementation described in [11], implemented as an intermediate driver which is specified in NDIS as a driver facilitates communication between the operating system and upper level protocol drivers.

The network components of online games at present time are mostly developed with the assumption of wired networks. The networking architectures of these games are mainly based on the Client-and-Server (C/S) structure, which provides unified and effective registration and access control mechanism. The server (or a set of servers) of a C/S structured multi-player games is supposed to be in a logically fixed location with a fixed IP address (or a set of IP addresses). These features indicate that the C/S structure is not suitable for the wireless ad hoc networks. When implementing applications that running over wireless networks with C/S structure, disadvantages restrain both of the performance and efficiency, especially when the topology of the networks changes dynamically and frequently, and the nodes inside the network are distributed in peer-to-peer pattern. The distributed peer-to-peer pattern of the wireless ad hoc networks shows that Peer-to-Peer (P2P) structure [5] is a better choice for MANETs. Peer-to-peer networks are typically used for connecting nodes in ad hoc connections for the purposes of sharing content files, audio, video, chatting and gaming information exchange, etc.

Simple Game Platform (SGP) is platform running on the Windows CE operating system aims to advertise, discover and launch the games information from an Xbox

which is connected to the Windows CE operating system via wired network. It is built on top of the .Net Compact Framework, imports the information of GPS and routes from Garmin Mobile XT API [7] and the wireless network status information from the OpenNETCF.Net. By analyzing the information of GPS and routes and wireless network status information, SGP predicts the connection duration of multi-player games in a smart way.

1.1 Dissertation Roadmap

This is a brief description of the remainder of this document, giving the overall layout of this dissertation.

Chapter 2 describes various technologies and solutions that have the functionality to share and propagate information in either wireless mobile or wireless vehicular ad hoc networks. The AODV routing protocol is also detailed in this chapter.

Chapter 3 gives an overview of the framework of the game platform. The requirement of online games is analyzed for vehicular networks. It describes the possible protocols (such as 3G and 802.11) and the scenarios for vehicular communication and the architectures of online games that are suitable for the game platform. Based on the discussion above, a proposed framework of the game platform is presented.

Chapter 4 gives the details of the implementation, including the plugs-in used by the system, such as Garmin system and OpenNETCF.Net. The game platform is implemented in two separate parts, the AODV implementation written in C running in the kernel of windows CE and SGP written in C# and C running as a user application in windows CE.

Chapter 5 evaluates the game platform including the AODV routing protocol and the SGP by using three PDAs along with three same IEEE 802.11b wireless cards and one

laptop and presents conclusions concerning the further development.

Chapter 6 concludes the dissertation, outlines the possible improvements and discusses the future work.

2 State of Art

2.1 Inter-vehicle Communication

Ubiquitous computing has been becoming a hot topic in recent research, distributed algorithms and protocols which satisfy specific requirements of ubiquitous computing receives a lot of attention, to achieve the aims of ubiquitous computing, some pre-work are presented, one of them is the wireless Mobile Ad Hoc Networks (MANETs), which gives a quite attractive research area because of its high flexibility, ad hoc infrastructure and its challenges such as inherently-highly-dynamic network topology and limited transmission range [20] to researchers.

As a specific area of MANETs, wireless Vehicular Ad Hoc Networks (VANETs) become a new favor among researchers because of its great market potential [20] [21]. Compared to MANETs, which has the characteristics of short range transmission, low bandwidth, omnidirectional propagation and low storage capacity, unique characteristics of VANETs contains the followings [20]:

- ***Rapidly-changing topology among vehicles***: vehicles run in different speed along roads, so the relative speeds of the vehicles are merely zero, and so vehicles change it relative position to others, a wireless connection is effective when the distance of two vehicles is less than the theoretical effective distance of the wireless transmission. For example, when the relative speed of two leaving vehicles is 10m/s and the theoretical effective distance is 80 meter, then the connection duration is at most 8 seconds.

- ***Unstable connection of wireless networks***: when the topology among vehicles changes fast, the quality of signal strength changes correspondently, vehicles in VANETs moves much faster than the normal nodes in MANETs, when the relative speed and distance among vehicles becomes larger,

connections becomes unstable even forces to break.

- ***Vehicles and routes information is a useful resource for applications***: vehicles normally run along pre-built roads, by analyzing vehicles and routes information, vehicles can exchange data in a predictable time interval efficiently; the position function related to time of vehicles can also be predicted.

- ***Propagated data are supposed to be compressed and aggregated***: the low bandwidth of wireless networks demands data be compressed before it send to others. Besides, data transmission to other nodes is usually broadcasted, when the number of hop count becomes larger and data packets increase, it consumes most of the bandwidth to further information to other vehicles.

2.2 Related Work

In this section, applications and algorithms that aim to solve problems encountered in MANET especially in VANET are discussed.

2.2.1 TrafficView

TrafficView [20] [22] is a scalable traffic monitoring system embedded in vehicles to form vehicular ad hoc networks to exchanging traffic information in a real-time way. Each TrafficView system includes three components [20] [22]: an embedded computer with display, a GPS receiver, and a short-range wireless network interface (IEEE 802.11b). Information of location, speed, current time and direction is provided by the GPS receiver while information of other vehicles around one specific vehicle is gathered and broadcasted in a peer-to-peer fashion using the short-range wireless interface. Dynamic and real-time information exchanged over the vehicular ad hoc network is showed in a map of the road in the display with annotations.

Figure 2-1 [20] [22] illustrates the software architecture of TrafficView. Information received through a broadcast message is first stored in the Non-Validated dataset considering outdated or conflicting information. After examined for validity, they are merged into the validated dataset. The system displays data from validated dataset periodically through navigation module. Aggregated data is broadcasted periodically by each vehicle.

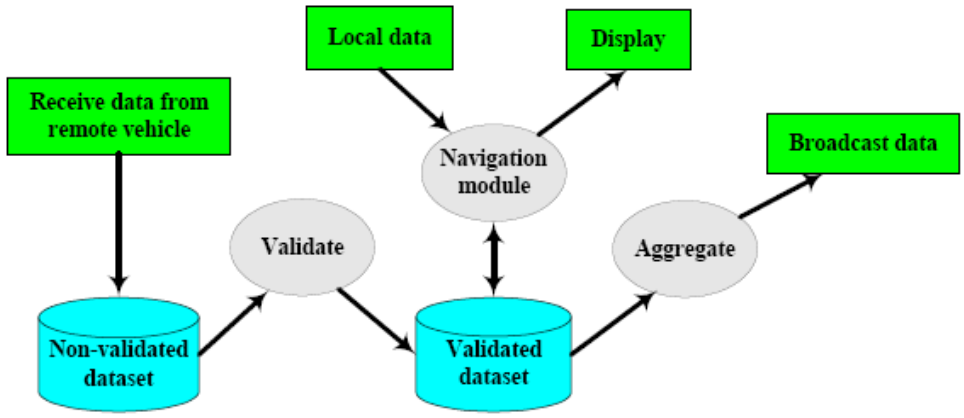

Figure 2-1: TrafficView Software Architecture [20]

TrafficView broadcast all data stored in a vehicle in a single packet for the purpose of consuming less bandwidth, reducing the number of re-transmissions due to collisions and avoiding with dealing with flow control. Semantic data aggregation was applied to this system for the purpose of reducing the size of packets which were broadcast periodically. Two data aggregation algorithms are discussed and evaluated in [20], one is the ratio-based algorithm, and another is the cost-based aggregation.

The System was implemented in Java over Linux by using HP iPAQs equipped with 802.11 cards for wireless communications.

2.2.2 EZCab

EZCab [22] is a ubiquitous computing application allowed people to book nearby cabs using their cell phone or PDAs equipped with short-range wireless network

interfaces. The EZCab works in a decentralized fashion rather than traditional centralized dispatching solution, which is not scalable due to [22]: 1) dispatching all requests through one or multiple cab dispatcher introduces waiting time for clients, especially during period of peak cab request, and 2) all cabs in the city should be monitored in order to dispatch the nearest cab to the client.

EZCab includes two components: 1) a MANET of computers embedded in taxis and 2) clients handheld devices, which communicate using short-range wireless network interfaces such as IEEE 802.11b. EZCab consists of two types of entities [22] client stations, which is PDA or cell phone, and driver stations which are embedded in cabs.

The system was implemented on top of the Smart Messages (SM), which define middleware architecture for application development over MANET. The self-routing mechanisms of SM enables newly developed routing protocols plug into the middleware. Three routing algorithms, *Flooding*, *Probabilistic On-Demand*, and *Probabilistic Proactive* were implemented and evaluated in [22].

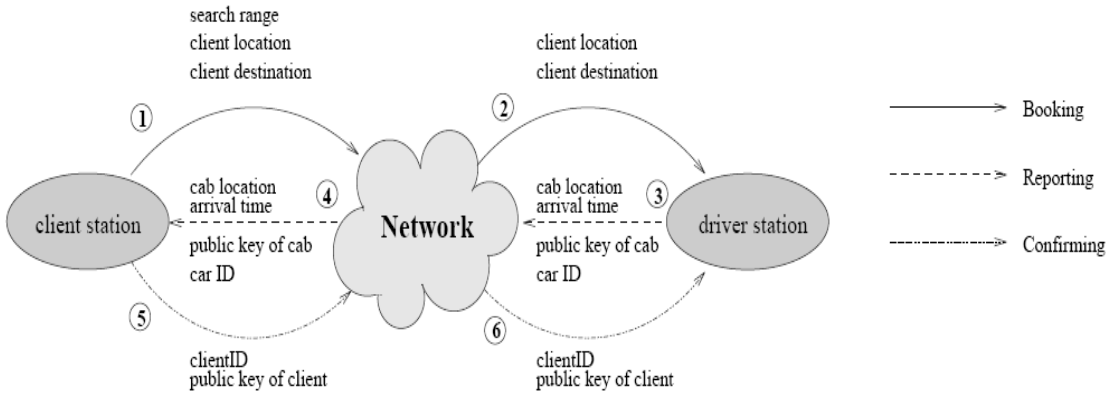

Figure 2-2: EZCab booking protocol [22]

Figure 2-2 shows the EZCab protocol for the cab booking phrase. During these phrases, the request of one specific request is forward by other clients through the network until a free cab is discovered.

The System was installed on HP iPAQs running Linux. The iPAQs use Orinoco's 802.11 cards for wireless communication, and each of them is connected to a Geko 201 GPS receiver [23].

2.2.3 A special-purpose peer-to-peer file sharing system for MANET

A special-purpose peer-to-peer file sharing system for MANET was implemented and evaluated in [24]. The system was tailored to both the characteristics of MANET and the requirement of peer-to-peer file sharing.

A special-purpose approach named Optimized Routing Independent Overlay Network (ORION) was designed for this system due to a lack of fixed infrastructure and no a-priori knowledge of arriving and departing peers. ORION consists of an algorithm for construction and maintenance of an application-layer overlay network that enables routing of all types of messages required to operate a P2P file system, i.e., queries, responses and file transmissions [24]. ORION provides an efficient algorithm by combining with technologies known from the AODV routing protocol [1] and the Simple Multicast and Broadcast Protocol for MANET [25].

Figure 2-3, 2-4, 2-5, 2-6 show one four-node scenario to illustrate the operation of the IRION search algorithm.

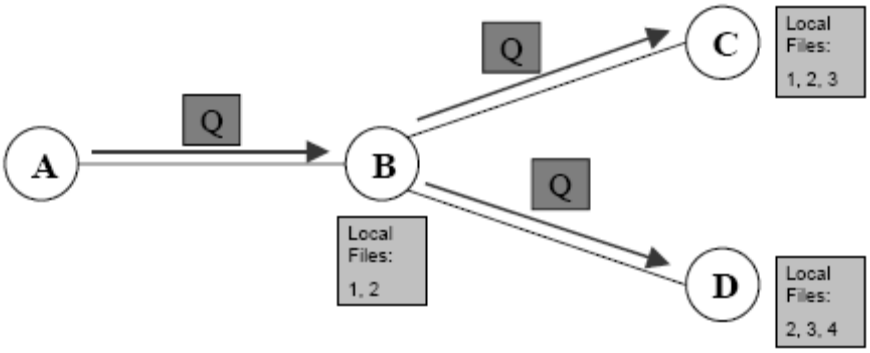

Figure 2-3: Operation of IRION 1-Node A floods a QUERY message with keywords matching to files 1 to 4 [24]

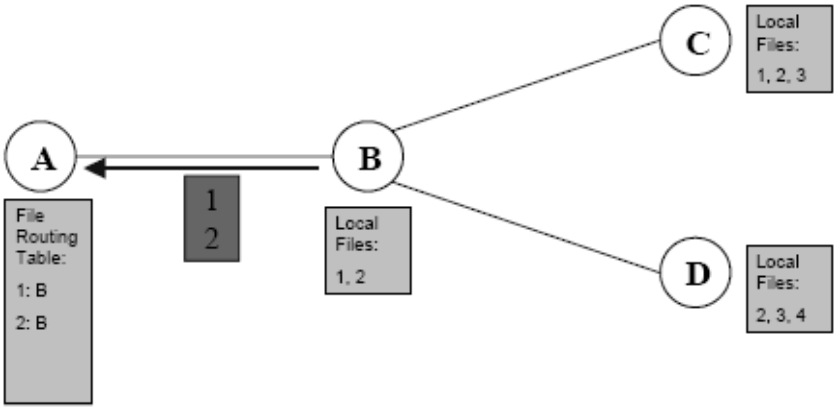

Figure 2-4: Operation of IRION 2-Node B sends a RESPONSE message with identifiers of local files [24]

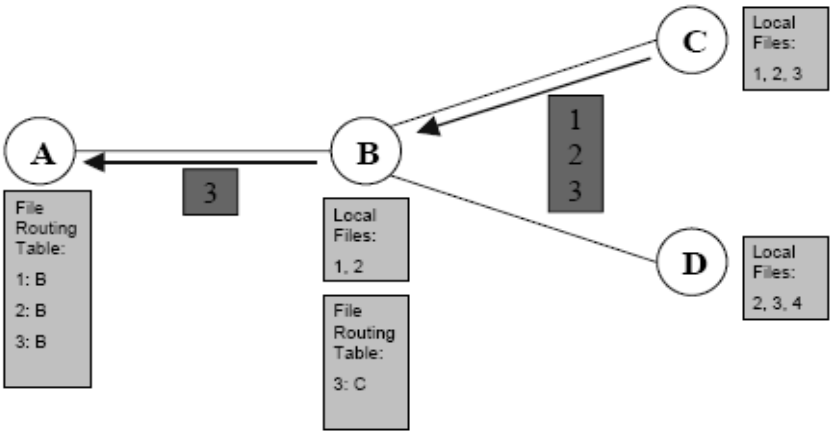

Figure 2-5: Operation of IRION 3-Node C sends a RESPONSE message with identifiers of local files, Node B filters and forwards RESPONSE of C [24]

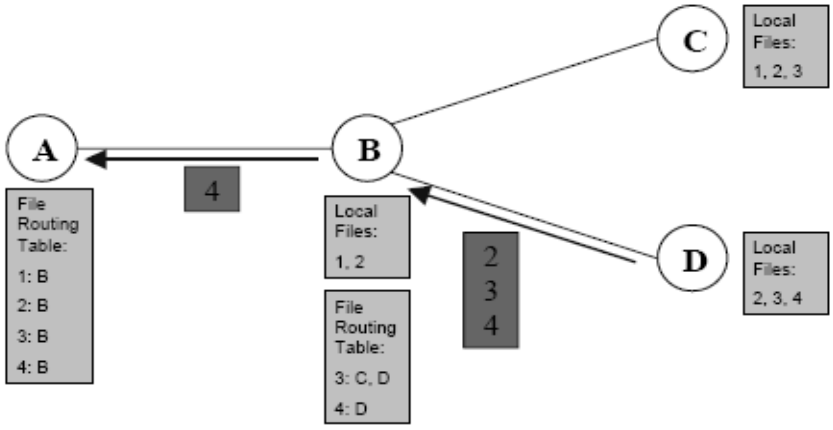

Figure 2-6: Operation of IRION 4-Node D sends a RESPONSE message with identifiers of local files; Node B filters and forwards the RESPONSE message [24]

The simulation of the performance of both the ORION searching algorithm and the ORION transfer protocol shows that ORION transfer protocol has better performance in message volume, successful transfer ratio and overhead than the off-the-shelf approaches.

