Vanet-X: A Videogame to Evaluate Information Management in Vehicular Networks

Sergio Ilarri
IIS Department
University of Zaragoza
Zaragoza, Spain
silarri@unizar.es

Eduardo Mena
IIS Department
University of Zaragoza
Zaragoza, Spain
emena@unizar.es

Víctor Rújula
IIS Department
University of Zaragoza
Zaragoza, Spain
han.vikktor@gmail.com

ABSTRACT

Vehicular Ad Hoc Networks (VANETs) are attracting considerable research attention, as they are expected to play a major role for Intelligent Transportation Systems (ITS). Thus, according to a recent survey by ABI Research[1], about 62% of new vehicles will be equipped with vehicle-to-vehicle (V2V) communications by 2027. Vehicular networks offer new opportunities for the development of interesting mobile applications for drivers, but at the same time they also bring challenges from the data management point of view. Thus, for example, techniques should be developed to estimate the relevance of the information exchanged among the vehicles and to propagate the relevant data in the network efficiently and effectively. As testing the proposals in a real large-scale scenario is impractical, simulators are often used.

In this paper we present *Vanet-X*, an online multiplayer driving videogame that we have developed to help in the difficult evaluation task of data management strategies for VANETs. The idea behind the proposal is to exploit the potential of players around the world driving vehicles in the videogame to effortlessly collect data that can be used to extract some conclusions and fine-tune the proposed data management strategies. So, for example, the videogame allows to evaluate if a certain data management strategy is able to provide useful information to the driver/player (i.e., if the presented information represents an advantage for him/her). We argue that this videogame can be a good complement for existing simulators. As a proof of concept, we have performed some preliminary tests that show the potential interest of the proposal.

1. INTRODUCTION

The widespread availability of mobile devices and the development of wireless communication technologies (such as Wi-Fi, WAVE, etc.) have encouraged the development of services for drivers within the context of *Intelligent Transportation Systems* (*ITS*). In particular, *Vehicular Ad Hoc Networks* (*VANETs*) have become an attractive research area [1, 14, 15, 20, 24, 26, 30]. In these vehicular networks, the vehicles can exchange information directly by using short-range wireless communication technologies. This decentralized architecture provides some advantages over other solutions such as the use of 3G communications: e.g., no need of an infrastructure, quicker transmission of safety-related data in the vicinity, localized communications without the need of a centralized server, and free of charges (which also encourages the participation of peers in the network). Numerous types of events can be relevant for drivers (e.g., accidents, traffic congestions, an ambulance asking the right of way, available parking spaces, etc.). These events can be exchanged in the vehicular network and stored locally by the vehicles. Then, a query processor can periodically evaluate the interest of those events and decide if they should be shown to the driver; there may be implicit queries (e.g., information about an accident in the direction of travel will be relevant for any driver) and explicit queries (e.g., a driver may indicate his/her interest in finding an available parking space or in receiving information about other specific types of events).

However, although VANETs offer interesting opportunities for the development of data services for drivers, they also bring new challenges. Thus, several difficulties arise from the point of view of data management [5]. As an example, estimating the relevance of events in order to disseminate them effectively and efficiently in the network is a challenge [2]. Similarly, disseminating information about a scarce resource (e.g., an available parking space) to many vehicles can lead to competition situations among them to try to reach the resource [7]. As a final example, the relevance of events must also be considered in order to decide if a specific event received by a vehicle should be shown to the driver or not [3].

A big challenge is how to evaluate the data management techniques proposed. Evaluating them in a real scenario with a significant number of vehicles is simply impractical and expensive. Therefore, simulations are frequently used in this field. However, even with simulations the evaluation task can be very time-consuming. For example, many proposals depend on a number of parameters that can be fine-tuned for a given scenario (e.g., see [2, 31]), and determining a good choice of parameters for general evaluation is quite challenging. On the other hand, crowdsourcing strategies where users play the role of drivers could help to

[1] `http://www.abiresearch.com/press/v2v-penetration-in-new-vehicles-to-reach-62-by-202`.

introduce human behavior and facilitate new tests initiated by the users themselves.

So, in this paper we propose a complementary approach that can be used in conjunction with the use of simulators. In particular, we argue that we can benefit from players having fun with a driving game to easily collect interesting data that can be used to extract some conclusions and fine-tune the proposed data management strategies. The videogame is inspired by the classic videogame *Rally-X* (`http://www.klov.net/game_detail.php?game_id=9259`, videogame released in 1980) but it is a new development, with different goals, game modes, and spirit. So, the basic idea is that the vehicles can receive information through the vehicular network and different data management techniques can be plugged in the videogame (e.g., different data dissemination strategies). Data received from other vehicles, if evaluated as interesting by the local query processor in the car, are shown on a radar and can provide a competitive advantage to the player. During the game, a variety of data are collected (e.g., number of messages received by the vehicles, network overhead, time required by the vehicles to complete their goals in the game, etc.), that can be analyzed later. So, while playing, players contribute to collect data for a variety of scenarios, and these data can be exploited to evaluate the effects of particular data management strategies.

The structure of the rest of this paper is as follows. In Section 2 we describe the high-level architecture of the videogame and its features. In Section 3 we summarize the main behaviors implemented for the computer-managed vehicles. In Section 4 we present some basic aspects about the way the data are collected for later analysis. In Section 5 we present the results of the first experiments that we have developed as a proof of concept. In Section 6 we present some related work. Finally, in Section 7 we present our conclusions and some lines of future work.

2. ARCHITECTURE AND FEATURES

Vanet-X is a car videogame that can be played by multiple players connected to the Internet (see Figure 1 for a snapshot showing parking spaces).

