Content Delivery Architecture for Communication Device-to-Device Wireless Networks

Charles Tim Batista Garrocho*, Mauricio Jose da Silva† and Ricardo Augusto Rabelo Oliveira‡
Department of Computer, Federal University of Ouro Preto
Ouro Preto, MG – Brazil
Email: *ctgarrocho@gmail.com, †badriciobq@gmail.com, ‡rrabelo@gmail.com

***Abstract*—The Device to Device (D2D) communications have become essential in daily life. Current technologies characteristics preclude the transparent exchange of content among devices. To address this, an architecture that manages the Wi-Fi interface device is defined. It promotes communication between devices, allowing transparent content exchange without user intervention. Two applications that employ the use of this architecture are presented. The first one, for personal devices, proved to be scalable in tests with up to nine devices. The second one, for vehicles, proved to be feasible when applied in scenarios with low speed, causing a low packet loss and high transmission rates.**

Keywords–Wireless network architecture; Management; D2D and V2I applications.

I. INTRODUCTION

The wireless communication networks have become essential in the information society. People can connect to data networks from anywhere, through various communication devices and technologies. The vehicle is a place where users spend much of their time every day [1][2].

In recent years, mobile devices such as cellphones, smartphones and tablets are gaining popularity and evolving, making the user interaction with the device a less virtual and more realistic experience. The integration of sensors, such as Bluetooth, Wi-Fi Direct, accelerometer, compass, gyroscope, microphone, camera, Global Positioning System (GPS) and radio turned simple cell phones into powerful portable machines [3][4].

A. Wireless Network Technologies

The primary means of access to the information is through cell phone networks, that allow us to have instant access to the internet services, as long as the device is located inside a cell of an antenna [5]. However, cellular networks may be lacking or fails, in case of partial or total communication infrastructure failures caused by natural disasters [6], government censorship [7], or even by interruptions in the Internet or mobile network services [8].

Although the wireless interface technologies such as Wi-Fi ad-hoc, Wi-Fi Direct and Bluetooth offer capabilities Peer-to-Peer (P2P) for information exchange in the absence of cellular network, limitations of the protocol specification, chipsets and operating systems on mobile devices make these technologies mostly useless in practice.

Current mobile devices do not support Wi-Fi ad-hoc [9], except on devices with a rooted operating system, as in [10]. Bluetooth is limited in terms of communication distance and bandwidth as well as device discovery without human interaction [11]. In addition, the Bluetooth takes a long time for pairing and most of the attempts are unsuccessful [12]. Communication via Wi-Fi Direct is another option, but the input of a Personal Identification Number (PIN) is mandatory, which demands interaction with the user, and the group formation can take up to two minutes [13].

These characteristics of the cited technologies, especially the cell phone technology, prevent the development of applications that require transparent communication, that is, the formation of the communication network and the exchange of content without the need of user interaction with the device. To solve these problems, the use and management of the Wi-Fi interface of the devices are proposed, so as to allow a transparent communication to the user.

B. Contributions

The main contribution of this paper is a content distribution architecture, where devices can become a wireless access point, or a client connected to a network provided by another device. The architecture manages the Wi-Fi interface, forms the communication network and enables data transmission transparently to the user.

As a proof of concept, the following applications have been developed:

- The first application, *Crowd Wi-Fi*, allows the exchange of information among mobile devices transparently, allowing its use in events with agglomerations of people, such as restaurants or museums;
- The second application, *Black Box*, allows the transparent exchange of information between vehicles and infrastructures installed in parking lots, which allows its use in trucks or bus fleet companies.

The results showed that the *Crowd Wi-Fi* application was able to transparently distribute content from one device to several others simultaneously. The system proved scalable, simultaneously transmitting content to 8 devices at an average transmission rate of 17Mbps.

Just like in the Wi-Fi implementation, the results of the experiments with the *Black Box* application were also encouraging. The application behaved well for data transfer between a vehicle and an infrastructure. At a distance of up to 30 meters, the system was able to deliver an average transfer rate of 500kbps, a packet loss rate of 25% and an average delay of 30ms.