In [27], Tests based on the hypothesis that the motion of vehicles on a highway can contribute to successful message delivery are carried out. The simulation environment consists of two components [27], a traffic micro-simulator for vehicular traces generation and a network simulator for transmission of messages. The results of simulation show that when the density (cars per km) of vehicles is less than 60, the delay can be ignored. In [28], a Speed Dependent Random Protocol (SDRP) for road information dissemination is proposed and evaluated. In [29], a Received Message Dependent Protocol (RMDP), which disseminates and propagates in dynamical intervals depending on the number of reception messages, is proposed.

The related work discussed above, mostly concerns the algorithms of dissemination of traffic information, based on the algorithms, a lot of protocols are proposed. One of the important characteristics of dissemination of traffic information is that, the foreseeing connection duration among vehicles is not important, in contrast,

aggregation of data is required to reduce the amount of data packets. The ORION algorithm for a peer-to-peer file sharing system, shows a good performance, but is not suitable for the game platform to discovery players, which requires the predication of connection duration among vehicles.

When player information is found in a wireless ad hoc network, a relatively mature ad hoc routing protocol is required to maintain the routes information when games are under play. In the remainder of this chapter, some of the ad hoc routing protocols are described. A semi-finished AODV routing protocol for Windows CE operating system is discuss in [11], by fixing and improving the performance of the semi-finished AODV routing protocol implementation, games can play in good performance.

2.3 Ad Hoc Routing protocol

A mobile ad hoc network is a self-configuring network in which mobile nodes are connected by wireless links. Compared to convectional networks, MANETs exhibit the following properties:

- The topology of the wireless network may change randomly in an unpredictable way since wireless nodes in a MANET are free to move arbitrarily. As such links in a wireless network are changing, breaking and remade frequently.

- Given the interference from other wireless node and multi-path fading problems, links in a wireless network may suffer from radio transmission propagation effects.

- It may not feasible to maintain large amount of routing information while the scalability of an ad hoc network expanded and the topology of the wireless network changed dynamically.

The existing routing protocols such as AODV, Dynamic Source Routing (DSR) [8] [9], Temporally Ordered Routing Algorithm (TORA) [8] [10] and Associativity Based Routing (ABR) [8] [12], etc. for wireless networks are designed to overcome the above problems.

There are three types of ad hoc routing protocols: *proactive*, also known as table-driven routing protocol, which updates their routes to destination by periodically distributing routing table in the network, reactive, also known as on-demand routing protocol, which finds their routes to destination by flooding routing request packets in the network only on-demand and *hybrid*, which tries to combine the advantages of both proactive and reactive routing protocols.

In the remainder of this section, the AODV routing protocol is explained in detail as it was selected for our project.

2.3.1 AODV Routing Protocol

AODV routing protocol is a routing algorithm to discover route across wireless mesh network. It is capable of both unicast and multicast routing. In this section, AODV of unicast is described. AODV offers quick adaptation to dynamic changing network topology, low processing and memory head. It is a reactive routing protocol, meaning that it establishes the route to a destination only on demand.

Reactive protocols have the advantage of being more scalable than table-driven protocols [2]. They only maintain the routes in use. The disadvantage of this method is that the delay in route finding when a node wishes to communicate with a new node for which there is no entry in the route table. The DSR and AODV routing protocol are examples of on-demand protocols.

A number of routing messages are defined for the AODV routing protocol, including Route Requests (RREQs), Route Replies (RREPs), Route Errors (RERRs) and the Hello message. These messages are communicated via UDP, and aim to discover destinations and maintain routes among nodes.

RREQs Messages: The AODV does not play any role when there is a route to the destination. When a route is needed to a new destination, the node broadcasts a RREQ message to find the new destination. This RREQ message contains several import fields: the originator IP Address, the destination IP Address, the originator sequence number, the destination sequence number and the hop count which identifies the lifespan of the message.

The RREQ message can be received by the originator and the neighbors. When the RREQ Message is received by the neighbors, the neighbors either rebroadcast the message if they do not have a valid route to the destination or send a RREP message back to the originator if they have a valid route to the destination or it is the destination itself.

The hop count is decremented by one at each hop and the message is discarded when the number of hop count reaches zero.

RREPs Messages: When a node receives a RREQ message, it will either discard the RREQ message silently if it has received a RREQ message with the same originator IP Address and RREQ ID within at least the last path discovery time or unicast a RREP message back to the node from which it received the RREQ message if it contains an up-to-date route to the destination or it is the destination itself. Once a RREP message is created, it is unicasted to the next hop toward the originator of the RREQ. The RREP message may be propagated along the reverse path of the RREQ message which is originated from the originator or via a new one.

If the node generating the RREP message is not the destination itself but an intermediate node which has a valid route to the destination, the intermediate node also sends a *gratuitous* RREP message to the destination along the route it knows.

HELLO Messages: The HELLO messages are broadcasted periodically when a node wants to detect the connectivity of its neighbors. A HELLO message is simply a RREP message with TTL=1, meaning that the message will not be propagated by its neighbors. A node checks its neighbors by listening for the periodical HELLO messages. A node detects a broken link by the absence of a HELLO message after a HELLO message interval, meaning that the nodes can no longer directly communicate, and a RERR message may be generated and propagated.

RERRs Messages: A RERR message may be either broadcast or unicast from a node to its neighbors when a link to the next hop can not be detected by a node. When a node receives a RERR message, it removes all the routes from its routing table that contains the invalid next hop.

Figure 2-7 shows a scenario when node B wants to find a route to node N. In Figure 2-7-a shows the potential connections among neighboring nodes established by wireless short range interfaces. At first, node B floods a RREQ message to find the destination N to its neighbors, A and C shown in Figure 2-7-b, node A and C rebroadcast the RREQ message to it neighbors, hosts receiving the RREQ message set up a reverse route to B shown in Figure 2-7-c, this continues until the RREQ message reaches the destination N, shown in Figure 2-7-d, instead of rebroadcasting the received RREQ message, N generates a RREP message and may send it alone the reverse path, when a route is finally established from node B to N, N can start to send data packets to N.

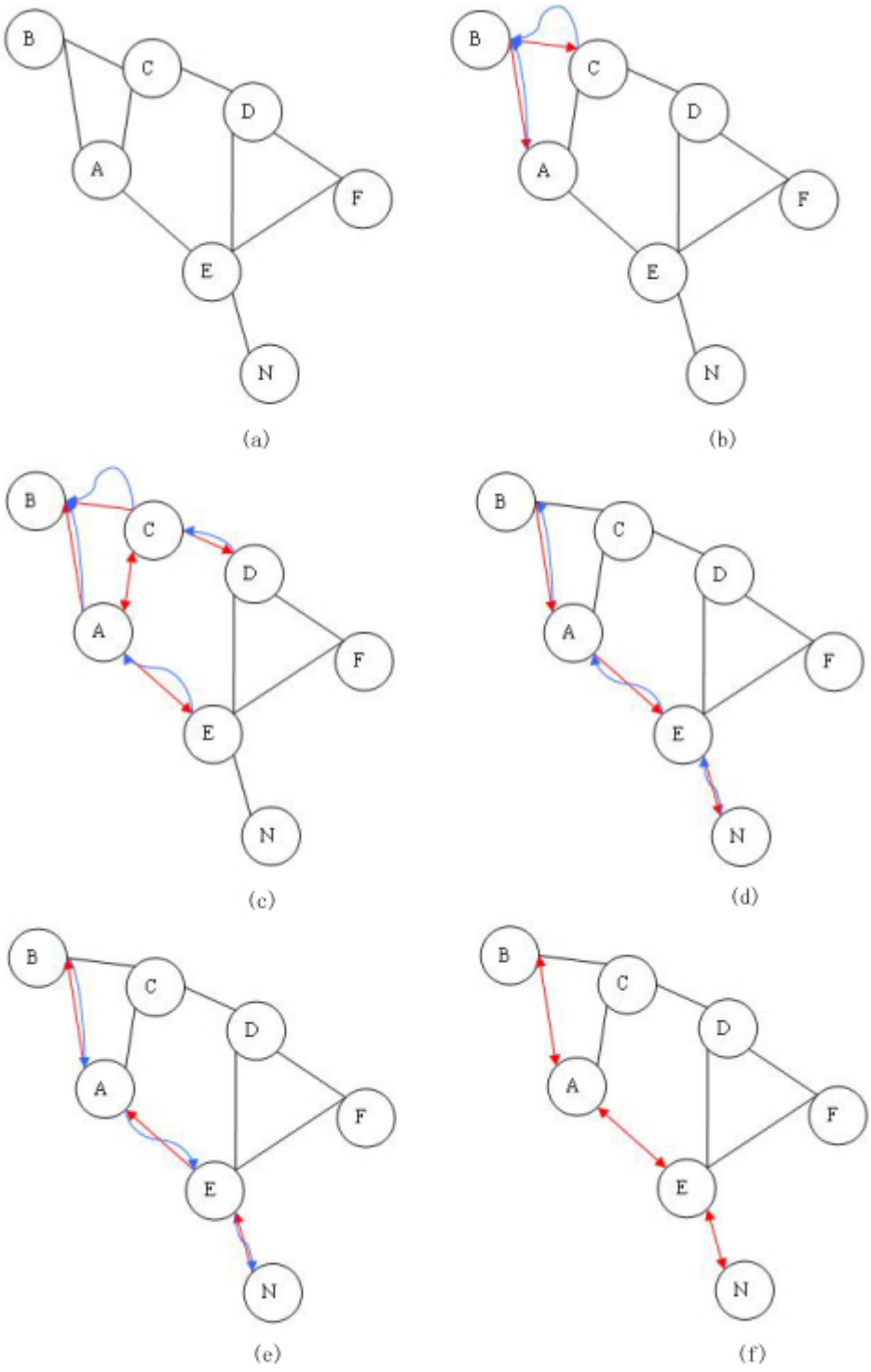

Figure 2-7: routing discovery of AODV routing protocol

3 Design

3.1 Study of Online Games

Online games, starting in the 1980s with Multi-User Dungeon, Domain or Dimension (MUDs), are games played over networks, commonly the Internet. The range of online games can be from simple text based games to games with vivid display effects and realistic virtual worlds populated by many players simultaneously. As games become more complex, exchange of data increases among players. The existence of relatively high latency in the Internet is a manifest factor that affects the performance of online games.

Multi-player games have become popular for their interaction among humans and the increasingly powerful personal computers. As the proliferation of multi-player games, millions of players get involved in one single game simultaneously. Such a large number of players in the same time require servers of the online game of high throughput and fast processing ability. Many studies have worked on user tolerance for high latency and the effects of latency on user performance in real-time strategy games, as well as sports and first person shooter games.

The large scale and centralized management of online games demand that these games organized in a Client-Server (C/S) structure. Subscribers of one online game get access to the same server (or a set of servers) with other subscribers, who get the game services from the online game supporting provider by paying a few amounts of fees. Compared to the C/S structured online games, the Peer-to-peer structured games provide another useful method for small scale and decentralized management for players. In the games using C/S structure, the servers are responsible for resolving any state inconsistencies and updates of players, while in the games using P2P structures, updates occur during the communications among each players.

The reminder of section 3.1 is divided into four sections. In section 3.1.1, the typical categories highly accepted by the players of online games are introduced. Section 3.1.2 presents details of the most two popular architectures widely used by present online games, based on the two architectures, a hybrid structures for online games is proposed. The most important features of online games, bandwidth requirements and state consistency are analyzed and the theoretical formulas and empirical data of bandwidth requirements and state consistency online games are detailed in section 3.1.3. New features of multi-player vehicular games are described based on the discussion above in section 3.1.4.

3.1.1 Categories of Online Games

The high interaction among humans of online games attracts more and more players devote their spare time for the fun with other players around the world. The amount of online games is increasing exponentially. At the moment, many game companies are diverting to the development of online games. As the number of players becoming larger, their age ranges from teenager to senior; all kinds of games are required and developed for their different requirements. But, typically these online games can be categorized as follows based on the popularity among players:

Shooter Games: First Person Shooter (FPS) games are the most popular shooter games at present, other shooter games such as on-rails shooter [13], are viewed from a first-person perspective, while flight simulators frequently involve the use of weapons; they are not considered as first person shooter games. Players in first person shooter games are usually formed into teams, which fight the other groups during the games. Counter Strike, Doom and Far Cry, etc. are the most popular first person shooter games at the present time.

Real-time Strategy Games (RTS Games): Real-time strategy games are real-time computer games in which players act as a strategist. Players are responsible for

gathering resource, building base infrastructure, developing technology, recruiting forces and conquering territory or citizens of other players. Players get direct control over individual units in RTS games. The RTS games concentrate a war or a whole dynasty into minutes or hours. Empire Conqueror, Warcraft are the most popular RTS games are the typical RTS games widely played during these years.

Role-Playing Games: Computer Role-Playing Games (CRPGs) are originally derived from traditional role playing games, especially *Dungeons & Dragons*, and use both the settings and game mechanics found in such games [14]. If the real-time strategy games is considered as games running on a macro level, the role-playing games is running on a micro level. The players in a role playing games act a real person or someone the like, they improved their ability after each battle or interaction with others based on the game mechanics. World of Warcraft is the most popular role-playing online games at the moment.

Other games: Besides the shooter games, real-time strategy games and role-playing games, there are other type online games, such as racing games, commercial strategy games and sports games. These games have relatively less players and the features of these games are more or less part of the above three types.

3.1.2 Architectures of Online Games

Online game aims to build a game community in which players can acquire higher game service and interact with other players in a real-time manner. The game service provider stores the updated information of players and prevents it from sniffing by malicious network users. These features demand that the game service provider has a high performance server and fixed IP address. Two main architectures, client-server and peer-to-peer structures, are used for multi-player games at present. The features of online games indicate that the C/S structure is much suitable for multiplayer Internet games.

In the C/S architecture, there is usually one host (or a set of hosts) designated as a server to run as back-end applications that all clients are connected to. The server is responsible for account registration and game service management of players. All actions from players are sent to the server. The server then advances the simulation by sending the actions to players, who update their game view after receiving data from server.

The C/S architecture is easy to build compared to P2P architecture for several reasons [5]. First, there is just one connection for each player; so it is simple to implement the client. Second, it is simpler to maintain state consistency with a centralized server, and cheating becomes harder with a centralized validation. Finally, it is easy to monitor the performance of games with a centralized server. However, some problems appear in centralized architectures. First, if the server is down, it crashes all players. Second, high bandwidth is required at the server since all the data are relayed by server. In addition, a server relay method increases the time delay for each update.

In the P2P architecture, each player is responsible for their own game simulation. Peers have to use a distributed agreement protocol (i.e., trailing state [15] or bucket synchronization technique [16]) to resolve issues of inconsistencies. The P2P structure takes advantage of reduced message latency between peers and elimination of a single point-of-failure [5]. But the bandwidth requirement increases linearly with the number of players.