2.1 Main Features

We summarize some features of the game as follows:

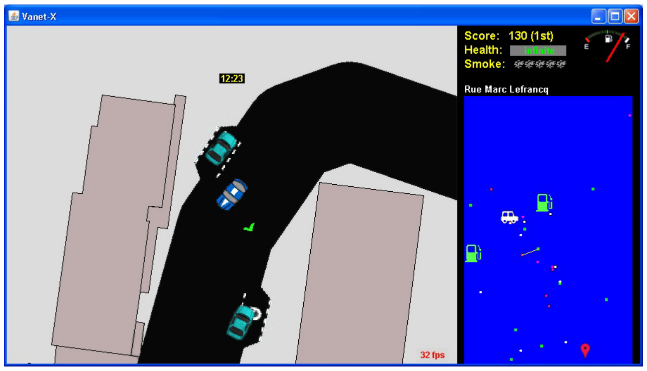


Figure 1: Cars and parking spaces in Vanet-X

- It is implemented in Java as a Java applet, so only a Java Virtual Machine and a browser is needed to play. A desktop application version is also available.

- Both real (human) and computer-controlled players can participate in the game. Human players can join a game through the Internet.

- The game can be configured to execute on a server and create new games when necessary. Alternatively, the computer of any user can play the role of a server and start a new game that other users can join.

- Any real map can be used in the game, by selecting and downloading the data of the desired area from *OpenStreetMap* (`http://www.openstreetmap.org/`).

- To increase the playability, real maps are combined with some extra elements, such as enemy cars, smoke emission devices to disturb enemies (see Figure 2), evolution of events in game time rather than in real-world time, higher maximum speeds for cars controlled by humans, when the driver has a task to go to a certain building he/she has to park nearby and then go by foot to the destination (he/she will be a vulnerable target for enemy cars, that will try to hit him/her, as shown in Figure 3), the car can get damaged and be repaired by paying a certain price (points accumulated during the game), there is infinite or limited fuel depending on the game mode (requiring refueling in a petrol station when running out of fuel in the second case), etc.

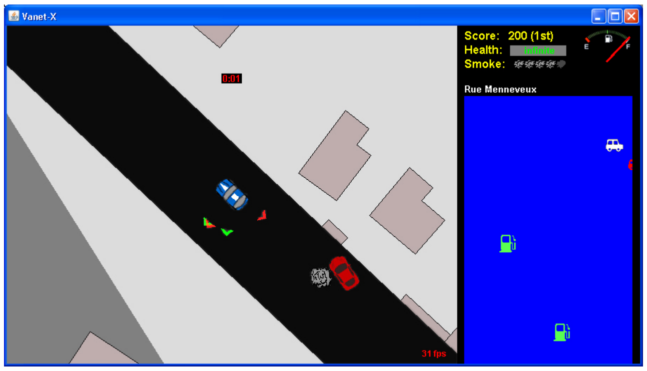


Figure 2: Trying to escape from an enemy vehicle

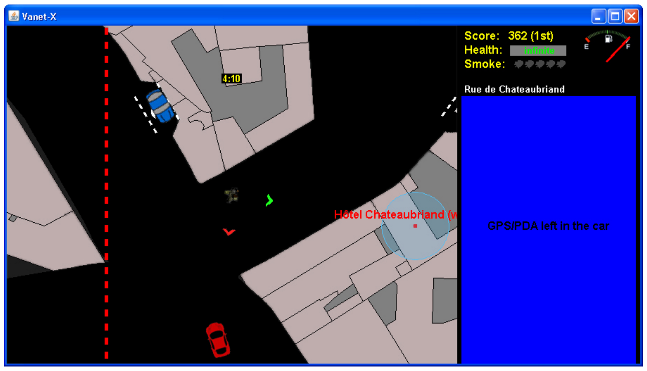


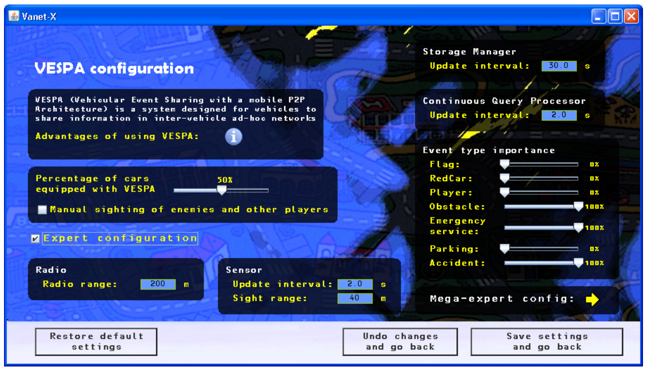
Figure 3: Driver going by foot

- A wide range of game modes is available (see Table 1). Thus, for example, we offer games where the goal is to collect some items along the roads, and task-based games where the players have to complete a series of goals in sequence (with tasks such as parking the vehicle, going to a certain building/business or address, etc.) as soon as possible to win the game. As shown in the table, some game modes can be cooperative, competitive, or both. For the tasks implying going to a certain location, the task may require reaching that location with the car, park and then get there by foot, or just park as near as possible (in this last case, the score for completing the task will be inversely proportional to the distance between the parking location and the final destination). Competitive games involves from 1 to 4 teams in the game, being the winner the team that obtains more points during the game.

Game mode	Multiplayer mode	Inmortal	Possibility to get out of the car and walk	Infinite fuel
Capture the flag (capture 5 flags)	cooperative competitive	no	no	configurable
Capture the enemy cars (1 or more)	cooperative competitive	yes	no	no
Solve tasks (1-3 tasks)	cooperative competitive	no	yes	configurable
Survival (1 or 2 tasks)	cooperative competitive	no	yes	configurable
Park (find one available parking spot)	competitive	yes	yes	no

Table 1: Summary of game modes

- Some default data management strategies, inspired by the work performed in the VESPA project [2, 3, 4, 6, 7], have been implemented. Different tuning parameters can be modified through the graphical user interface of the videogame (see Figures 4 and 5). Moreover, the design of the videogame allows an easy integration of other data management alternatives.