The rest of the paper is organized as follows: in Section II, related works are presented. In Section III, an overview of the architecture is presented. In Section IV, the *Crowd Wi-Fi* application is presented. In Section V, the *Black Box* application is presented. In Section VI, the scenarios and metrics used for evaluation are presented. In Section VII, the results of the experiments are presented. Finally, in Section VIII, the conclusions are presented.

 ISBN: 978-1-61208-411-4

II. Related Works

The emergence and ripening of P2P content distribution significantly reduced dependence on content delivery in content distribution networks as well as bottlenecks between consumers and content providers. A lot of research regarding P2P content distribution networks has been done so far, but little has been researched on the application of P2P content distribution in wireless networks [14].

Some studies aim to optimize the latency of the response time and power consumption of the devices in wireless content distribution networks by caching the content. Boscovic et al. [15] points out that Internet access via mobile devices is increasing, and that caching content among devices can increase the availability of content and decrease the response time when accessing data.

In [16] and [17], the similarities of video content requests by mobile phone users are pointed out. The proposal is to cache the content of popular video files in smartphones and to explore the D2D communication to transmit popular videos, thus avoiding requests to the Base Station (Fixed Mobile Phone Service Station). The authors claim that their proposal improves the video transfer rate by one or two orders of magnitude.

Sharma et al. [18] developed an architecture as well as demo applications to provide communication among mobile devices in the absence or ineffectiveness of cellular infrastructure. It is presumed that at least one mobile device has cellular data connectivity, and this connectivity is shared among all devices through a mobile ad hoc network.

The work developed in [18] is the one with more similarities to the proposal of this article. Its architecture is divided into three layers: Mobile Ad Hoc Network (MANETs), Middleware Content Centric Networking (CCN) and Delay Tolerant Networking (DTN), and applications. These layers mainly enable a network abstraction to the applications. As for the proposal described in this article, it provides a description of an architecture composed of several modules. Although the architecture does not abstract the network layer, as was done in [18], applications developed with the architecture of this article allows a better use of the Wi-Fi interface functions, since it communicates directly without relying on a layer. Moreover, architecture capabilities are implemented, specified and evaluated in real network scenarios consisting of personal devices and vehicles, while most other studies only rely on simulations [15][16][17].

III. Architecture Overview

The architecture incorporates a collection of devices that, together, enable the formation and management of the content distribution network. Figure 1 illustrates the components of the architecture, which is divided into four modules. Each module has its particularity and a special function. In this architecture, a device that is a wireless access point will be called *Leader*, and the devices that connect to it will be called *Client*.

The *Main* module is the first module to be initialized in the architecture. It is responsible for running and managing the three other modules. Its first operation is to run the *Manager* module.

The *Manager* module can perform two distinct operations. The first operation consists of scanning wireless access points and, if any are found, establishing connections to them. If the connection is established, the *Slave* module is run by the *Main* module. The second operation consists in configuring the wireless access point on the device. Once the access point is configured, the *Master* module is executed by the *Main* module. The Service Set Identification (SSID) and password of the network are constant, so all devices know which wireless network they must search and connect to.

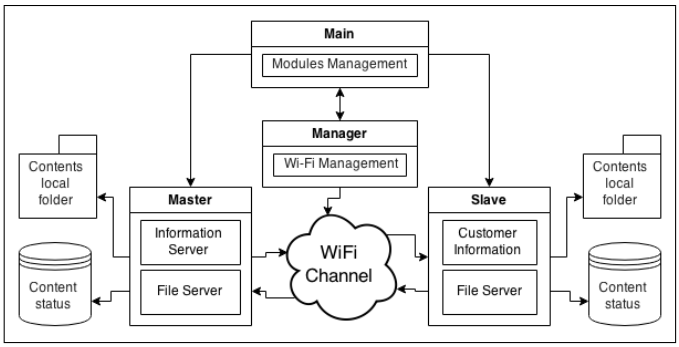


Figure 1. Schematic model of the Architecture.

The *Master* module runs exclusively on a *Leader* device, and it has information about all devices connected to it. For this reason, this module must provide information on the state of the network through an information server. On the other hand, the *Slave* module runs exclusively on a *Client* device that is connected to the *Leader*. The *Slave* module requests information about the communication network to the *Master* module running in the *Leader* device.

The contents exchanged among devices connected to the information network are transmitted through a content server on the *Master* and *Slave* modules. This server must be multitasking in order to support multiple simultaneous requests from *Clients* or *Leader* devices.