The P2P architecture is not as popular as the C/S architecture in the game industry at the present time. The most important reason is that it is much more difficult to implement and configure compared to the C/S architecture; the other reasons include the lacks of centralized monitoring and generating the revenue via subscriptions for companies.

Based on the two basic architecture discussed above, in [5], a new architecture called Peer-to-Peer with Central Arbiter Architecture are proposed. The information is updated during the communication among each peer and arbiter, the central arbiter only send update to all the players when inconsistencies occur, then the players roll back to the previous state.

3.1.3 Bandwidth Requirements and State Consistency

The bandwidth requirements and state consistency are basic factors that affect the performance of multiplayer games. Suppose that the online games are built on top of networks with effective connection and a relatively constant bandwidth and network speed. In [5], a prototype of band width requirements and state consistency of online games is discussed.

In a multiplayer game, a player executes a certain loop called *player update period* T_U [5] during the game, this loop includes input from the local player, data from remote players and updates of local state. During the *player update period* each player will send certain amount of data, this size of data can vary, depending on the activity of the player during that loop iteration. To simplify the calculation of bandwidth requirement, we assume that all updates have the same size, namely L_U bytes. So if a player sends and receives N updates in every update period, the bandwidth requirement is NL_U / T_U .

The *state* $S_i(t)$ of a game for player i at a time instant t is a collection of all attributes that describe completely the view of the game as player i sees it at time t . If all players have the same state, the game is performing under an ideal state. But there would be some delay between players because of network delays. An action of player i at time t will be known by player j at time $t + D_{i,j}$. The time $D_{i,j}$ can be different due

to the different network delay at every time. We just consider the minimal time value of each pair, and call this time period *inconsistency resolution latency* [5]. The most interesting inconsistency period is the *Maximum Inconsistency Period (MIP) (*T_I*)* [5], which is the maximum inconsistency resolution latency among all players for a given game architecture.

Based on the definition above, some formulas are given for the C/S and P2P architecture on the bandwidth requirement and state consistencies.

For the C/S architecture, the bandwidth requirement for each client is:

$$\frac{L_U + L_G}{T_U} = \frac{(N+1)L_U}{T_U} \quad (1)\ [5]$$

The L_G means the size of global received message, and N stand for N players

The bandwidth requirement for server is:

$$\frac{N(L_U + L_G)}{T_U} = \frac{N((N+1)L_U)}{T_U} \quad (2)\ [5]$$

The *MIP* is:

$$T_I = \max_i \{T_U + T_S + R_{i,S}\} \quad (3)\ [5]$$

The T_S stands for the server latency and $R_{i,S}$ is the round-trip time between player i and server *S*.

For the P2P structure, the bandwidth requirement for each peer is the same:

$$\frac{2(N-1)L_U}{T_U} \approx \frac{2NL_U}{T_U} \quad (4)\ [5]$$

The MIP is:

$$T_I = \max_{i,j} \{2T_U + R_{i,j}\} \quad (5)\ [5]$$

The theoretical formulas of the prototype shows that the bandwidth requirements increase linearly for the client of C/S structure and the peers of P2P structure, but with $O(n^2)$ for the server of C/S structure. Even thought the MIP for online games is not affected by the number of the players, it has different requirement for different type of games.

The multi-player FPS games have the feature of fast pace and highly iteration among players, it requires better network environments than other online games. A performance study of FPS games on the QoS sensitivity was discussed in [17]. The experimental approach is organized a number of game session for both Quake 3 and Halo. The perceived quality is from 1 to 5, where 1 means bad and 5 excellent.

Figure 3-1 shows the mean perceived quality of different round trip times. From Figure 1, the performance is tolerable when the round trip times (RTT) are less than 150ms. But if we want to get a better performance the RTT should be less than 100ms.

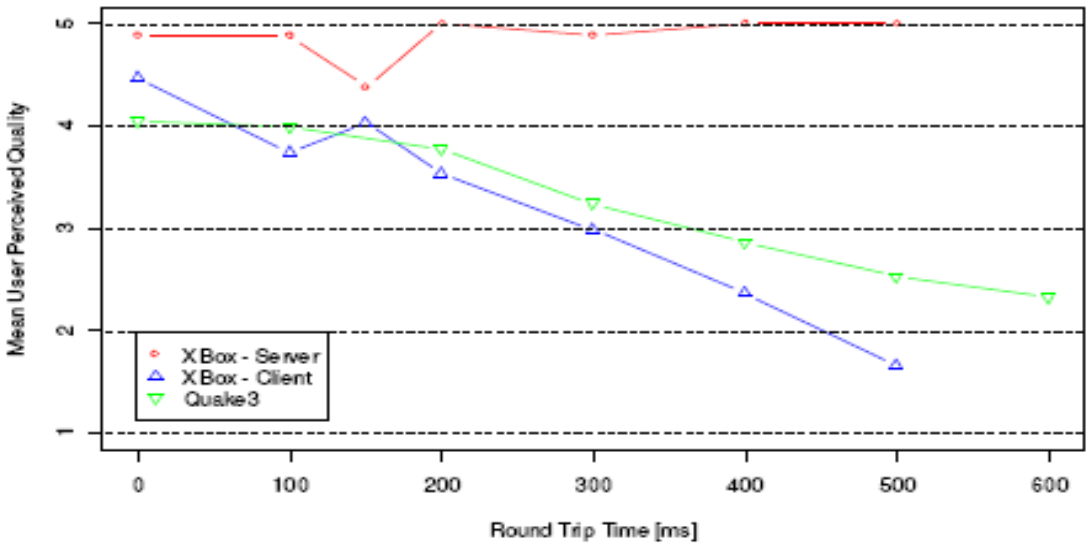

Figure 3-1: Mean perceived quality - Mean perceived quality for different round trip times of Quake 3 [8]

Network testbeds for online games are set up to study the effects of performance of online games related to network latency. A middleware is installed to control the network delay. In [18], experimental data of tolerable latency for online games is analyzed. Figure 3-2 outlines the performance versus latency for different types of games. The perceived quality is from 0(worst) to 1(best). Figure 2 shows the first person shooter game and the racing games are more sensitive to game latency than the Sports, RPG and RTS games.

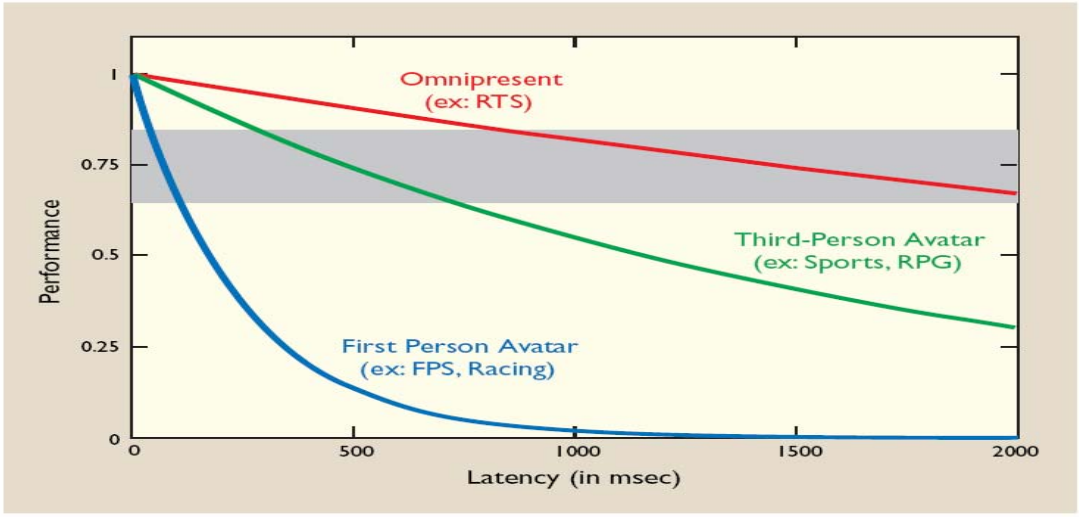

Figure 3-2: Player performance versus latency for game categories. The horizontal gray band represents the threshold for player tolerance for latency [18].

3.1.4 Multi-player Vehicular Games

The features and requirements of online games discussed above mainly concern on the Internet online games, which are supposed to build over the wide-ranged wired networks. To develop multi-player vehicular games, new features of the vehicular ad hoc networks compared to wired networks should be considered. Unlike current online games, multi-player vehicular games are more difficult to build because of the unstable vehicular communications. The network capability is not changed frequently even though traffic congestion occurs during some time. A fixed size of data buffer and propagating data through reliable TCP connection are considered to develop an online game. The unpredictable network reliability and rapid-changing transmission

ability of vehicular ad hoc networks makes it difficult to develop the multi-player games by using the same methods of current online games. Besides, multi-player vehicular games are not supposed to have large amount of players restricted by the number of nodes inside the vehicular ad hoc networks.

Figure 3-3 (see section 3.3.2) illustrates the situations when two players are playing game between vehicle D and vehicle E. The connection between vehicle D and E is supposed to be built in an end-to-end TCP connection to simplify procedures of multi-player vehicular games, but the bandwidth and latency change from time to time due to the rapid-changing topology of vehicular networks and unstable signal strength between vehicles. For a better performance, each player in a multi-player vehicular game should know the tendency of bandwidth to reset their buffer in time for transmitting data in a smart way in a tolerable time delay. The statistic data of the tendency of bandwidth is retrieved from the part of vehicular communication.

The supposed situation above just considers two players in one multi-player game, when the number of players increases, an algorithm should be considered to maintain the connections between players. Besides, different games have different bandwidth requirement and maximum tolerable latency; these factors are more restricted in multi-player vehicular games.

3.2 Wireless Technologies for Vehicular Networks

Wireless access is one of the emerging features of vehicular networks after automobile manufacturers continue to incorporate more and more technology into their automobiles. A network infrastructure is needed to support mobility of vehicular communication. Wi-Fi (IEEE 802.11), WiMax (IEEE 802.16), MBWA (IEEE 802.20), and 3G (ITU) these four wireless technologies become and will become the basic infrastructures for the next-generation wireless networking constructions.

A methodology of comparing the performance, coverage area, reliability, security and mobility of these four wireless technologies is proposed in [19]. By comparing these five characteristics of these wireless technologies, suitable solutions can be found by using a single technology or a combination of technologies.

Performance: bandwidth and latency are the very precious features to determine the performance of these technologies. It determines what type of applications can be run by using the different technologies. When playing an FPS game, latency should be keep below 150ms, otherwise, it nearly impossible for players to play the game.

Coverage Area: the effective distance between two communicating nodes of wireless technologies is the coverage area, which indicates the predicted number of nodes in a certain area. In a wireless ad hoc network, the selected wireless technology determines the maximum contiguous wireless area mostly results from the increased number of hops lower down the performance.

Reliability: the average number of dropped packets, average number of disconnects and whether the technology is affected by environmental factors such as line of sight, weather is measured as reliability [19], which potentially affects the functioning of an application.

Security: transmission of many services requires a certain level of encryption and authentication for sensitive data. It demands the network provides a secure connection from MAC layer communication to application layer.

Mobility: vehicular networks are built on top of running vehicles. The speed of vehicles affects the reliability of connection obviously when vehicles run fast.

	3G
Performance	<2Mbps(bandwith), 500ms(latency)
Coverage Area	Typically 1-5miles
Mobility	310mph at 144kps, 75mph at 384kps 6mph at 2Mbps
Reliability	Non-Line of Sight
Security	Based on GSM with enhancements

Table 3-1: Empirical data for 3G [19]

WiMax and MBWA are the latest technologies to be approved or under development at the moment. Even though the features of these two proposed technologies are attractive for vehicular networking construction, but it is impossible to carry out in real application at the moment.

Wi-Fi and 3G are the latest deployed wireless technologies for the next years. Wi-Fi was developed by the IEEE 802.11 working group aims to solve indoor short range wireless communication initially, but ventured into mobility recently. It includes the approved 802.11a, b, g, n, and p specification. 3G is specified by ITU aims to provide a long-range wireless communication services for voice and data with a high-speed compared to current GSM method. By comparing these two technologies, a suitable solution of network infrastructure can be found for the project. Table 1 and Table 2 show the empirical data of Wi-Fi and 3G.

	Wi-Fi 802.11p
Performance	54Mbps(bandwith), 50ms(latency)
Coverage Area	Up to 1000 feet
Mobility	<100mph
Reliability	Non-Line of Sight
Security	Wired equivalent Privacy

Table 3-2: Empirical data for Wi-Fi (802.11p) [19]

The empirical data in Table 3-1 and 3-2 illustrates that latency is the main issue when try to deploy online games by using 3G wireless technology. It indicates that most of the popular online games such as FPS games, racing games and role-playing games, etc. can not performance well. Compared to 3G the lower latency of Wi-Fi is a better solution for multi-player vehicular games, even though the coverage area of Wi-Fi restricts the maximum number of players of one certain game in a contiguous area.

3.3 Scenarios for Vehicular Communications

Vehicular communication of multi-player games is the most challenging parts of this project. Basically, there are two methods to realize it. The first method is connecting players to the Internet; multiple-hops are possible before a player connects to the Internet. The second method is finding a nearby vehicle dynamically as an intermediate broker for data transmission. Another challenge relates to new features introduced by vehicular communication to multi-player vehicular games, which are different from current online games. New features include rapid-changing topology of vehicular networks and unstable signal strength between vehicles. How to cache and propagate data based on the current vehicular communication environment makes it more complicated to build the car-to car multi-player games.

The adoption of wireless communication technology to support inter-vehicle communication makes the exchange of data fast. Besides, traffic information exchanging between cars, Internet access from vehicles are the other exciting technologies for business and family use. People can solve their emergent business problems on their way to companies by using these technologies. When encountering traffic jam, drivers or passengers can play online games to overcome burden. Some categories of vehicular communications applications are:

– *Applications for business*: traffic information service, email service, vehicular online shopping service, positioning service, navigation service, tolling service and automatic petrol filling service, etc.

– *Applications for entertainment*: car-to-car games, video chatting between cars and music sharing, etc.

In the remainder of section 3.3, two basic methods, based station and peer-to-peer method, proposed for vehicular communication are detailed.

3.3.1 Base Station Method

The foreseeing applications outlined above give us an imaginable picture of the future applications running over automobiles. There are two types of networking access related to the vehicular game applications, 1) connecting to the Internet to play games with other players, which are either in a vehicle or use a desktop computer, 2) connecting to peers to play games with other players in other vehicles, usually in peer-to-peer.

For the first type of networking access, vehicles get access to the Internet via base station by using General Packet Radio Service (GPRS) or High Speed Downlink Packet Access (HSDPA) technologies. A typical structure is shown in Figure 3-4.

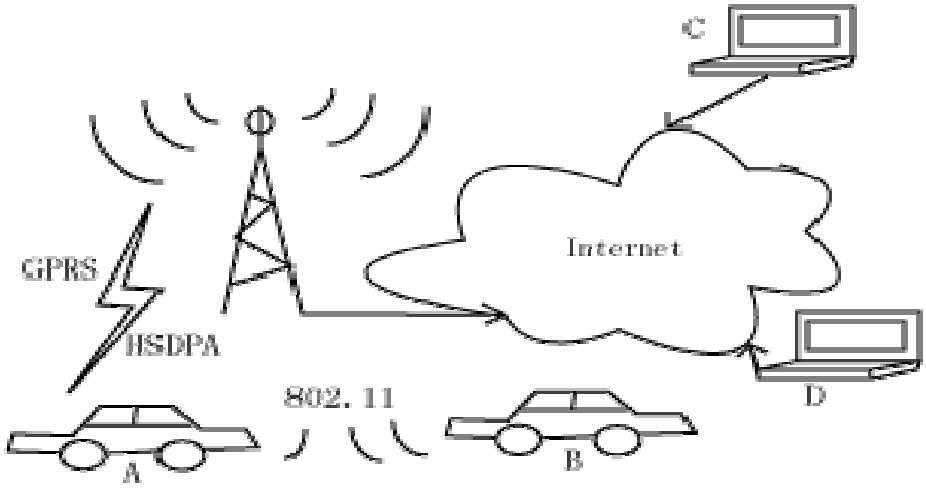

Figure 3-4: network structure of using base stations

HSDPA is a 3G mobile telephony protocol in the HSPA family, which provides a smooth evolutionary path for UMTS-based networks allowing for higher data transfer speeds [30].