Vanet-X

VESPA configuration

VESPA (Vehicular Event Sharing with a mobile P2P Architecture) is a system designed for vehicles to share information in inter-vehicle ad-hoc networks

Advantages of using VESPA:

Percentage of cars equipped with VESPA: 50%

☐ Manual sighting of enemies and other players

☑ Expert configuration

Radio
Radio range: 200 m

Sensor
Update interval: 2.0 s
Sight range: 40 m

Storage Manager
Update interval: 30.0 s

Continuous Query Processor
Update interval: 2.0 s

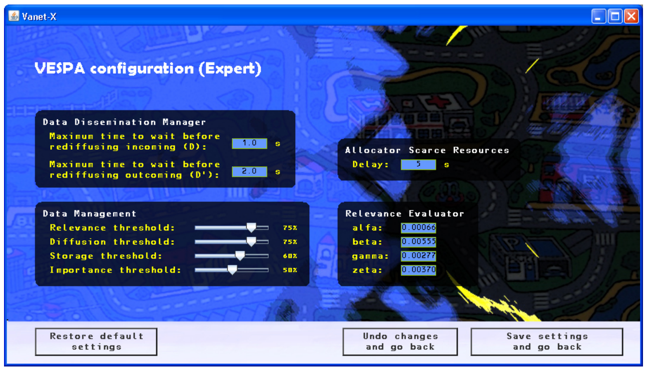
Event type importance
Flag: 0%
RedCar: 0%
Player: 0%
Obstacle: 100%
Emergency service: 100%
Parking: 0%
Accident: 100%

Mega-expert config:

Restore default settings | Undo changes and go back | Save settings and go back

Figure 4: Data management: basic options

- There is a "radar" (e.g., on the right part of Figure 2 we show a basic radar, and on the right part of Figure 1 a radar in debug mode that shows some extra elements about the scenario) that can provide some information to the players. For example, a player can see the following on the radar: his/her location, the petrol stations, and the destination location (if any). Besides, if the option to use a data sharing strategy for that vehicle has been enabled, it will also show data about interesting events received from other vehicles, such as free parking spaces, enemy vehicles, items to


Vanet-X

VESPA configuration (Expert)

Data Dissemination Manager
Maximum time to wait before rediffusing incoming (D): 1.0 s
Maximum time to wait before rediffusing outcoming (D'): 2.0 s

Allocator Scarce Resources
Delay: 5 s

Data Management
Relevance threshold: 75%
Diffusion threshold: 75%
Storage threshold: 60%
Importance threshold: 50%

Relevance Evaluator
alfa: 0.00066
beta: 0.00555
gamma: 0.00277
zeta: 0.00370

Restore default settings | Undo changes and go back | Save settings and go back

Figure 5: Data management: advanced options

pick up (e.g., flags in Figure 6), priority vehicles like ambulances, etc.

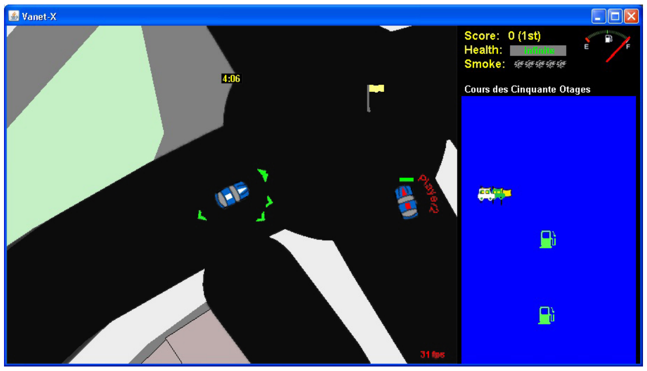


Figure 6: Picking up flags during the game

From a more technical point of view, we have used the Java programming language to develop the video game. Besides, some auxiliary libraries have been useful. For example, we use *Apache Xerces2 Java* (`http://xerces.apache.org/#xerces2-j`) to extract data from the XML files obtained from OpenStreetMap, *JLayer* (`http://www.javazoom.net/javalayer/javalayer.html`) to decode and reproduce MP3 files for the game music, *Guava-12.0* (`https://code.google.com/p/guava-libraries/`), etc.

2.2 Basic Architecture

The basic architecture of the videogame is presented in Figure 7 (the part concerning the collection of statistics about the game is not shown here, as it will be described in Section 4). At a high-level, we can briefly describe the main components as follows:

- A *client* application receives commands from the player, sends them to the server, and receives from the server information about the objects that should be rendered on the screen (see Figure 8).