A *Leader* device simultaneously runs the *Main*, the *Manager* and the *Master* modules, while the *Client* device simultaneously runs the *Main*, the *Manager* and the *Slave* modules. In the developed applications the architecture can be used in two different ways: in the first way, one device is defined as *Leader* and all other devices as *Client*. In the second way, devices take turns acting as *Leader* and *Client*.

IV. Crowd Wi-Fi: Transparent Content Distribution Among Personal Mobile Devices

This application aims to transparently distribute content among multiple personal mobile devices in a scalable way. An example of situation where it could be used are events where there are concentrations of people, like restaurants or museums. The application was developed using the Android 4.1 operating system.

The *Crowd Wi-Fi* is divided into four modules (Table I) that follow the characteristics of the architecture. The four modules run in the background and are not affected by other applications running on the foreground on the device.

The *Main* module is the first and only Activity of the application, which takes care of the communication among modules and the management of the user interface. The *Manager* module is responsible for the device's wireless interface. The Android's Wi-Fi manager class is used both for scanning

 ISBN: 978-1-61208-411-4

TABLE I. CROWD WI-FI APPLICATION MODULES.

Module	Type	Operations
Main	Activity	Runs and manages the other modules. It's also responsible for the user interaction with the application.
Slave	AsyncTask	Requests information regarding the network to the *Master*. Provides and requests files.
Master	AsyncTask	Provides information about the state of the communication network. Provides and requests files.
Manager	Service	Manages the Wi-Fi interface and defines whether the device will be *Leader* or *Client* when communicating with the *Main*.

wireless networks and turning the device into a wireless access point.

The application's *Manager* module considers the battery level of the device for setting the duration of the the scanning for wireless networks. The lower the battery level is, the longer the scanning will be. The higher the battery level is, the shorter the scanning will be and more likely will the device become a wireless access point. The battery level is not only used to define the duration of the scanning but also when the communication network is already established. When the communication network is formed, the *Slave* module sends the device's current battery level to the the *Master* module running in the *Leader* device so it can decide which device will be the next access point should the network be destroyed (Figure 2).

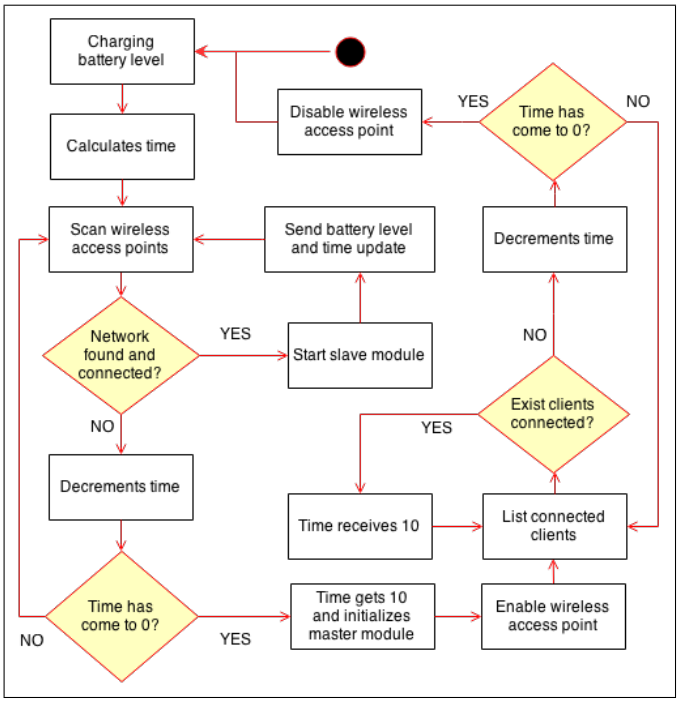


Figure 2. Activity diagram *Manager* module of the application *Crowd Wi-Fi*.

If the device becomes a *Leader*, the *Master* module is executed. This module is a AsyncTask that is triggered by the *Main* module. It is a multi threaded server that enables simultaneous connections from the *Slave* modules of the *Client* devices. The *Master* module handles three types of requests.