Since vehicles are running on roads in an organized fashion with specific speeds and the distance of two base stations is measurable, the dynamic topology among vehicles and base stations is predictable. But periodic changes of internet access point increase the latency compared to the normal Internet access with a fixed location. The latency of 3G is nearly 500ms from the empirical data, so the total latency between two players may be greater than 600ms. A prototype of this networking access can be built by considering the speed of a car, the distance of two base stations. In Figure 26, vehicle A is connected to a base station, which connects directly to the Internet, using GPRS or 3G, vehicle B is connected to vehicle A by using short-range wireless network interface (such as IEEE 802.11p), players in vehicles C and D directly connect to the Internet. Since vehicle B needs one hop before it connects to the Internet, the latency will be greater than vehicle A.

The problems of this structure are 1) when a vehicle leaves one base station for another base station, how to build a stable connection when handover occurs, 2) when a player connects to the Internet via another vehicle using IEEE 802.11 protocol, how

to dynamically maintain the connection between vehicles. The issues of vehicular communication are discussed in section 3.3.2.

3.3.2 Peer-to-peer method

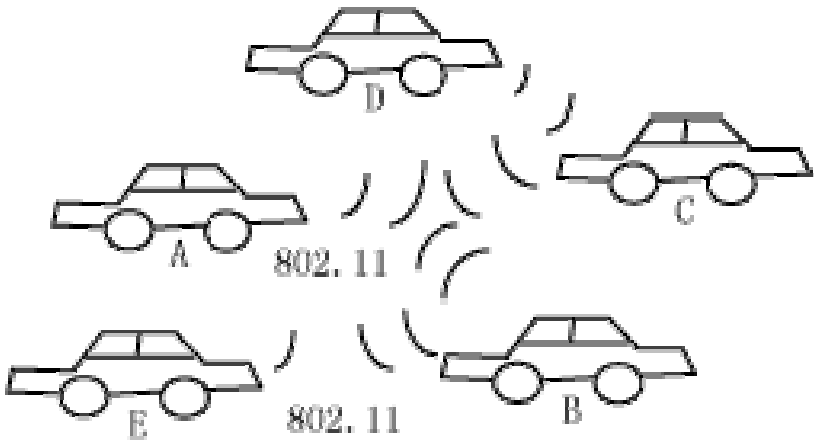

Figure 3-3: network structure of using peer-to-peer methods

The second type of networking access is a “subtle subset” of the first one. It concerns the communication between vehicles rather than the whole network. Propagating data between each vehicle has a peer-to-peer communication feature, and by considering the rapid-changing relative positions of vehicles, a dynamic networking architecture using adjacent vehicle as an intermediate broker to relay data to destination vehicles is adequate for data exchange between vehicles. Figure 3-3 shows the proposed structure of peer-to-peer vehicular communication.

First, supposing vehicle E gets connected with vehicle D and the relative distance between vehicle D and E is expanding, it does not take a long time for the signal strength between vehicle D and vehicle E to reduce into a low level. Before they get disconnected, they should find a new broker as their relay station. The target broker vehicle can be considered in the following factors: 1) the relative speeds between peer vehicles and target broker, 2) the estimated duration of new connections, 3) signal strength between peer vehicles and target broker. But problems may be more complicated than what it is considered here. First, what is the maximum number of

hops can be considered in one continuous connection because signal strength and performance will reduce dramatically due to the increase in hops, and as the number of hops increase, latency becomes larger. Second, how to find a target broker efficiently while the hops are increasing is a potential challenge. The simulation in [27] shows that when the density of vehicles is in a low number, the messages would be stored for a long time, before it finds a new target. For multi-player vehicular games, this phenomenon is not tolerable. Besides, when the hops increase, it is not easy to find a target broker for all the vehicles which are engaged in one connection. Third, when should the system decide to disconnect a connection is another issue facing the dynamic networking topology.

3.4 System Architecture

One aim of this project is to build a game platform for windows CE operating system running over vehicular networks. The functionality of the game platform consists of building a certain vehicular ad hoc network with routing information that vehicles can communicate with each other in time, registering game services in local application, launching game services to other players, discovering game services from other players, updating game services in time for the local players and predicting connection duration between two nodes in vehicular networks.

As discussed in section 3.2, Wi-Fi network infrastructure is a better solution to build a vehicular ad hoc network for the multi-player vehicular games. Based on Wi-Fi a routing protocol is required to find the route among vehicles. As discussed in chapter 2, AODV routing protocol is a reactive protocol which discovers routes to destination only on-demand. In section 3.3, the issues facing multi-player vehicular games demand predication of connection duration. AODV routing protocol does not consider the moving tendency and speed of vehicles when it establishes the route to other nodes, besides, the specific features of vehicular network is also not considered such

as the route information. Even though AODV routing protocol is not an ideal routing protocol for vehicular ad hoc networks, the implementation simplicity of this routing protocol and an existing semi-finished implementation of AODV makes it is a better choice for this project. In order to solve the problems that the AODV routing protocol facing multi-player vehicular games, a module inside the SGP to predict the connection duration among vehicles are proposed.

The other functionality of the game platform indicates that an application running in the application layer as a daemon is required for game services management. Besides, since AODV routing protocol lacks of predicting connection duration for wireless networks, the application should also support the functionality of predicting the connection duration to provide a better performance for the players. One of the differences between mobile ad hoc network and vehicular ad hoc network is that vehicles are running on a certain route which has only two directions and the foreseeing direction of vehicles is predictable. The connection duration is relatively easy to predict by acquiring the speed, direction, route information of vehicles.

Another aim of this project is to connect, discover, launch and play Xbox games via the game platform, meaning that the game platform works as an assistant for the Xbox games like the Xbox Live, which provides services for players who connect to in the Internet. Xbox Live requires players get access to the Internet. In contrast, this game platform works in a peer-to-peer manner, meaning that information of game services is propagated among vehicles, no player is required to connect to the Internet, game services are launched by each player, discovered and updated by other player. Players play games in a vehicular ad hoc network.

The overall structure of the whole game platform is illustrated in Figure 3-5. Each Xbox is connected to a game platform (C2C) running on top of windows CE operating system via wired network connection, the C2C uses the AODV routing protocol to find routes and run an application to manage game services between

vehicles.

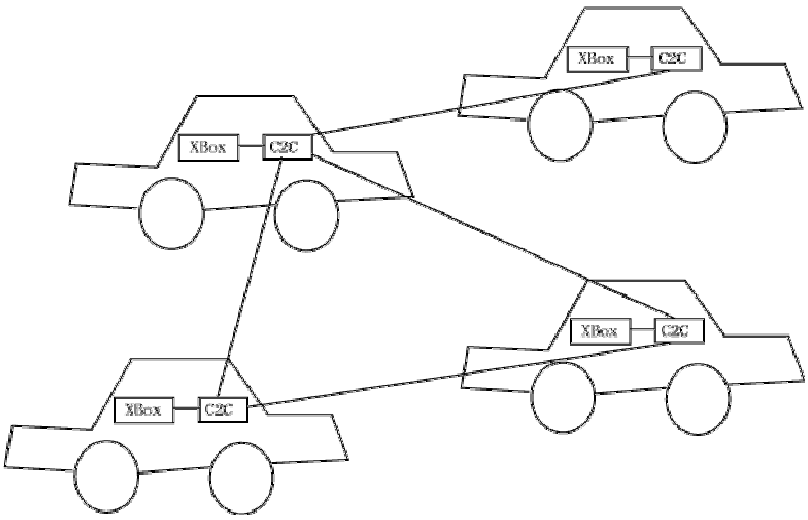

Figure 3-5: system architecture

3.5 AODV Implementation for Windows CE

The wireless ad hoc networks are new network pattern compared to old wired network, which are designed for configured and known links. The kernel routing table of current operating systems is built to store the routes that the originating node can reach by using the proactive routing protocols that running on top of wired networks. When an application layer function attempts to open a socket to a destination which is not store in the kernel routing table, the function call will return an error code immediately. This is a worse situation for the reactive ad hoc routing protocols, such as DSR and AODV since they attempt to find the routing to a destination only on demand. A mechanism must be designed to notify the on-demand routing protocol to start a routing discovery, and in the meanwhile any data packet that intend to send to the destination must be cached by the originating node. The particular required extended protocol stack capabilities, including notification mechanisms, for the AODV routing protocol are [11] [31]:

1. To start a route request when it is required. Reactive routing protocols only start its route discovery when packet is sent by the up-layer application or, a packet to an unknown destination is received from other nodes.

2. To buffer the outstanding packets before a route to destination is found. When a route discovery cycle is needed, the attempted outstanding packets should not be discarded, a queue is recommended to buffer the outstanding packets. After a route to destination is found, these queued packets are then forwarded to network.

3. To determine the lifetime of one cached route. The up-to-date routes to destinations are buffered in AODV routing protocol, but in a wireless ad hoc network, nodes move frequently, some routes expire after a certain period. For the vehicular ad hoc network, the movement of vehicles is relatively predictable. The determination of lifetime for one cached route may also be predictable

4. The daemon functionality for the management of HELLO message and route error message. During a route discovery cycle, if there is no route to destination, operating system drops the packets silently and send a destination unreachable to the originator, instead, a route error message should be generated and send to the originator. Besides, HELLO message is supposed to broadcast periodically to update the routes to destinations.

The functionalities discussed above are supported in modern operating systems. The implementation of AODV routing protocol for Windows CE in this project is based on [11], in which a semi-finished implementation is developed with some critical bugs. A brief description of the methodology adopted in [11] is given in the remainder of section 3.5.

3.5.1 Windows CE Overview

Windows CE operating system is developed for smart devices such as PDAs, smart phones and embedded devices in vehicles. It supports four different CPU families, SHx, MIPS, ARM, and x86.

Some new features are introduced in Windows CE comparing to windows desktop operating system, such as limitation of memory, unicode problem, componentization [4], etc.

Unlike desktop operating system has mass storage device, Windows CE programs almost run on devices that never have disks. This constrains virtual memory technical using in Windows CE and results in low-memory environment for Windows CE programming. Memory allocation may be failed because of limitation of memory recourse, as well as the un-normal termination of applications. This gives a big challenge involved in porting existing Windows applications to Windows CE.

Programs run on Windows CE should be unicode, which is a standard for representing a character as a 16-bit value as opposed to the ASCII standard of encoding a character into a single 8-bit value. Unicode and ASCII standard are both used in current windows desktop operating system, which makes a programmer from windows desktop to windows CE painfully when they encounter character issues.

Windows CE can be broken up and reconfigured for different environments of specific purposes such as Pocket PCs, Windows Automobiles and Smart Phones. So when programming applications for different smart devices, the configuration of components should be considered.

Windows CE removes some redundant functions supported by its larger cousins. It makes Windows CE a subset of the desktop operating system.

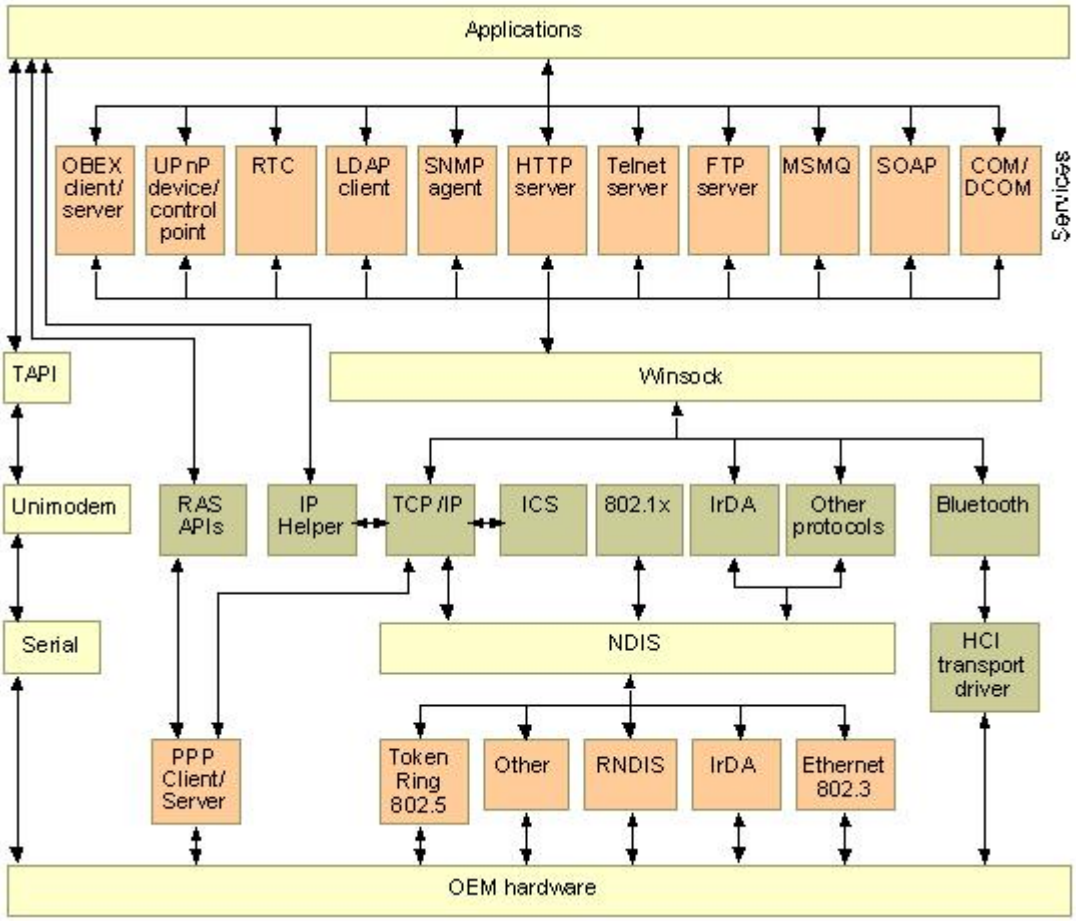

Figure 3-6: Communications and networking architecture - Communications and networking architecture for Windows CE [2]

The networking protocol stack in Windows CE operating system is a “subset” of windows desktop networking protocol stack. Basically, Windows CE operating system provides networking and communications capabilities that enable devices to connect and communicate with other devices and people over both wireless and wired networks. Figure 3-6 illustrates services that Windows CE operating system supported. Manufacturers of Windows CE devices can tail the wanted services and the core components for their devices when building the specific operating system for their devices. The core components include the core protocols, services and interfaces, such as TCP/IP, Winsock and Network Device Interface Specification (NDIS), which facilitates communication between the Windows CE operating system and network

adapter and protocol drivers [2]. The NDIS interface is located between an upper-level protocol driver (such as the TCP/IP protocol driver) on the top of the communications architecture, the intermediate and miniport drivers in the middle of the communications architecture, and a network adapter at the bottom of the communications architecture.