- The *server* receives the input from the clients, updates the current status of the game (e.g., by considering the movements performed by the vehicles and the tasks that they complete), and generates new goals and events as needed (see Figure 9). The server is multithreaded, with a thread per vehicle that performs a

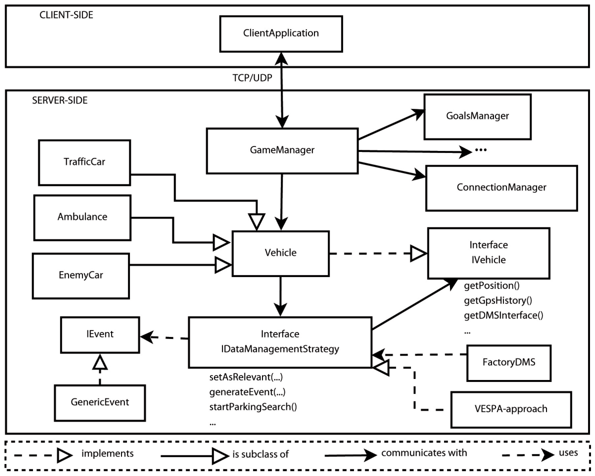


Figure 7: Basic architecture of Vanet-X

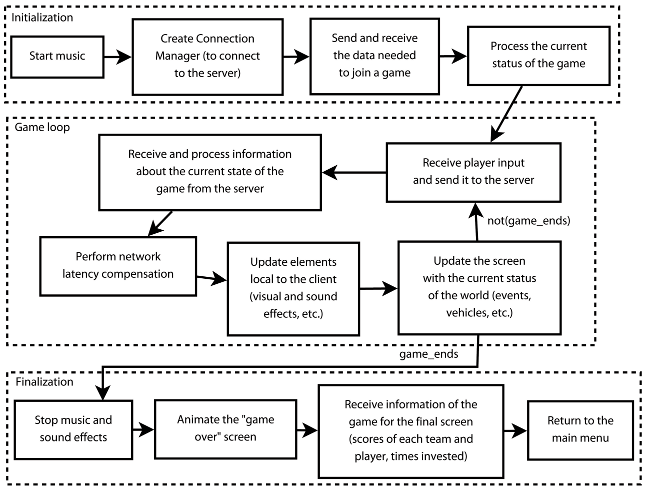

Initialization
Start music
Create Connection Manager (to connect to the server)
Send and receive the data needed to join a game
Process the current status of the game
Game loop
Receive and process information about the current state of the game from the server
Receive player input and send it to the server
not(game_ends)
Perform network latency compensation
Update elements local to the client (visual and sound effects, etc.)
Update the screen with the current status of the world (events, vehicles, etc.)
game_ends
Finalization
Stop music and sound effects
Animate the "game over" screen
Receive information of the game for the final screen (scores of each team and player, times invested)
Return to the main menu

Figure 8: Basic functioning of a client

basic cycle of "while a vehicle is alive, perform actions and check for potential collisions".

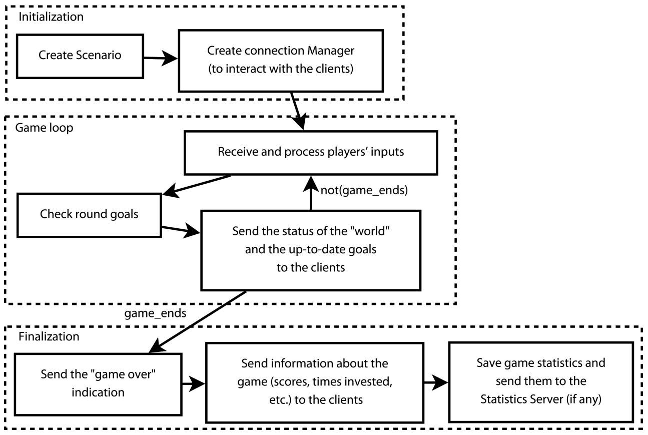

Initialization
Create Scenario
Create connection Manager (to interact with the clients)
Game loop
Receive and process players' inputs
not(game_ends)
Check round goals
Send the status of the "world" and the up-to-date goals to the clients
game_ends
Finalization
Send the "game over" indication
Send information about the game (scores, times invested, etc.) to the clients
Save game statistics and send them to the Statistics Server (if any)

Figure 9: Basic functioning of the server

- An interface *IDataManagementStrategy* declares the methods that should be implemented by a *data management strategy* to allow its integration with the videogame (e.g., a method to define the types of events that are interesting for the driver, a method to generate an event, etc.).
- Another interface *IVehicle* is implemented by the *vehicles* to allow interacting with them (e.g., to obtain a reference to the data manager in the vehicle or to obtain information about the GPS location).

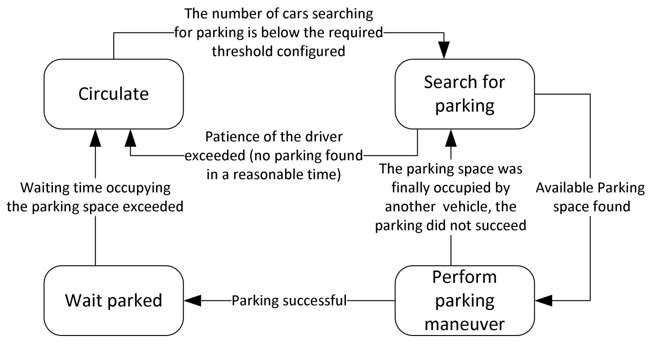
Any data management strategy can potentially be integrated in this framework, as long as it implements the interface *IDataManagementStrategy* and calls the appropriate methods to inform the vehicles (interface *IVehicle*). So, we can easily plug in different alternative data management techniques for testing.

3. BEHAVIORS OF THE VEHICLES

We have implemented several behaviors for the vehicles controlled by the computer, which adapt the steering behaviors proposed in [23]. In particular, we consider the following basic behaviors:

- *Seek* implies directing the vehicle towards a certain static target, by adjusting its direction and speed.
- *Flee* is the opposite behavior to *Seek*, as it implies getting as much further as possible from the target.
- *Pursuit* is similar to *Seek*, but in this case the target is a moving object. So, the expected movement of the target is estimated, to try to catch it.
- *Evasion* is the opposite behavior to *Pursuit* (i.e., based on *Flee* instead of *Seek*).
- *Arrival* implies the progressive reduction of speed as the vehicle approaches the target.
- *Obstacle avoidance* provides vehicles with the ability to dodge vehicles and other obstacles.
- *Wander* generates a random trajectory, to represent a vehicle traveling around with no clear objective. This is useful, for example, to represent a vehicle that is searching for an available parking space in the vicinity.
- *Path following* allows a vehicle to circulate within the boundaries of a certain path.
- *Unaligned collision avoidance* is a behavior that tries to avoid the collision of vehicles moving in different directions. Thanks to this behavior, vehicles can estimate a potential collision risk with other vehicles in the near future, to try to avoid it.