To make a file available on the network, the Client's *Slave* module gets the file's address and its name and builds a PUT request, sending it to the *Leader* device. The *Leader*'s *Master* module receives the Client's request and updates its local list of files available on the network as well as verifies if this new file exists on its local folder. If it does not, the *Master* module triggers a task requesting the new file to the Client's *Slave* module.

When there are no files to be made available, the *Slave* module of the *Client* devices performs two other requests to the *Leader* device, which are the LIST and the AP. In the LIST request the *Leader*'s *Master* module must respectively return lists of all the files available on the network, all the Internet Protocol (IP) addresses of *Client* devices currently connected and, finally, all device's battery levels. The *Leader*'s *Master* module sends an answer to the Client's *Slave* module containing the requested information. The Client's *Slave* module validates the list of files available on the network, and, for each file it does not have in its local folder, it triggers a task to request the file to all the devices existing in the IPs list.

After the LIST request, the *Slave* module of the *Client* device proceeds to the AP request. In this request, the *Client* device sends a message containing the device ID, its Media Access Control (MAC) address, and its current battery level. The *Leader*'s *Master* module receives this request and adds or updates this information in its local list of connected *Client* devices' battery states.

The *Master* and *Slave* modules also contain a file server. The server accepts a range of connections requesting for files and, for each request, it answers if the file exists or not. The requesting device receives the answer and if the file exists on the requested device its transmission is initialized, otherwise the connection is closed and the requesting device opens a new connection with another device, using its list of available IPs.

V. BLACK BOX: VIDEOS RECORDING MANAGEMENT AND DELIVERY IN VEHICLES

This application was developed for a truck fleet company. Each vehicle is equipped with a PandaBoardES card and a camera that monitors the driver. The main goal of this system is to record video and transmit it in chunks to a server of the truck company.

This application is divided into five modules (Table II) that follow the characteristics of the architecture presented here. The five modules run in the background, both in the vehicle and in the infrastructure.

TABLE II. BLACK BOX APPLICATION MODULES.

Module	Type	Operations
Main	ShellScript	It's the first to initialize and manages the other modules.
Camera	Python	Records videos with a specified duration and manages the available disk space.
Slave	Java	Requests for information and transmits videos to the infrastructure.
Master	Java	Offers a list of videos related to the requesting vehicle and receives videos of vehicles.
Manager	ShellScript	In the vehicle, it scans and connects to the network of the infrastructure. In the infrastructure, it configures the wireless access point for the vehicles to connect.

The *Main* module is the first to be initialized by the application and it's responsible for running and managing the five other modules. It's executed every minute by the Unix crontab tool and checks whether the other modules are running, initializing the ones that are not. To determine if a particular module is running, the application uses the Unix ps

 ISBN: 978-1-61208-411-4

tool together with the grep command with the module name to filter the results. If the operation returns nothing, the module is not running and then it is initialized.

A. Vehicle

The vehicle is only a *Client* device. It scans wireless access points and establishes connections to the ones it finds. The *Main* module of the vehicle runs and monitors the *Camera*, *Slave* and *Manager* modules.

The *Camera* module uses the camera installed in the vehicles to record videos and also manages the available disk space. It is divided into two threads. The first thread records videos from time to time in a folder. The length of each section of video is defined in a configuration file. The names of the video files are defined using the current system date and time. The second thread uses the Unix psutil tool to manage the available disk space. The maximum disk space to be used is defined in a configuration file, and if the limit is reached, the oldest video is removed from the folder. This second step is executed every minute.

The vehicle *Manager* module performs a sequence of operations, as shown in Figure 3. It uses the WPA supplicant tool to scan the wireless access points in order to verify if the vehicle is within a given cell. In order to perform the scanning, the SSID, password and Wi-Fi Protected Access (WPA) security type of the networks to be found are loaded from a configuration file. If a wireless network is found, the *Manager* module connects to it and uses the ping command to verify if the connection was established with the *Leader*. If the ping returns an error, the vehicle can not communicate with the *Leader*, in which case it runs the dhclient command on the wireless interface. The dhclient command uses the Dynamic Host Configuration Protocol (DHCP) protocol to obtain an IP address from the *Leader* and uses it to configure the wireless interface. If the ping check is successful, the vehicle checks whether the *Slave* module is running and, if not, it is initialized and its Process Identification (PID) stored, so it's not necessary to rerun it on the next interaction. The *Manager* module also monitors the *Slave* module, so it runs only when the ping test succeeds. If the ping check fails and the *Slave* module is running, it is then terminated through its PID.