Networking communications between applications by using Winsock (except Bluetooth) in Windows CE operation system is managed by the NDIS interface, data packets are first transmitted to the NDIS, then dispatched to the corresponded infrastructures before reaching networks.

3.5.2 NDIS Intermediate Driver

NDIS driver stacks have three types of drivers, protocol driver, miniport driver and intermediate driver. NDIS intermediate drivers include a protocol driver interface at their lower edge and a miniport driver interface at their upper edge. The protocol interface of intermediate driver allows it to load above a driver with a miniport driver interface. Therefore, intermediate drivers can load above miniport drivers or other intermediate drivers. The miniport interface of intermediate driver allows it to load below a driver with a protocol lower edge interface. Therefore, intermediate drivers can load below protocol drivers or below other intermediate drivers. For the upper layer protocol drivers, NDIS intermediate driver works like a miniport driver, but for the lower layer miniport drivers NDIS intermediate driver works like a protocol driver. Figure 3- 7 shows relationship among NDIS intermediate driver protocol driver and miniport driver.

Before data packets flow to the under layer networks or upper layer applications, the mangling operation can be performed inside the intermediate driver.

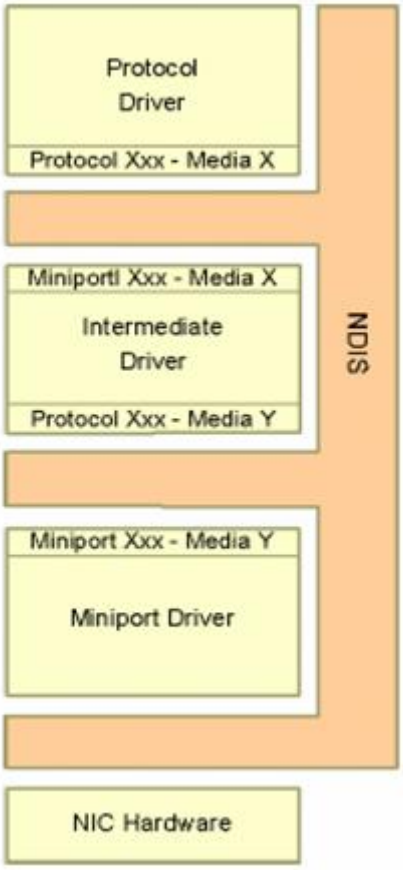

Figure 3-7: NDIS intermediate driver [2]

3.5.3 Implementing AODV as NDIS Intermediate Driver

The particular requirements of implementing AODV routing protocol in modern operating system discussed in section 3.4 indicates that the intermediate driver must fulfill the functionalities from 1 to 4.

Of particular interest of requirement 1, to start a route request when it is required, normally, in the IP layer, packets are discarded if there is no route to the destination. So even though we can find a route to the destination after a route discovery cycle, some packets would be lost. To solve this problem, we convince the IP layer by adding a default rout the destination for the unknown destination, buffers these data packets in a queue. After finding the route to the destination, these data packets are forwarded. This procedure also satisfy requirement 2. Of particular interest of

requirement 3, to determine the lifetime of one cached route, a thread is used to set an instant time value to check routes periodically. The check mechanism does not consider real movements between connecting nodes, but simple removes the routes when they are expired. Another thread is also created to periodically broadcast the HELLO message to check the availability of routes. A route error messages are sent to the originator when packets arriving with a next hop destination for which this node does not have an entry in its routing table. Requirement 4 is satisfied.

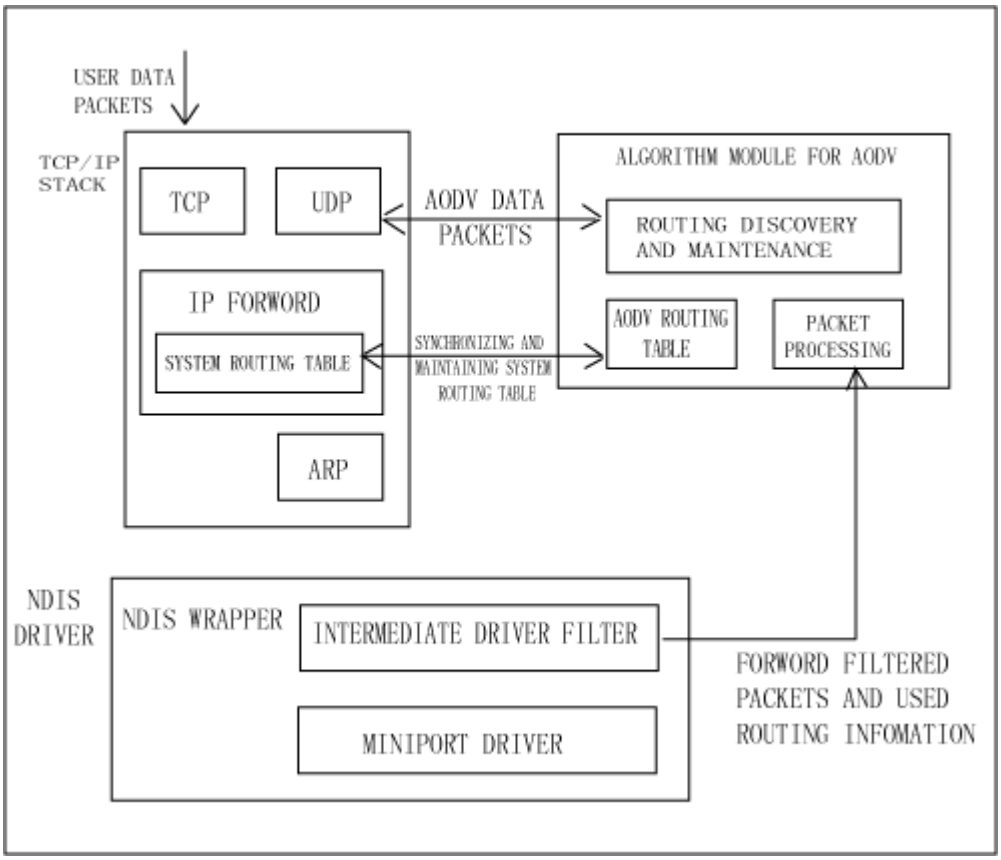

Figure 3-8: architecture of AODV routing protocol for Windows CE

In Windows CE operating system, NDIS intermediate driver runs in the protected user mode, which removes the barrier of communication between application and drivers. Figure 3-8 illustrates the architecture of AODV routing protocol for Windows CE. User data packets from applications are filtered and queued before route to destination are found. When a route is found, synchronization between the system routing table and AODV routing table is performed, meaning that a new route is added into the

system routing table. Then, the queued data packets are sent to destination. Data packets from network are filtered before the algorithm module for AODV makes a corresponded action, such as forwarding data packets to upper layer application when it is the destination, forwarding data packets to a know destination.

The semi-finished AODV routing protocol for Windows CE has some critical bugs.

Bugs and improvements made for it are listed as follows:
Non-Ethernet size address problem
Link list error
Broadcast efficiency improvement
Packets reading issues for non-standard wireless cards
Infinite blocking error
Hard reset problem
Non-same Ethernet routing extension

3.6 Simple Game Platform

When a player wants to play games with other players, the player can try to play the game via LAN, Internet or other methods that can connect two or more players together. When a game is connected to the Ethernet, the game itself searches other players and tries to connect them. It is not feasible for an outsider who is not in the same private network tries to play games with the players inside another private networks, unless they are all connected to a game service or adopt the Network Address Translation (NAT) [26] method which enables multiple hosts on a private network to access the Internet using a single public IP address.

The situation of vehicular ad hoc network users has changed compared to wire network users for playing games, even though users may be in the same Ethernet because of the instability and dynamic changing topology of ad hoc wireless network,

especially for games which need stable network status, low latency and relatively long-time connection duration. It is nearly impossible for an ad hoc network maintains its connection while nodes are in a high speed moving from one place to another, such as running vehicles on roads, but the tendency of moving nodes can be predicted, if velocities, directions and locations of vehicles are known. By analyzing this information, connection duration and foreseeing signal strength of existing connections are predictable.

Games with a LAN support searches other players in the same Ethernet, but without any assurance whether there is another player who intends to play the same game inside the same Ethernet until it finds one. A platform is desired to give an exact answer before a game started.

The remainder of section 3.6 is divided into two sections. Section 3.6.1 describes algorithms, namely Connection Duration Prediction (CDP), designed for SGP and gives an overview of SGP. In section 3.6.2, the architectural details for SGP are discussed.

3.6.1 Algorithm for SGP

Connection duration and foreseeing signal strength among moving nodes are predicted by SGP by analyzing the periodically-updated velocities, directions and locations of the nodes.

In the following discussion this section, we assume that connections are built from one node to another by using black arrows. Figure 3-9 shows a typical scenario of these moving nodes.

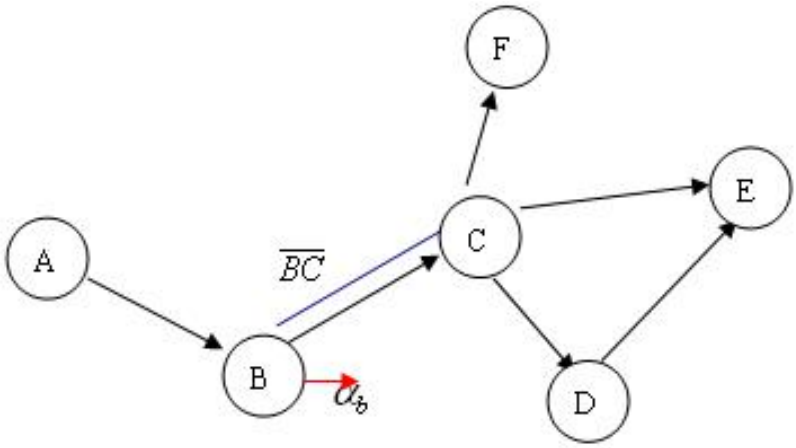

Figure 3-9: Scenario for moving nodes in MANET

At time t_0, Node B is supposed to move in direction along the red arrow α_b, it has a north velocity v_{yb} , an east velocity v_{xb}, a horizontal angle of the velocity vector in radian θ_b. The position vector from node B to node C is $\overline{BC}$, and the angle of the position vector in radian is φ_{bc}. In the meanwhile, the corresponding values for node C are $[\alpha_c, v_{yc}, v_{xc}, \theta_c, \overline{CB}, \varphi_{cb}]$. Since we analyze the instant moving tendency of Node C based on node B, we build a Cartesian coordinate system for node B and node C shown in Figure 3-10.

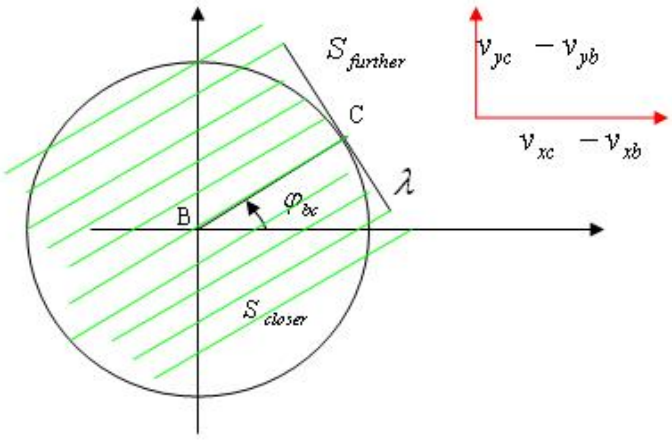

Figure 3-10: Cartesian coordinate system for node B and C

At time t_0, Node B is located at the origin of Cartesian coordinate system and node C is located at point C. The relative north velocity between node B and C is $v_{yc} - v_{yb}$ (from node B to node C), and so the relative east velocity is $v_{xc} - v_{xb}$, angle φ_{bc} indicates the angle of position vector of node C in Cartesian coordinate system. Since node B and node C are on moving at time t_0, we make a triple $[v_{yc} - v_{yb}, v_{xc} - v_{xb}, \varphi_{bc}]$ for node C relative to node B which is supposed to be still at time t_0.

There are two instant moving tendencies of node C related to node B at time t_0, one is getting closer (marked as S_{closer}) to node B and the other is getting further (marked as $S_{further}$). Circle O is drawn with the origin as the center $(0,0)$ and with a radius of $|\overline{BC}|$. Tangent λ is drawn with the point C as the point of tangency.

If at time $t_0 + \Delta t$ (Δt is very small value), node C is below tangent λ (we define a node below tangent λ when a point is getting closer to circle O, above when getting further), we say that node C is in status S_{closer}, otherwise node C is in status $S_{further}$.

Based on the knowledge of Analytic Geometry, we compute that if ϕ, satisfies the value range of formula (6), then node C is in status S_{closer}, otherwise node C is in status $S_{further}$.

$$\frac{\pi}{2} + \varphi_{bc} \le \phi \le \frac{3\pi}{2} + \varphi_{bc} \tag{6}$$

The ϕ means the angle of the relative velocity vector between node B and node C.

Supposed that there is a route from node A to node C via node B, the instant moving tendencies of node B related to node A and node C related to node B are computed by

formula (6), then the instant moving tendency of node C related to node A can also be computed. We mark S_{closer} as -1 and $S_{further}$ as 1, we mark the tendency of two neighboring nodes (node i and node j) as $S_{i,j}$, which has two values $\{1,-1\}$. So the instant moving tendency of node C related node A is defined as follows:

If there is one value equals $S_{further}$ then the whole instant moving tendency between A and C is $S_{further}$. The instant moving tendency is the same for a route with N nodes. Based on the assumption at the beginning of this section, route from node A to node C is established via B, if a sub-route is getting worse, then the whole capacity of the route becomes worse. By acquiring the instant moving tendency from the originator to destination, instant tendency of connection quality is predictable.

In order to predict the signal strength between node A and node C, we mark the signal strength of two neighboring nodes (node i and node j) as $\omega_{i,j}$, then for simplicity we select the minimum value of the signal strength as the signal strength between node A and node C, thus:

$$\omega_{A,C} = \min\{\omega_{A,B}, \omega_{B,C}\} \quad (7)$$

For a route which has N nodes the equation is:

$$\omega_{A1,An} = \min\{\omega_{A1,A2}, \omega_{A2,A3}, ..., \omega_{An-1,An}\} \quad (8)$$

By using this equation, the predicted final signal strength will be a bit better than the actual signal strength, because other factors are not considered.

For vehicles that running on a certain road, the instant moving tendencies among cars are not enough to predict the relatively long-time connection duration. Figure 3-11 shows a scenario of two moving vehicles in the same road.

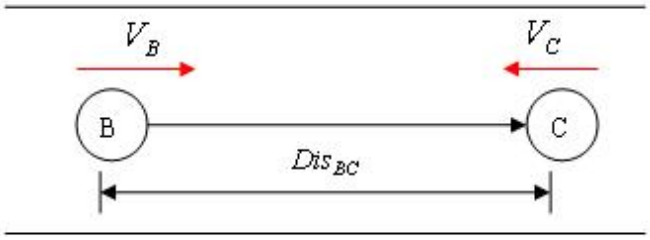

Figure 3-11: Scenario of two moving vehicles 1-Scenario for two moving vehicles in the same road

There are only two directions of the moving vehicles. The velocities along north and east are defined as positive, while along south and west are negative.