Of course, all the vehicles exhibit the whole set of behaviors at the same time, applying a priority ordering in case several behaviors could be applied at the same time and are in conflict to each other. Based on the previous basic behaviors, we have defined the schema of a normal behavior for different types of vehicles: enemy cars (that try to catch the players or flee from the players, depending on the game mode), ambulances (as representatives of emergency vehicles which may ask the right of way), and traffic cars (that represent neutral cars in the game). As an example, the basic behavior of traffic cars is shown in Figure 10.



The number of cars searching for parking is below the required threshold configured
Circulate
Search for parking
Patience of the driver exceeded (no parking found in a reasonable time)
Waiting time occupying the parking space exceeded
The parking space was finally occupied by another vehicle, the parking did not succeed
Available Parking space found
Wait parked
Parking successful
Perform parking maneuver

Figure 10: Basic behavior of a traffic car

4. DATA COLLECTION AND EXPLOITATION

In this section, we summarize the strategy applied for data collection during the game and the corresponding exploitation of results. If a certain configuration option that activates the collection of statistics during the game is enabled, several data are collected: data about the scores obtained by the players, the time needed by vehicles (the ones controlled by humans as well as those managed by the computer) to perform certain tasks (such as parking), and other measures about the performance of the data management strategy applied (e.g., events created, events that are considered relevant by each vehicle, etc.). When the game ends, all these data are stored in several files on the game server, along with a file that contains information about all the configuration parameters used in that game (e.g., game mode, configuration parameters used for the data management strategy considered, the wireless communication range simulated, etc.).

To centralize the data collected, it is possible to set up a *Statistics Server*, which is a process executing continuously on a certain computer. In this way, the clients playing the game automatically connect to the Statistics Server when a game ends, in order to communicate the statistics collected during the game. Besides, it is possible to connect to the *Master Server* by using a terminal client (called *Statistics Client*) that allows seeing and modifying the configuration parameters as well as retrieving the statistics files generated. Another option is to avoid the use of a Statistics Server and collect the statistics in the computer that plays the role of a server for a game. If we consider configuration settings where there is a predefined game server and all the clients connect to it to start a new game or join an existing game, this option also keeps the statistics in a single location. However, if there are several game servers then the statistics would have to be centralized manually.

Figure 11 provides an overview of the way the different components of the game, and particularly the Statistics Server, are distributed in a network. Notice that we actually distinguish between a *Master Server* and a *Game Server*. The Master Server is executing on the server machine and a client first connects to it (so, it is the entry point for clients); then the Master Server checks if a Game Server is available and if not it creates one; finally, it returns the port number of its Game Server to the client, as the client will interact with the Game Server during the game.

It should be noted that, as we collect information about the performance of human players, the skills of those players

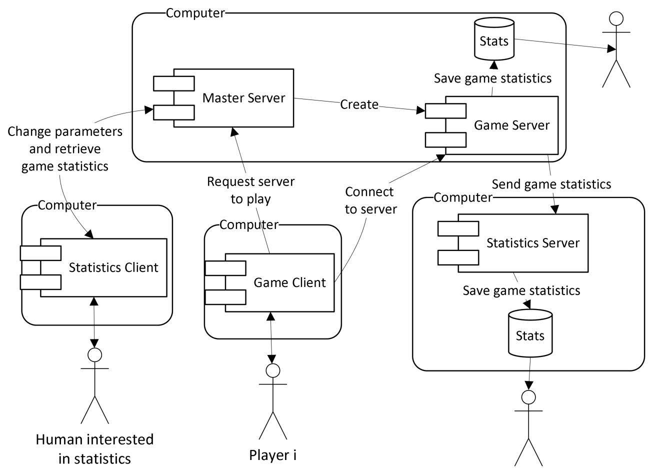


Figure 11: Deployment of components in a network

with the game will have an impact on the results and this has to be taken into account when exploiting the results. Indeed, directly comparing the achievements of several human players without considering their game skills could lead to wrong conclusions. For example, *player1* without a data sharing system could perform better than *player2* with a data sharing system, but we should not necessarily conclude that the use of such a data sharing system is harmful. In other words, we should always compare players with the same skills. For this reason, each human player is assigned a certain *skill level* (which may change along time, as the player improves his/her performance) and the statistics about players are tagged with the skill level corresponding to that player. Besides, players that have a skill level below a certain threshold are (by default) not allow to participate in games with collection of statistics enabled, as performance data about them are assumed to be unreliable and besides their clumsiness could interfere with the normal development of the game. The skill level of a player is computed based on his/her ability to complete missions in the game (tasks per time unit).

5. EXPERIMENTAL EVALUATION

We have performed a few preliminary experiments to evaluate the interest of our proposal. As a use case for testing, we focused on the case of available parking spaces, as these are events that represent scarce resources, which implies additional challenges for data management (i.e., the competition among vehicles should be minimized).

5.1 Data Management Strategies

As a data sharing strategy for the vehicles, we considered the following options.

5.1.1 VESPA-P*: VESPA With No Reservation*

First, we adapted the proposal in [2], developed in the context of the system *VESPA* (*Vehicular Event Sharing with a mobile P2P Architecture*) [4, 6], which is based on the computation of an *Encounter Probability* (*EP*).