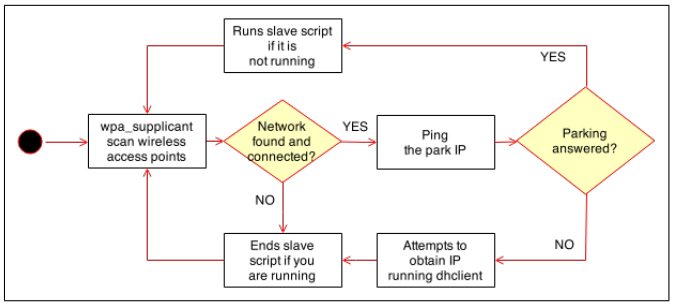


Figure 3. Activity diagram *Manager* module of the application *Black Box*.

The *Slave* module is a Java implemented client and its role is to care for the communication and management of local videos of vehicles. This module communicates with the *Master* module of the infrastructure and requests a list of videos of the vehicle. It receives the list and transmits the existing videos to the infrastructure.

B. Infrastructure

The infrastructure is nothing else than a *Leader* device, ie, a wireless access point. In the first step, an iscp-dhcp-server is set in the infrastructure's wireless card. This server manages the IP addresses of the vehicles that establish a connection to the infrastructure's wireless card. The IP address ranges and the Wi-Fi interface to be used are configured on the server.

The *Main* module of the infrastructure executes and monitors the *Master* and *Manager* modules. Unlike what occurs in the vehicle, the *Manager* module only configures the wireless access point on the infrastructure. In order to do this, it uses the hostapd tool that loads a SSID, a password and a type of WPA from a configuration file.

A sequence of message exchanges occurs between the *Master* and *Slave* modules in order to transmit the videos. The message exchange process is illustrated in Figure 4.

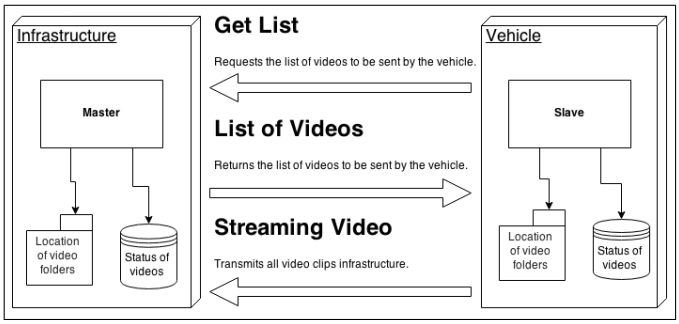


Figure 4. Communication between *Slave* and *Master* modules.

In the infrastructure, the *Master* module is a server implemented in Java and manages the videos of the vehicle. A list of videos is stored locally and contains information about the status of the videos. The infrastructure receives the list of videos of vehicles to be transmitted from the fleet company server. When a vehicle establishes connection to the Wi-Fi network of the infrastructure, the *Slave* module requests to the *Master* module a list of videos to be transmitted. The *Master* module uses a vehicle identifier to filter the videos related to it. Then, a list of videos is sent to the vehicle. The *Slave* module receives the list of videos and transmits all the video files that are requested. The *Master* module on the infrastructure receives the videos, change the videos' status and transmit them to the fleet company's server over the Internet.

VI. Evaluated Scenarios and Metrics

In this section the scenarios and metrics evaluated in the experiments are presented.

A. Crowd Wi-Fi Application

The experiments were conducted in the laboratory, in a controlled scenario in which the devices were on a table, and therefore not in movement. 9 tablet devices were used and the application was modified so that all devices requested the same 9MB file. The experiments were performed 30 times on each device. The main objective is to evaluate how the network behaves when a single device simultaneously transmits the same file to multiple devices in varying quantities.

The evaluation was performed by measuring the necessary time for the formation of a topology where the devices could communicate, the packet delay time, the packet loss rate and

 ISBN: 978-1-61208-411-4

the transmission rate. In all the experiments, the applications were executed all at once in the 9 tablets. The first experiment measured the average time spent by all the devices to establish a connection to the *Leader* and form the network topology. For the transmission delay, it was measured the time lapse between the transmission of the file and its reception on the destination. Regarding the packet loss rate, it was compared the number of packets transmitted to the number of packets actually received. Data were obtained through calculations performed on the application itself.