At time t_0, vehicle B is supposed to move with speed V_B in point P_B, and corresponding values for vehicle C are $[V_C, P_C]$, the theoretical effective distance of signal between two vehicles is Dis_{Theo}, and the current distance between vehicle B and C is Dis_{BC}. $S_{B,C}$ (has two values, one is S_{closer} as -1 and $S_{further}$ as 1) is defined as the instant moving tendency between vehicle B and C., so $S_{B,C}$ equals:

$$S_{B,C} = \begin{cases} S_{further}, If\ \ (V_C - V_B)(P_A - P_B) > 0 \\ S_{closer}, \quad otherwise \end{cases} \tag{9}$$

If the speed of vehicle B and C is stable in next certain period, then the foreseeing connection duration T_{BC} between B and C can be computed as:

$$T_{B,C} = \begin{cases} +\infty, V_B - V_C = 0 \\ \dfrac{Dis_{Theo} - S_{B,C} \bullet Dis_{BC}}{|V_B - V_C|}, V_B - V_C \neq 0 \end{cases} \tag{10}$$

Formula (10) does not consider the issues shown in Figure 3-12, in which vehicle B and C are supposed to be in the same road in a certain small area within a certain time

interval. In Figure 10, vehicle B has a route R_B and vehicle C has a route R_C. At time t_0, they are in the same road L_1, but C intends to leave road L_1 at point P_1 and turn to road L_2, whereas B still in road L_1 in the next few minutes.

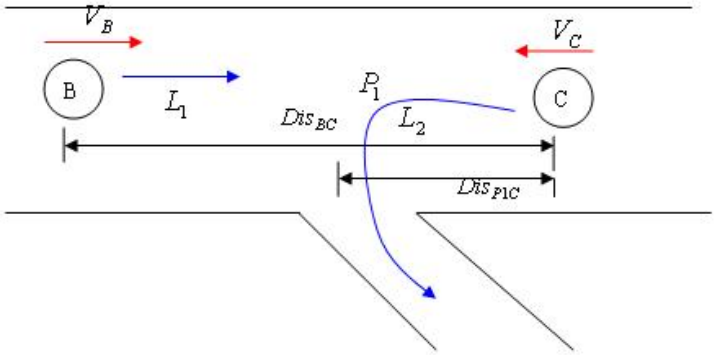

Figure 3-12: Scenario for two moving vehicles 2- Scenario for two moving vehicles, vehicle C leaves for another road

Given the fact that most of wireless ad hoc networks for vehicles are built in a city, where roads are among buildings, signals are mostly prevented by buildings, we assume that the connections would break soon after one vehicle leaves for another road while another vehicle does not follow. The real effective distance Dis_{Eff} between vehicle B and C is defined to replace the theoretical effective distance Dis_{Theo}.

$$Dis_{Eff} = \begin{cases} Dis_{Theo}, \ If \ \ Dis_{Theo} < D_{B,P1} \\ D_{B,P1}, \ \ Otherwise \end{cases} \tag{11}$$

So the modified formula is:

$$T_{B,C} = \begin{cases} +\infty, V_B - V_C = 0 \\ \dfrac{Dis_{Eff} - S_{B,C} \bullet Dis_{BC}}{|V_B - V_C|}, V_B - V_C \neq 0 \end{cases} \tag{12}$$

The result of $Dis_{Eff} - S_{B,C} \bullet Dis_{BC}$ should be larger than zero, otherwise, formula (12) is of no use.

For a route with N vehicles, whenever the connection between two neighboring vehicles is broken, the whole route is broken, so the predicted connection duration is:

$$T_{A1,An} = \min\{T_{A1,A2}, T_{A2,A3}, ..., T_{An-1,An}\} \quad (13)$$

The information of these vehicles is broadcasted periodically at each vehicle. By using the instant moving tendency formula, formula (8) and (13), the instant tendency of signal strength and relatively long-time connection duration can be predicted. The algorithms discussed above is named

3.6.2 Architecture of SGP

When a player is connected with other players in a wireless ad hoc network, the player first checks whether there are other players intending to play the same game. SGP registers games a player intend to play and advertises periodically the related game information to other players who are in wireless ad hoc network. Figure 3-13 shows how the information of games is registered, advertised, and updated.

The information of games is registered by the GameServiceRegistration into the GameServicePool. Periodically, the GameServiceDispatcher dispatches the information of games by calling the DataAggregation module which gets the GPS information from Garmin system. The traffic information of one particular vehicle is analyzed and combined with the information of games into one combine data packet and then broadcasted to other players by calling the BroadcastSender. By listening to one particular port, when a combined data packet is received by the BroadcastListenerThread, the received data packet is decapsulated then passed to the VehicleConditionAnalysis module if it is a valid data packet and is not broadcasted by

itself. The information of another vehicle is analyzed together with the information of vehicle from the owner for the purpose of the foreseeing duration and signal strength of connections. Then the games information is updated by the UpdateGameServiceInfo module and displayed to players.

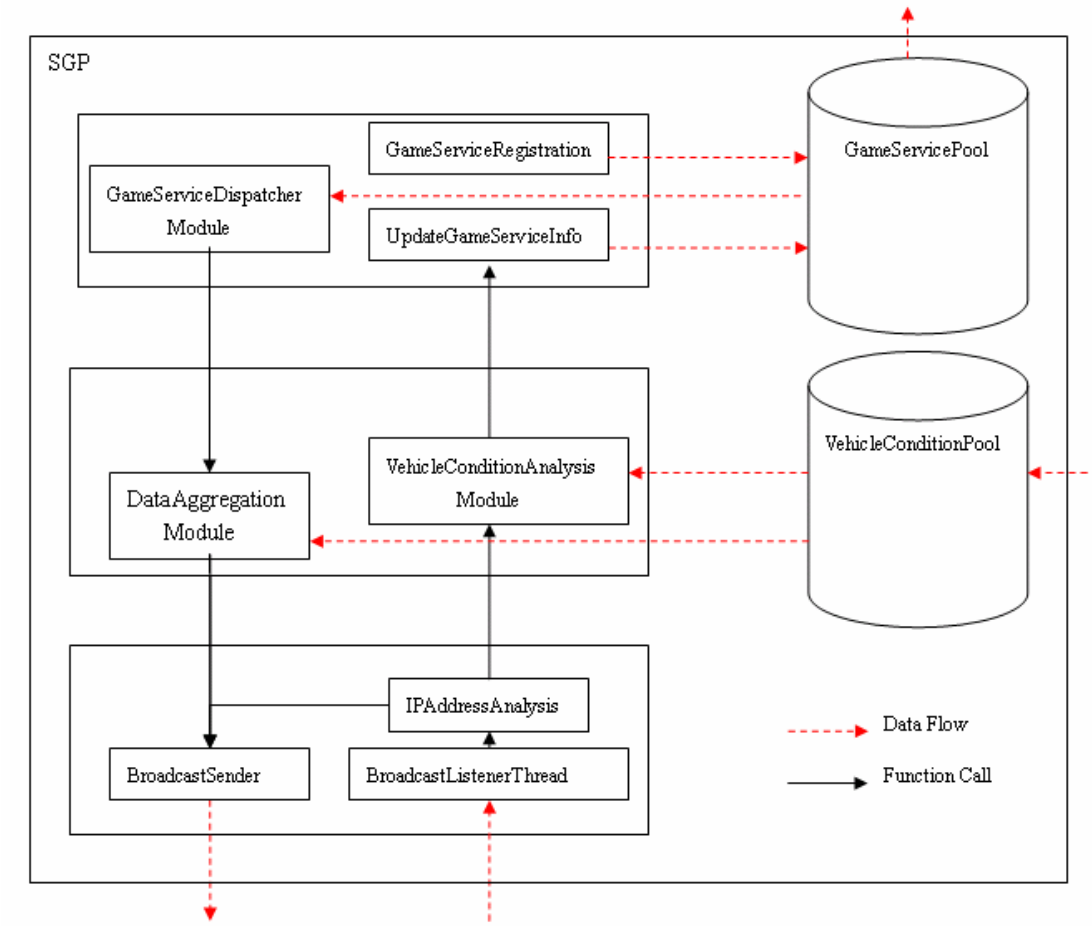

Figure 3-13: Architecture of SGP

A SGP header is defined for the data packet transmitted among nodes. It contains two main structures, one for game services, and another for vehicles. The GameService structure contains the following data:

- ***hopCount***: an integer value, time to live of one SGP packet
- ***seqNum***: an integer value, sequence number of the generated SGP packet, increments by one after each broadcasting
- ***gameID***: an integer value, game ID mapping to one particular game

The VehicleData structure contains the following data:

- ***currentRoadName***: a string value, current driving road name
- ***currentPosition***: a Position structure, current position of vehicle
- ***nextWaypointName***: a string value, next significant waypoint for leaving
- ***nextWaypointPosition***: a Position structure, position of next significant waypoint
- ***nextWayoutDistance***: a double value, distance from current position to the next significant way point
- ***direction***: a Boolean value, positive while the direction of vehicle is along north or east, otherwise negative
- ***speed***: a Speed structure, speed of vehicle

The Position structure contains the following data:

- ***lat***: a float value, latitude component of the position in radians
- ***lon***: a float value, longitude component of the position in radians

The Speed structure contains the following data:

- ***eastVelocity***: a float value, the East component of the velocity in meters per second.
- ***northVelocity***: a float value, the North component of the velocity in meters per second
- ***radian***: a float value, the horizontal vector of the velocity in radians

4 Implementation

4.1 The Software

In this section, the adopted framework and software including .Net Compact Framework, Garmin system and OpenNetCF.Net, that our system running on top of are described briefly.

4.1.1 .Net Compact Framework

.Net Compact Framework is a smaller version of .Net Framework runtime written for embedded and battery powered devices such as device running Windows CE operating system, Xbox console, etc. Programs running above .Net Compact Framework are supposed to compile in a managed code. The unique requirements of embedded devices make it a challenge to write applications using only managed code. Embedded applications and some mobile applications require the application to be tightly integrated with the device. Because one of the features of the runtime is to isolate the hardware from the application, an embedded managed application sometimes needs to break the bounds of the runtime and directly access some operating system functions [4].

The GPS APIs of Garmin Mobile XT are implemented in an unmanaged code and the SGP network communication part that broadcasts the game information via wireless cards is also implemented in an unmanaged code, meaning that they are not built on top of the .Net Compact Framework but directly get access to the drivers of devices or some operating system functions.

4.1.2 Garmin Mobile XT

The Global Positioning System (GPS) is a global navigation satellite system made up of a constellation of 24 satellites placed into orbit by the U.S. Department of Defense.

It works in any weather conditions, anywhere in the world, 24 hours a day, enables a GPS receiver to determine its location, speed and direction.

Newer Garmin GPS receivers with WAAS (Wide Area Augmentation System) capability can improve accuracy to less than three meters on average [6].

The Garmin Mobile XT is a part of the Garmin Software Development Kit. Garmin Mobile Smartphone SDKs provides mobile applications running on Windows Mobile or Palm OS devices with access to GPS information, interactive maps and intelligent routing including live traffic conditions. With these tools, developers can easily create Location-Based Service (LBS) solutions for a wide range of Palm OS and Windows Mobile devices that are often already in use by enterprise customers [6].

4.1.3 OpenNETCF.Net

OpenNETCF.Net is a part of the OpenNETCF Smart Device Framework, which is compiled in managed code providing developers with many useful classes to use within their .NET Compact Framework applications. It provides some “additional” functionality for the .Net Compact Framework. The OpenNETCF.Net provides usefully libraries for the network applications that built for the Windows CE operating system, such as providing information of wireless network and wireless cards.

4.2 Implementation Description

In [11], a module description is presented for AODV routing protocol, since the basic structure is based on the existing implementation, for the details of AODV routing protocol implementation, please refer to [11]. Bugs fixed in AODV routing protocol are detailed in section 4.2.1.

The SGP is divided into three layers, game services layers, vehicular information layer and network layer. The first two layers are built on top of .Net Compact

Framework by using C#, the network layer is coded in an unmanaged code which is written in C.

4.2.1 Bugs Fixed in AODV Routing Protocol

Non-Ethernet sized address problem: in version 0.1, the function for comparison of two MAC layer address were supposed to have a standard Ethernet size. It caused system crack during the debugging stage. A new function was written to satisfy the generalized sized address.

Link list error: in version 0.1, a grammar error, the pointer to the list did not point to the next data of the link list, may cause an infinite while loop.

Broadcast efficiency improvement: in version 0.1, messages are broadcasted to all the interfaces with IP address 255.255.255.255, even though the AODV routing protocol just binding itself to one certain interface of the listed interfaces of network cards, this burdens the operating system to check all the listed interfaces. By restricting the selected interface which, in fact, is an IP address, the broadcasting procedure only contacts the binding interface of the wireless network card.

Packets reading issues for non-standard wireless cards: in version 0.1, all incoming packets are assumed to in one buffer and exactly at the first buffer out of the whole buffer chains. This is not general fact of the incoming data packets. A whole transverse of the buffer chains should be performed to prevent data packets loss. We added the *ReadBuffersFromPackets* function to fulfill the general requirement. The following codes show the algorithms, two functions *NdisQueryBufferSafe* and *NdisGetNextBuffer* work together to get the rest of the data from the data chain.

```
nNdis_Status = NdisAllocateMemoryWithTag(&lpBuffer,uiTotalBufferLength,TAG);
lpBufferOut = lpBuffer;
```

```
 if(nNdis_Status != NDIS_STATUS_SUCCESS){
            return;
 }
do{
            NdisQueryBufferSafe(CurrentBuffer, &VirtualAddress,
                                    &CurrentLength, NormalPagePriority
                                  );
      if( !VirtualAddress ){
                  return;
      }
      NdisMoveMemory(lpBuffer,VirtualAddress,CurrentLength);
      lpBuffer += CurrentLength;
      *lpNumberOfBytesRead += CurrentLength;
     NdisGetNextBuffer(CurrentBuffer, &CurrentBuffer);
      if (!CurrentBuffer){
                  return;
      }
}while(!CurrentBuffer);
```

Infinite blocking error: in version 0.1, the initialization of AODV routing protocol could not be finished because of an infinite blocking error. It was designed on the assumption that each time when started the service of AODV routing protocol in Windows desktop operating system, there is IP change for the AODV routing protocol module since AODV routing protocol service is newly added-in and no memory is saved for the last usage. The fact has changed when AODV routing protocol is running inside the Windows CE operating system. It is no longer a newly added-in service, but implemented as internal driver for the system, status is stored. The Register inside the Windows CE operating system records the successfully-bound interface of the listed interfaces. So, each time when Windows CE is started, there may be no IP address changed, while the function *NotifyAddrChange* blocks until a IP changes.

Hard reset problem: in version 0.1, the assumption of power management of Windows CE operating system is also based on Windows desktop operating system, but the fact is, power management of Windows CE operating system is only a subset. Many functions are not supported. We simply change the status of power management

to NDIS_STATUS_NOT_SUPPORTED.

Non-same Ethernet routing extension: in version 0.1, route discoveries were restricted within the same Ethernet. We extend it by adding an expanded searching interface.

Version 0.2 of AODV routing protocol gives better performance than version 0.1, but there are still bugs inside. The obvious inefficiency of this implementation is that it tried to modify the IP_TTL by using the *setsockopt* API from Winsock, but the fact is Windows CE operating system does not support this functionality at present. It does not affect the results when the hops are less than the default value defined by Windows CE operating system, but when the expected hop count is larger than the default value, it could possibly prevent the expecting results come out, besides, if the expected hop count is less than the default value, it would also decrease the performance of AODV routing protocol. In order to solve this issue, the whole structure may need to be changed.

4.2.2 Module Description for SGP

Define.cs

This file contains basic definitions of the whole system, the error codes, const values and operations are defined as static structures. The important SGP header is also defined here together with useful functions for data conversion. Besides, meta-data for one single game service is also defined here together with the basic functions for data operation, such as the single game service definition and the data pool for game service.