The EP between a vehicle and an event estimates the likelihood that the vehicle will meet the event, based on geographic computations that estimate the spatio-temporal relevance of the event. For example, the relevance decreases with the distance between the event and the vehicle, with the time since the event was generated (e.g., consider the case of information about an available parking space, which can be unoccupied only for a limited amount of time), and the direction of the vehicle (e.g., if it is approaching the event or not). In particular, the directions of both the vehicle and the event are estimated and several *penalty coefficients* (α, β, γ, and ζ) are used to weigh the importance of four estimated parameters: the minimum distance to the event over time (Δd), the time until the closest position to the event (Δt), the age of the event at the closest position (Δg), and the angle between the vehicle and the event (c).

So, when a vehicle receives an event it computes its EP and disseminates the event again if the computed EP exceeds a certain *dissemination threshold* (DT). The intuition is that vehicles should disseminate data that are relevant for them (as those data are also probably relevant for the neighboring vehicles). Two other thresholds are managed: the *storage threshold* (ST) and the *relevance threshold* (RT). The ST determines the minimum value of the EP for an event to be stored locally in the vehicle, and the RT the minimum value needed to show the event to the driver.

Besides, the proposal in [2] proposes a contention-based approach for data dissemination in order to limit the network overhead in the dissemination of messages (basically, when there are several candidate vehicles to re-disseminate an event, the message will be disseminated only by the vehicle located further away from the vehicle that disseminated the message previously). Several parameters are used in the protocol, such as D (the maximum time to wait before redifusing) and D' (time to wait for an acknowledgement that a message sent previously was received by some other vehicle).

5.1.2 VESPA+P*: VESPA With Reservation Protocol*

Communicating the availability of a single parking space to many vehicles could lead to an unfruitful competition among the vehicles to try to reach the same parking space, leading to dissatisfaction of the drivers and parking times that could even exceed those that would be obtained if no data sharing system were used. For this reason, the work presented in [7] proposed an enhancement to the previous approach *VESPA-P* for the case of scarce resources such as parking spaces. It provides an allocation protocol that coordinates a procedure that ensures that the information about an available parking space is communicated to a single interested vehicle.

5.1.3 Blind*: No Data Sharing*

Finally, we also considered an approach where no data sharing strategy is used. In this case, the vehicles receive no information and the only data available for the drivers are what they see with their own eyes. For vehicles trying to find available parking spaces, this will lead to a *blind search*.

5.2 Experimental Settings

The basic configuration of the videogame for the experimental evaluation is as follows. The communication range considered for the vehicles is 200 meters and a maximum of 50% of the vehicles are assumed to be equipped with a data sharing application. The penalty coefficients used to compute the EP for VESPA are: α=1/1500 ($\Delta d \leq 500$ meters), β=1/180 ($\Delta t \leq 60$ seconds), γ=1/360 ($\Delta g \leq 120$ seconds), and ζ=1/270 ($c \leq 90°$); these are parameters that can be considered for a "medium" (not small, not large) dissemination area, according to [2]. The RT and the DT are both set to 75%, and the ST is 60%. The query processor on each vehicle re-evaluates the relevance of the events received with a refreshment period of 2 seconds, showing on the radar the events that are considered relevant. For the dissemination protocol, D is set to 1 second and D' to 2 seconds.

5.3 Experimental Results

We have simulated a varying number of vehicles moving in an area of 1 squared kilometer around the street "Sophie Oury" in the city of Valenciennes (France). In this scenario, we measured the time needed by the vehicles to find free parking spaces near certain destinations. In Figure 12 we show the reduction on the average time needed by a human player to find an available parking space near the target. The experimental results show the interest of sharing data among the vehicles (with both *VESPA-P* and *VESPA+P*), as these data can later be shown on the radar to provide interesting information to the drivers. Besides, according to these results, using a reservation protocol to avoid the competition problem (*VESPA+P*) is particularly beneficial.

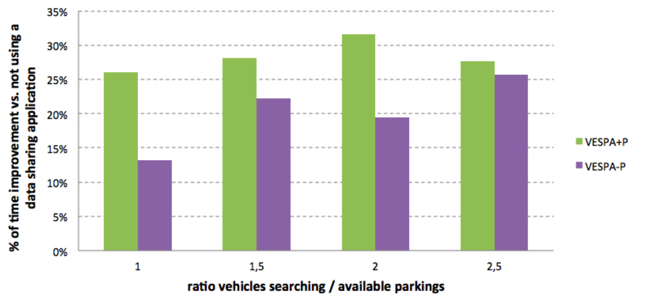


Figure 12: Time to park by a human

In Figure 13 we compare the performance of human players (vehicles controlled by humans) and computer players (vehicles controlled by the computer), by showing the reduction on the average time needed to find an available parking space near the target when using *VESPA+P*. According to these experimental results, we can see that the human players get more benefit from the use of the data sharing strategy. The difference may be due to the way the artificial intelligent behavior of the computer vehicles is implemented.

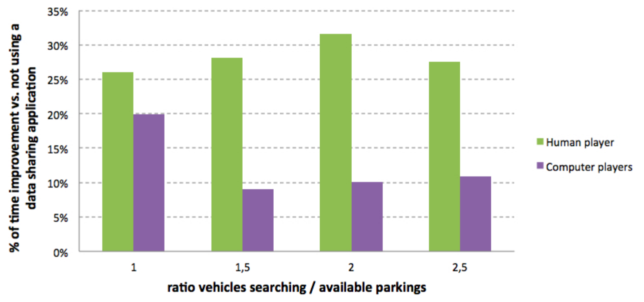


Figure 13: Time to park: human vs. computer

The experimental results obtained correspond to data collected during a total of 14 hours playing the videogame (about 400 parking actions by the human player during this game time). The results are consistent with our intuition and with other experimental results obtained previously by using a simulator. Nevertheless, more tests are needed to validate the results and evaluate other scenarios. For example, we started to obtain some first preliminary results with games played by more than one human player. It is also interesting to perform experiments with other types of events (e.g., accidents, obstacles on the roads, etc.); with information about them, drivers could try to avoid those hazards and so decrease the total travel time.