GameServices.cs

The extended operations of game services are implemented in this file, functionalities provided here including clearing all game services, adding new game services,

updating game services and removing one certain game service.

GameServiceDispatcher.cs, GameServiceInfoUpdate.cs, GameServiceMonitor.cs, GameServiceRegistration.cs

These four files extends the functionalities defined in GameServices.cs, class *GameServiceDispatcher* works as thread, which dispatches information of game services periodically, class *GameServiceMonitor* is another thread, monitoring the received information of game services, whenever a game service expires, it will be removed from the game service pool. The definition of expiration is explained later. Class *GameServiceRegistration* provides APIs for a player to register a new game into SGP. Class *GameServiceInfoUpdate* provides APIs for the under layer when a game service should be updated.

GPSLib

This is a separated part of SGP. It is compiled as a wrapper of GPS management in SGP. It contains operations and algorithms related to Garmin system. This dynamic link library imports the APIs from Garmin system to get the basic GPS data as well as route information. Functions for instant moving tendency and connection duration are exported as APIs.

Algorithms are implemented based on the formulas discussed in section 3.5.1, computation of the instant moving tendency between two neighboring vehicles are implemented with a bit different from formula (9). Based on the definition of positive, if the relative coordinate from owner to P1 (P1 minus owner) is above the line $x + y = 0$, thus, $x1 + y1 > 0$, then we say the relative position from owner to P1 is positive, otherwise negative. Then by computing the sign of difference between two speeds of vehicles, the whole value comes to us. Even though the longitude and latitude components of position are measured in radians, a function is provides to compute the distance between two nodes, otherwise, if we only need the sign of the differences, no change has been made.

The leaving points of vehicles are defined as the significant points along routes a vehicle following, such as corners of road, crossroad, or T-junction of road. Based on the speed of vehicle, data packets are periodically broadcasts to neighbors to update the whole network.

SGPLib

This is another separated part of SGP. It is compiled as a wrapper of basic networking operation such as open socket, close socket from Win APIs, which are compiled as an unmanaged code. It also provides management for broadcasting messages to other players. The receiving data from network layer is implemented as another thread, which listens to a specific port for SGP application.

SGPLib.cs, GPSModule.cs

These two files are the wrappers of the *SGPLib* and *GPS.Lib* for the importing APIs from those two modules.

DataAggregation.cs, WirelessCards.cs

These two files provide the functions for aggregating information of game service, vehicle data and wireless cards data together. Class *wirelesscards* gets the signal strength between two wireless cards from the APIs supplied by OpenNetCF.Net.

SGP.cs, Program.cs, SGP.Design.cs

These three files provides the implementation of user interface, the entry of application and functions for system configuration and management.

5 Evaluation

5.1 Evaluation Goals

The game platform includes two separate parts: the AODV routing protocol which runs inside the protected space of Windows CE operating system, and the SGP which runs in the user space on top of .Net Compact Framework and Win APIs. The tests for these two parts are first evaluated separately, and then a combinational evaluation of these two parts is carried out.

Evaluation goals of the AODV routing protocol for Windows CE operating system are divided to two parts, the first goal is to test the basic functionalities including RREQ message, RREP message, RERR message and HELLO message as well as performance and coverage tests, and the second one is to test the interoperability of the AODV routing protocol between Windows CE operating system and Windows XP operating system.

Evaluation goals of SGP are to test the registering, discovering and launching functionalities. The GPS module of SGP was not tested since the Garmin system does not support the virtual GPS device and we do not have one at the moment when we carried out our tests.

A combinational test for the game platform is carried out among three PDAs by using two real PPC games (one is a racing game and another is a real-time strategy game), which support multi-player in Ethernet. The real performance of AODV routing protocol and SGP are measured during the test.

5.2 The Tests

The tests are carried out inside among HP iPaq 5500 series with accessories of Slots

and three same 802.11b wireless cards (Cisco AIRONET 350 series wireless LAN adapter) and a Lenovo laptop (Thinkpad T60). Three PDAs built one wireless ad hoc network via the 802.11b wireless cards. For the test of interoperability of AODV routing protocols between Windows CE and Windows XP, one wireless ad hoc network is built among two PDAs and one laptop since we only have three wireless cards.

5.2.1 AODV Routing Protocol Testing

Tests were carefully carried out for the AODV routing protocol because it runs in the kernel of Windows CE operating system. A small error in the kernel of Windows CE system may cause the whole system crashes down. In order to make the test real, we traced the running AODV routing protocol by dumping the debugging information inside one txt file. To test the basic functionalities, we used the application vxUtil, which is designed to facilitate the network development and test for Windows CE operating system, to send ping messages to other device as well as to trace the routes to one destination. For multi-hop test, three PDAs moved away from each other in a line until we could not find the routes. The distance between two PDAs with the longest distance was also measured. Parts of the debugging information are shown below.

--- AODV Intermediate Driver v0.2 ----------
AODV: ProtocolBindAdapter successfully allocated packet pools
==>PASSTHRU::ProtocolBindAdapter CISCO1
AODV: ProtocolBindAdapter successfully allocated packet pools
==>PASSTHRU::Miniport Initialize: pBinding 0096C7A0
<==PASSTHRU::MiniportInitialize: pBinding 0096C7A0, status 0
AODV: Successfully added faked arp entry for interface 2
AODV: Successfully added default root for interface 2
AODV: Route Added Successfully

AODV::EnableAodvOnAdapter success, status 0

AODV: Initialization complete, AODV is running.

<==PASSTHRU::ProtocolBindAdapter: pBinding 0096C7A0, status 0

Generating a RREQ for: 192.168.0.6

Generating a RREQ for: 192.168.0.6

AODV: Route Added Successfully

AODV: Route Added Successfully

AODV: Broken link as next hop for - 192.168.0.5

AODV: Route Deleted Successfully

AODV: Broken link as next hop for - 192.168.0.6

AODV: Route Deleted Successfully

AODV: Route: 192.168.0.5 has expired and is being deleted from the Route Table

AODV: Route Added Successfully

AODV: Route: 192.168.0.6 has expired and is being deleted from the Route Table

AODV: Route Added Successfully

AODV: Route: 192.168.0.6, adapter gone, deleting route entry!

AODV: Route Deleted Successfully

AODV: Route: 192.168.0.5, adapter gone, deleting route entry!

AODV: Route Deleted Successfully

==>PASSTHRU::CloseAdapterComplete: pBinding 0096C7A0, status 0

<==PASSTHRU::CloseAdapterCompleteAODV: Last reference to adapter gone, freeing structure

The debugging information above shows that a route request message is generated when a call from application is started, if a destination is found, a route to the destination is successfully added into the kernel routing table. Periodically, HELLO messages are broadcast to neighbors to check if the existing route is available. When routes existing in routing table expire, they are deleted.

The results of multi-hop test show that when routes to a destination require one hop,

the effective distance is less than the two times of neighboring distance. Besides, the sequences of established routes of multi-hop do not corresponding to the physical distance among nodes even though the other factors influence the signal strength can be ignored. For example, physically node C is in the middle of node A and B, the routes from node A to C can be either node A directly connects to C, or node A first connects to B then to C if C is first turn off and route from node A to B is established.

Performance tests carried out for AODV routing protocol are based on 1) comparisons of real latency between two neighboring nodes by just using 802.11 and the latency among nodes by installing AODV routing protocol. The results show that the average latency between two neighboring nodes without installing AODV routing protocol is approximately 2ms, after installing the AODV routing protocol, the average latency between two neighboring nodes is 3ms, when a ping message is sent to a destination via an intermediate PDA, the average latency among them is approximately 5ms on the condition of a good signal strength.

Tests among PDAs and laptop show the similar result as the tests among PDAs, except that the AODV routing protocol for Windows XP was not implemented very well, that we needed to reset the AODV service in our laptop many times before it could work well.

5.2.2 SGP Testing

Functional Tests of SGP were done among three PDAs, a daemon was written to simulate the registering procedure by periodically adding a new game service into system. When a game is registered, system propagated the newly-added services to other PDAs, Figure 5-1 shows the user interface of the working SGP. Game Services are displayed as a tree structure at the top of SGP, when a game service is selected, the corresponding information of players is displayed.

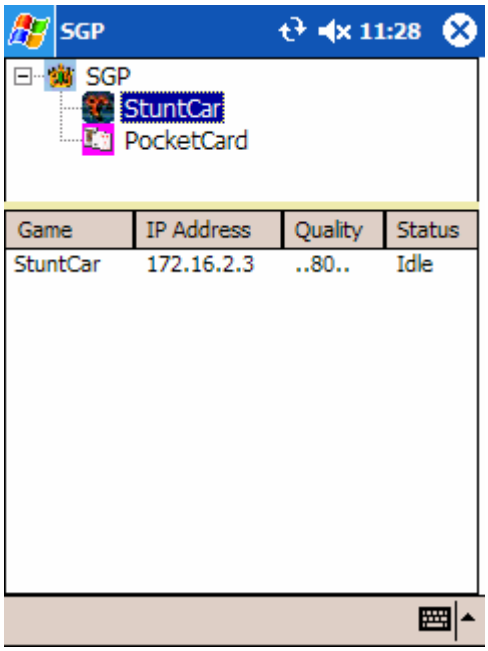

Figure 5-1: User interface of SGP

5.2.3 Game Platform Testing

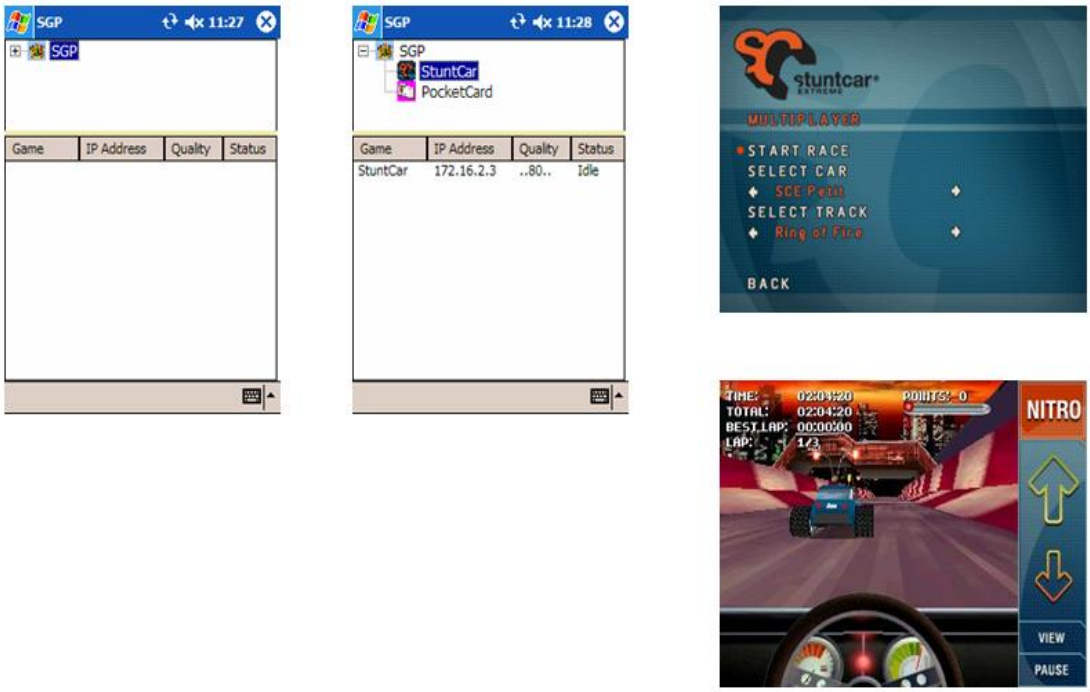

Figure 5-2: Snapshot of racing game

Real performance tests for the game platform was carried out by using two existing PPC games among PDAs, the first one was a racing game, which was sensitive to

latency, the second one was real-time strategy game. Game services were first launched by the SGP, and then by clicking the line of player information, the real game was started, Figure 5-2 and Figure 5-3 show the details of the tests. When playing multi-player games of racing game and strategy game, performances of them shows attractive results. The overall gaming stages were finished without latency.

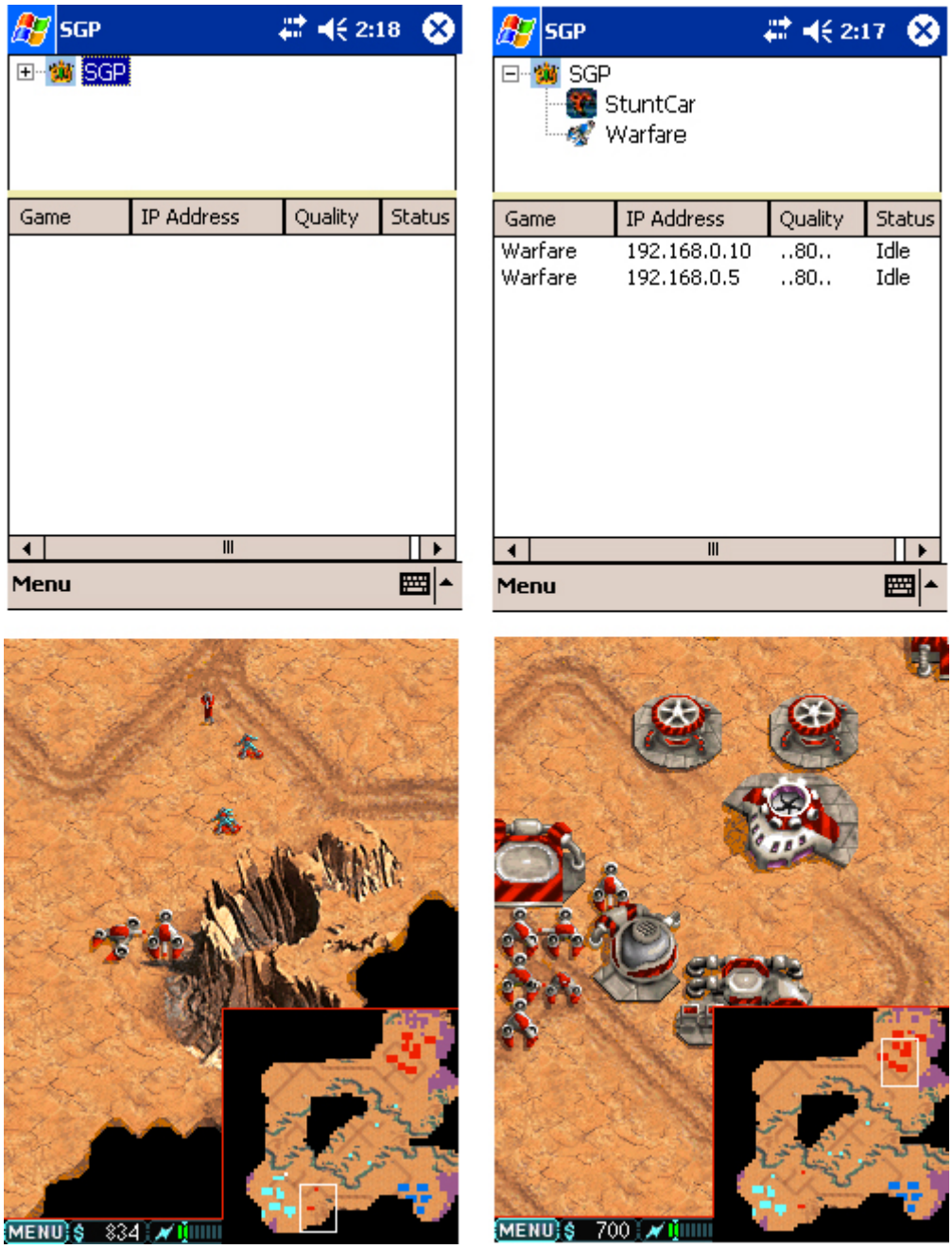

Figure 5-3: Snapshot of RTS game

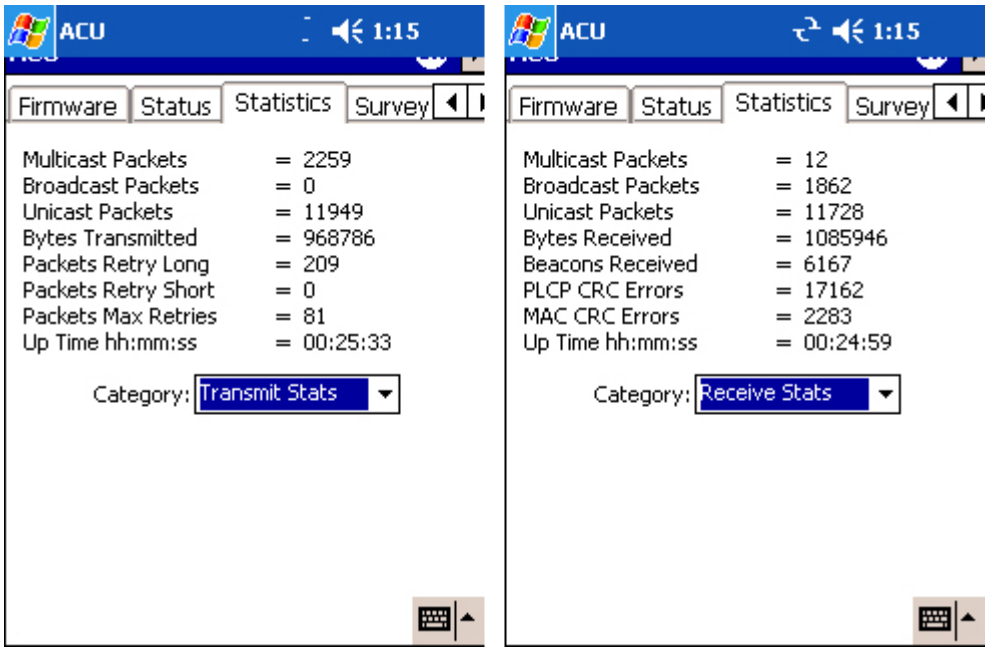

Figure 5-4: Data packets information 1- information of data packets after one game test under the condition of good network connection

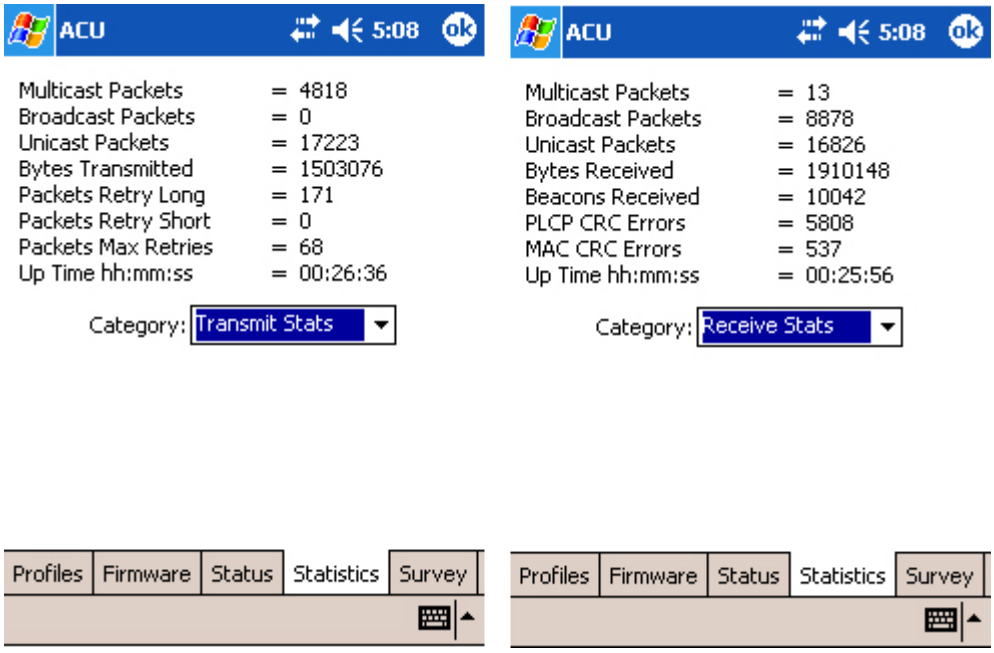

Figure 5-5: Data packets information 2- information of data packets after one game test under the condition of relatively worse network connection

Figure 5-4 shows the information of data packets after one game test, the transmitted and received bytes represents the total bytes that flows through the wireless cards, and

the unicast packets represents the data packets use to transmit the information of game during the game-playing time. Since the SGP and AODV routing protocol implemented dose not exchange the game services and routing information by unicast packets, the multicast packets and broadcast packets represents their total costs. These Figures from Figure 5-4 shows that, the ratio of the usage of data packets (the total of unicast packets) for the game out of the total number of multicast, broadcast and unicast packets is approximately 85 percent under the condition of good network connection. Figure 5-5 shows information of data packets after another game test under the condition of relatively worst network connection, in which the ratio is approximately 74 percent.

5.3 Evaluation of System

5.3.1 Result Analysis

Tests carried out among three PDAs and laptop reveals good results, the basic functionalities of game platform were well built, and the whole system runs stably. Even though current stage gives us a better performance, but there are still some issues we should consider:

1) The AODV routing protocol does not suitable for vehicular ad hoc network because of its neglects of the physical position of vehicles. The second issue of AODV routing protocol is its routs finding strategy, just as the result we have in our tests, when a new node is moving to an existing vehicular ad hoc network, routes to the new node will mostly build on top of the existing routes. This may not be a problem for the wireless ad hoc networks with low moving nodes, but for the vehicular ad hoc network, the performance decreases dramatically because of old routes breaks frequently. A lot of data packets would be used to find a new route. The useful packets to build the route are of low percentage in the whole beaconed packets.

2) The tests for connecting Xbox via Windows CE operating system are not carried out because of no real embedded PC device is available during current stage. We first want to connect Xbox to PDA, but our PDA does not wired connection. We have also tried to connect the Xbox via Windows XP desktop running Windows CE emulator, but of no use. The Windows CE emulator running inside Windows XP does not support the wireless cards, especially when a network service need direct access to the kernel of Windows CE operating system. The emulator network part of Windows CE communicates with the Windows XP through a virtual driver which tunnels data from network to the emulator.

3) The tests for GPS module of SGP are not carried out either since there was no Garmin device during our test stage, even though the basic GPS information is simulated, the route information from Garmin system is not available. The GPS module call the APIs from Garmin Mobile XT, no simulator for Garmin Mobile system is available at present time.

4) Large scale tests should be considered for further evaluation, the speed of vehicle, the number of connecting nodes, and more games are supposed to consider.

6 Conclusion

At the beginning of this dissertation, a lot of algorithms and protocols for VANET are introduced based on the applications they run and conclusions of these algorithms and protocols shows that they are not suitable to migrate to our projects since they mainly focus on the information propagation among vehicles and pay little attention on the prediction of connection duration. We select a semi-finished implementation of AODV routing protocol for Windows CE operating system to fix as our wireless ad hoc routing protocol. By analyzing the online game requirements and the scenarios for vehicular communication for multi-player games, a peer-to-peer structured game platform is proposed, implemented and evaluated.

To predict the connection duration among nodes in a VANET, we introduce an algorithm by analyzing the information of GPS data and routes from Garmin system, and import the APIs from OpenNETCF.Net to get the signal strength between neighboring nodes.

The evaluation of AODV routing protocol for Windows CE shows that it not suitable for vehicular communications because of its neglects of the physical position of vehicles and its strategies for route discovery.

The goals we have fulfilled at current stage are:

- The requirements of multi-player vehicular games demand low latency of network for a better performance, especially for the FPS games and racing games. C/S structure is not suitable for information exchange of multi-player games because of the mobility of vehicles.

- Most of the bugs of the semi-finished AODV routing protocol implementation

are fixed, and a simple game platform has developed to register, dispatch and update game services for players before they start to play games.

– Real tests carried out among three PDAs and laptop to test the performance and the interoperability of AODV routing protocol implementation between Windows CE and Windows desktop operating systems shows a good results. Tests for SGP have also given a good answer.

6.1 Future Work

Inter-vehicle communication has been becoming an attractive research area during and in the next decades, but still there are not matured algorithms and protocols to fulfill the requirements. Our project shows a good picture for the future vehicular applications for entertainment. Based on the solution proposed and evaluated in this dissertation, future work includes:

– To test the GPS module of SGP, the GPS module predicts the connection duration among vehicles by analyzing the information of vehicles from Garmin system, real devices of Garmin system are recommended, since there is no simulator for Garmin system at present.

– Because the AODV routing protocol is not suitable for VANET, some modifications are supposed to carry out. One possible method to improve the performance is to integrate the GPS module into the AODV routing protocol by considering the information of the moving vehicles. Since the implementation of AODV routing protocol is implemented inside the kernel mode of Windows CE to acquire a good performance, some modifications are needed to import the output from GPS module.

– In section 2, we conclude that, even though data packets can be propagated in the situation with a low density of vehicles, the latency is intolerable for applications like multi-player vehicular games, chat services and multimedia services. To cope with this problem, communication via 3G is a tradeoff, when short breaks occur.

– The current method of connecting Xbox via Windows CE to play games together is a bit device consuming and complex since it is nearly impossible to gather all these devices together to configure and establish the connection. This Xbox platform is also built on top of the earlier Windows CE platform and the Xbox platform supports the .Net Compact Framework, so, it may be possible migrate the AODV routing protocol and the SGP into the Xbox platform directly.

Bibliography

[1] C. Perkins, E. Belding-Royer, S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", http://www.ietf.org/rfc/rfc3561.txt, IETF, July 2003

[2] MSDN homepage, http://msdn.microsoft.com, Internet, Sept. 2007

[3] A. Wigley, S. Wheelwright, R. Burbidge, “Microsoft .Net Compact Framework,” Microsoft Press, 2003

[4] D. Boling, “Programming Microsoft Windows CE .NET,” Third Edition, Microsoft Press, 2003

[5] J.D. Pellegrino, C. Dovrolis, “Bandwidth requirement and state consistency in three multiplayer game architectures” *In Proceedings of the 2nd workshop on Network and system support for games***,** Redwood City, California,2003, Pages: 52 – 59

[6] Garmin Announces Developer Toolkit for GPS, http://www.gpslodge.com/archives/011520.php, Internet, Jul 2007,

[7] Garmin Mobile XT Programmer Reference, Garmin Mobile XT SDK, May, 2007

[8] Royer, E.M, Chai-Keong Toh, Santa Barbara, “A review of current routing protocols for ad hoc mobile wireless networks,” *In Personal Communications, IEEE*, Volume 6, Page(s) 46-55, Apr 1999

[9] D. B. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad-Hoc Wireless Networks,” Mobile Computing, Page(s) 153-81, 1996

[10] V. D. Park and M. S. Corson, “A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks,” *In Proc. lNFOCOM 97*, Apr 1997

[11] David West. “An Implementation and Evaluation of the Ad- Hoc On-Demand Distance Vector Routing Protocol for Windows CE”, Trinity College Dublin, 2003

[12] C-K. Toh, “A Novel Distributed Routing Protocol To Support Ad-Hoc Mobile Computing,” *In Proc 75th Annual lnt'l. Phoenix Conf. Comp. and Commun.,* Page(s) 480-86, Mar. 1996

[13] First Person Shooter, http://en.wikipedia.org/wiki/First-person_shooter, Internet, Sept 2007

[14] Role-playing game, http://en.wikipedia.org/wiki/Role-playing_game,Internet, Sept 2007

[15] E. Cronin, B. Filstrup, and S. Jamin, “An Efficient Synchronization Mechanism for Mirrored Game Architectures”, *In Proceedings of ACM NETGAMES,* April 2002

[16] L. Gautier, C. Diot, J. Kurose. “End-to-End Transmission Control Mechanisms for Multiparty Interactive Applications on the Internet”, In Proceedings of IEEE INFOCOM, April 1999

[17] Empirically Measuring the QoS Sensitivity of Interactive Online Game Players, http://caia.swin.edu.au/pubs/ATNAC04/zander-armitage-ATNAC2004.pdf, Internet, Dec 2006

[18] M. Claypool, K. Claypool “Latency and Player Actions in Online Games,” *In Communication of the ACM*, volume 49, No II, Page(s) 40-45, Nov. 2006

[19] Bringing Wireless Access to the Automobile: A Comparison of Wi-Fi, WiMAX, MBWA, and 3G, *http://www.rh.edu/~rhb/cs_seminar_2005/SessionB3/ribeiro.pdf* Internet, Dec 2006

[20] T.Nadeem, S. Dashtinezhad, S. Liao, “TrafficView: A Scalable Traffic Monitoring System,” *In Processing of the 2004 IEEE International Conference on Mobile Data Management.* Volume 5, Page(s):2946 – 2950, May 2004.

[21]W.Kellerer, “(Auto) Mobile Communication in Heterogeneous and Converged World,” *In IEEE Personal Communication*, Vol.8 (6), Page(s).41-47, Dec, 2001.

[22] P. Zhou, T. Nadeem, P. Kang “EZCab: A Cab Booking Application Using Short-Range Wireless Communication,” *In Pervasive Computing and Communications, 2005. Third IEEE International Conference on 8-12 March 2005,* Page(s):27 – 38.

[23] Exploring the Design and Implementation of Vehicular Networked Systems, *http://www.cs.rutgers.edu/discolab/traffic/papers/nsf-nets05.pdf*, Internet, Sept. 2007,

[24] A. Klemm, C. Lindemann, O.P.Waldhorst, “A special-purpose peer-to-peer file sharing system for mobile ad hoc networks,” *In Vehicular Technology Conference, 2003. VTC 2003-Fall.2003 IEEE 58th*, Volume: 4, page(s): 2758- 2763, Oct. 2003

[25] J.Jetcheva, Y. Hu, D. Maltz, “A Simple Protocol for Multicast and Broadcast in Mobile Ad Hoc Networks,”
http://www3.ietf.org/proceedings/01dec/I-D/draft-ietf-manet-simple-mbcast-01.txt, IETF Internet Draft (work in progress), July 2001

[26] Aravamudan, Murali, Tzeng, Hong-Yi, "Method for network address translation," US Patent 6,006,272, 1999

[27] Z. D Chen, H. T. Kung, D. Vlah, "Ad hoc relay wireless networks over moving vehicles on highways" *In Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing table of contents,* Page(s) 247 – 250, 2001

[28] M Saito, M Funai, T Umedu, "Inter-vehicle Ad-hoc Communication Protocol for Acquiring Local Traffic Information," *In Proc. 11th World Congress Intelligent Transport System,* 2004

[29] M Saito, J Tsukamoto, T Umedu, "Evaluation of Inter-Vehicle Ad-hoc Communication Protocol," *In Proceedings of the 19th International Conference on Advanced,* 2005

[30] High Speed Downlink Packet Access, http://en.wikipedia.org/wiki/HSDPA, Internet, Dec 2006

[31] V. Kawadia, Y. Zhang and B. Gupta, "System Services for Implementing Ad-hoc Routing Protocols," *In Proceedings of International Conference on Parallel Processing Workshops,* 2002