6. RELATED WORK

As far as we know, this is the first attempt to develop a videogame whose hidden purpose is to help with the evaluation of information management strategies for vehicular networks.

Nevertheless, the idea of trying to benefit from human actions to improve or evaluate a system is not new. Exploiting the power of people to perform large-scale tasks that are costly, time-expensive, or hard, is called *crowdsourcing* [33]. For example, *mCrowd* [32] benefits from sensors available on iPhone devices to perform collaborative tasks such as image tagging or road traffic monitoring. As another example, *reCAPTCHA* [29] exploits *CAPTCHAs* [22] (Completely Automated Public Turing test to tell Computers and Humans Apart), as a security measure to avoid web access to programs, in order to recognize words from scanned books that are challenging for OCR (Optical Character Recognition) systems. According to [10], "The practice of crowdsourcing is transforming the Web and giving rise to a new field".

Particularly relevant for our work with Vanet-X are those proposals that achieve the crowdsourcing results through the use of a videogame. A notable example is the *ESP game* [27], where players implicitly help to label images while playing the game. The use of videogames as learning tools is a clear example of the benefits of using educative videogames; as an example, *CodeSpells* [11] is a fantasy videogame where players have to write spells in Java. Other games with a hidden purpose exist, as commented in [28]. The multiplayer online game *Planet PI4* [16] intends to serve as a testbed environment for Peer-to-Peer (P2P) game architectures. It is also interesting to mention that the term *gamification* has appeared to denote a variety of software that is inspired somehow by videogames [8, 9].

There exist some driving videogames that, as Vanet-X, are based on the use of real road maps or city layouts, such as *Mini Maps* (`https://apps.facebook.com/minimaps/`) and *Push-Cars 2: On Europe Streets* (`http://www.push-cars.com`). However, unlike in Vanet-X, in these games the players do not contribute to any crowdsourcing task or data management strategy evaluation.

Finally, a good number of simulators of vehicular networks and mobility generators have been developed, such as *TraNS* [21], *SUMO* [19], *Veins* (Vehicles in Network Simulation) [25], *GrooveNet* [17], or *VanetMobiSim* [13]. Some interesting surveys can be found in [12, 18]. As commented along the paper, we argue that the videogame-based approach can be an interesting complement (but not a replacement) to the use of existing simulators to evaluate information management strategies for vehicular networks. Besides, mobility generators and vehicle simulators could potentially be used to generate neutral traffic for Vanet-X.

7. CONCLUSIONS AND FUTURE WORK

We have developed a videogame that can be used to evaluate data management strategies for vehicular networks, as a complement to existing simulators. Whereas the opportunity of crowdsourcing through a videogame is attractive, several challenges arise. Thus, the goal of developing a fun videogame required the introduction of several elements that would not appear in a real scenario (like enemy cars), which could have an impact on the results, but on the other hand this will attract people to play. Moreover, the results obtained can depend not only on the benefits offered by the data management strategy but also on the ability of the specific player. So, whereas the videogame can provide an ideal tool to collect many data for a variety of scenarios, the experimental results obtained have to be judged with caution (e.g., we label the collected data with the skill level of the player). Even with these limitations, we argue that the videogame helps to collect with less effort data that can be used to fine-tune a protocol and/or obtain some initial conclusions, prior to the evaluation in more realistic scenarios.

Additional information regarding the videogame is available at `http://sid.cps.unizar.es/Vanet-X/`, including a playable version of the videogame, some videos, and screenshots. This is a first step that shows the potential interest of exploiting videogames to evaluate data management strategies for vehicular networks. As future work, we would like to optimize and improve the videogame, as well as to develop a complete methodology and architecture to collect the data, evaluating the interest of the results obtained in other scenarios and in a larger scale.

8. ACKNOWLEDGMENTS

This research work is currently supported by the CICYT project TIN2010-21387-C02-02 and DGA-FSE. The data management strategy adapted and used as an example in the videogame has been proposed in the context of the VESPA project, and we would like to warmly acknowledge the collaboration with Dr. Thierry Delot in that project.

9. REFERENCES

[1] J. J. Blum, A. Eskandarian, and L. J. Hoffman. Challenges of intervehicle ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 5(4):347–351, 2004.

[2] N. Cenerario, T. Delot, and S. Ilarri. A content-based dissemination protocol for VANETs: Exploiting the encounter probability. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):771–782, 2011.

[3] T. Delot, N. Cenerario, and S. Ilarri. Vehicular event sharing with a mobile peer-to-peer architecture. *Transportation Research Part C: Emerging Technologies*, 18(4):584–598, 2010.

[4] T. Delot and S. Ilarri. Data gathering in vehicular networks: The VESPA experience (invited paper). In *Fifth IEEE Workshop On User MObility and VEhicular Networks (LCN ON-MOVE 2011)*, pages 801–808. IEEE Computer Society, 2011.

[5] T. Delot and S. Ilarri. Introduction to the Special Issue on Data Management in Vehicular Networks. *Transportation Research Part C: Emerging Technologies*, 23:1–2, 2012.

[6] T. Delot and S. Ilarri. The VESPA Project: Driving advances in data management for vehicular networks. *ERCIM News*, (94):17–18, July 2013. Special Theme on “Intelligent Vehicles as an Integral Part of Intelligent Transport Systems”.

[7] T. Delot, S. Ilarri, S. Lecomte, and N. Cenerario. Sharing with caution: Managing parking spaces in vehicular networks. *Mobile Information Systems*, 9(1):69–98, 2013.

[8] S. Deterding, D. Dixon, R. Khaled, and L. Nacke. From game design elements to gamefulness: Defining “gamification”. In *15th International Academic MindTrek Conference: Envisioning Future Media Environments (MindTrek'11)*, pages 9–15. ACM, 2011.

[9] S. Deterding, M. Sicart, L. Nacke, K. O'Hara, and D. Dixon. Gamification: Using game-design elements in non-gaming contexts. In *2011 Annual Conference on Human factors in Computing Systems (CHI'11) – Extended Abstracts*, pages 2425–2428. ACM, 2011.

[10] A. Doan, R. Ramakrishnan, and A. Y. Halevy. Crowdsourcing systems on the World-Wide Web. *Communications of the ACM*, 54(4):86–96, 2011.

[11] S. Esper, S. R. Foster, and W. G. Griswold. On the nature of fires and how to spark them when you're not there. In *44th ACM Technical Symposium on Computer Science Education (SIGCSE'13)*, pages 305–310. ACM, 2013.

[12] J. Harri, F. Filali, and C. Bonnet. Mobility models for vehicular ad hoc networks: A survey and taxonomy. *IEEE Communications Surveys & Tutorials*, 11(4):19–41, 2009.

[13] J. Härri, F. Filali, C. Bonnet, and M. Fiore. VanetMobiSim: Generating realistic mobility patterns for VANETs. In *Third International Workshop on Vehicular Ad Hoc Networks (VANET'06)*, pages 96–97. ACM, 2006.

[14] H. Hartenstein and K. P. Laberteaux. A tutorial survey on vehicular ad hoc networks. *IEEE Communications Magazine*, 46(6):164–171, 2008.

[15] G. Karagiannis, O. Altintas, E. Ekici, G. J. Heijenk, B. Jarupan, K. Lin, and T. Weil. Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions. *IEEE Communications Surveys & Tutorials*, 13(4):584–616, 2011.

[16] M. Lehn, C. Leng, R. Rehner, T. Triebel, and A. Buchmann. An online gaming testbed for peer-to-peer architectures. *ACM SIGCOMM Computer Communication Review*, 41(4):474–475, 2011.

[17] R. Mangharam, D. S. Weller, and R. Rajkumar. GrooveNet: A hybrid simulator for vehicle-to-vehicle networks. In *Second International Workshop Vehicle-to-VehicleCommunications (V2VCOM'06)*, pages 1–8, 2006.

[18] F. J. Martinez, C. K. Toh, J.-C. Cano, C. T. Calafate, and P. Manzoni. A survey and comparative study of simulators for vehicular ad hoc networks (VANETs). *Wireless Communications & Mobile Computing*, 11(7):813–828, 2011.

[19] J. E. Michael Behrisch, Laura Bieker and D. Krajzewicz. SUMO – Simulation of Urban MObility: An overview. In *The Third International Conference on Advances in System Simulation (SIMUL'11)*, pages 63–68. IARIA, 2011.

[20] S. Olariu and M. C. Weigle, editors. *Vehicular Networks: From Theory to Practice*. Chapman & Hall/CRC, 2009.

[21] M. Piorkowski, M. Raya, A. L. Lugo, P. Papadimitratos, M. Grossglauser, and J.-P. Hubaux. TraNS: Realistic joint traffic and network simulator for VANETs. *SIGMOBILE Mobile Computing and Communications Review*, 12(1):31–33, 2008.

[22] C. Pope and K. Kaur. Is it human or computer? Defending e-commerce with Captchas. *IT Professional*, 7(2):43–49, 2005.

[23] C. W. Reynolds. Steering behaviors for autonomous characters. In *Game Developers Conference*, pages 763–782. Miller Freeman Game Group, 1999.

[24] M. L. Sichitiu and M. Kihl. Inter-vehicle communication systems: A survey. *IEEE Communications Surveys & Tutorials*, 10(1–4):88–105, 2008.

[25] C. Sommer, R. German, and F. Dressler. Bidirectionally coupled network and road traffic simulation for improved IVC analysis. *IEEE Transactions on Mobile Computing*, 10(1):3–15, 2011.

[26] Y. Toor, P. Mühlethaler, A. Laouiti, and A. de La Fortelle. Vehicle ad hoc networks: Applications and related technical issues. *IEEE Communications Surveys & Tutorials*, 10(1–4):74–88, 2008.

[27] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *SIGCHI Conference on Human Factors in Computing Systems (CHI'04)*, pages 319–326. ACM, 2004.

[28] L. von Ahn and L. Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.

[29] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[30] T. L. Willke, P. Tientrakool, and N. F. Maxemchuk. A survey of inter-vehicle communication protocols and their applications. *IEEE Communications Surveys & Tutorials*, 11(2):3–20, 2009.

[31] B. Xu, A. M. Ouksel, and O. Wolfson. Opportunistic resource exchange in inter-vehicle ad-hoc networks. In *Fifth IEEE International Conference on Mobile Data Management (MDM'04)*, pages 4–12. IEEE Computer Society, 2004.

[32] T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner. mCrowd: A platform for mobile crowdsourcing. In *Seventh ACM Conference on Embedded Networked Sensor Systems (SenSys'09)*, pages 347–348. ACM, 2009.

[33] M.-C. Yuen, I. King, and K.-S. Leung. A survey of crowdsourcing systems. In *Third International Conference on Privacy, Security, Risk and Trust (PASSAT 2011) and Third International Conference on Social Computing (SocialCom 2011)*, pages 766–773. IEEE, 2011.