B. Black Box Application

The experiments were conducted on a 430 meters avenue located in the Federal University of Ouro Preto (Figure 5). A vehicle started moving from one end of the avenue (point 2), keeping the speeds of 60 km/h, 50 km/h, 40 km/h, 30 km/h and 20 km/h. The vehicle in point 1 acted as the infrastructure, standing still in the middle of the avenue. The distance between the two points was 216 meters.

Figure 5. Aerial view of the experiment region.

The evaluation of the network was performed by measuring the delay time, the loss rate and transmission rate of the packets. For the delay time it was measured the lapse between the time the packet was transmitted and the time it reached the receiver. As for the packet loss rate the number of packets transmitted was compared to the number of packets actually received. The data were obtained using the bwping software, which fired 512 bytes packets in a 2048 kbps transmission rate. Each experiment was conducted four times. The geographical positions of the vehicles were registered during the experiments.

VII. Results

In this section the results of the applications are presented.

A. Crowd Wi-Fi Application

The first experiment measured the time for devices to associate. The time was obtained through the Android application log. As stated in previous sections, 30 repetitions of this experiment were done. This experiment is important to evaluate the impact that the amount of devices has in the association time of the devices.

It can be observed in Figure 6a that when there are only a few devices, the formation time of the topology and its error rate are considerably larger. However, when the number of devices starts to increase the time to form the topology starts decreasing together with the error rate. Thus, it can be concluded that the topology formation behaves better in environments with larger numbers of devices, which makes it suitable for places with high concentrations of people, like restaurants, for instance.

The second experiment measured the packet delay time between the network communication devices. All devices stored the time when the packets were transmitted and the time when they were received in the destination. At the end, all stored times were collected and the average delay time was calculated. As stated in previous sections, 30 repetitions of this experiment were done. This experiment is important to evaluate the impact that the amount of devices has in the delay time of the packets.

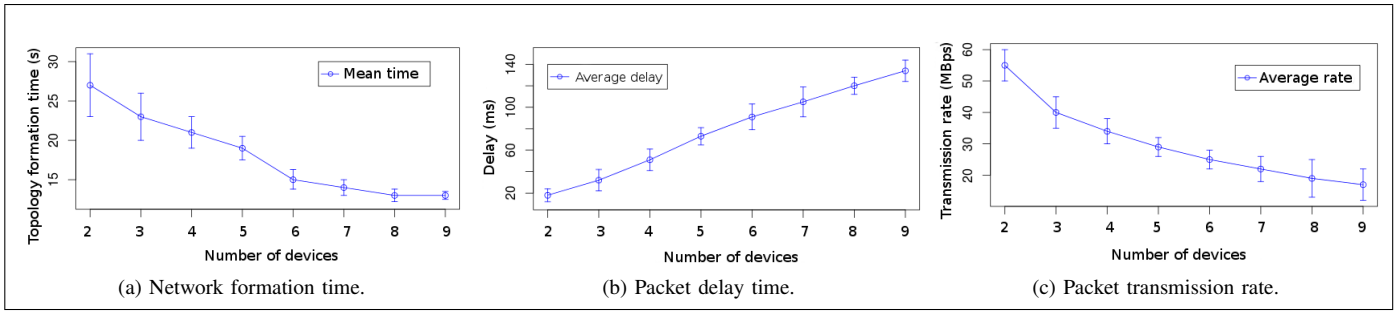
It can be observed in Figure 6b that the packet delay time increases as the number of devices that receive a file also increases. This happens because the device that transmits the file has more work to do as the communication channel is busier with more packages to be transmitted and processed at the same time.

The third experiment measured the packet transmission rate among devices on the communication network. The same process used for packet delay time was used in this experiment, but in this case, through the ages and the size of the files, it was possible to calculate the packet transmission rate.

It can be observed in Figure 6c that, like in the packet delay time, the number of devices also influences the transmission rate. The packet transmission rate decreases as the number of devices that receive the file increases. This happens because the file server on the device has more work to do as the connection bandwidth of this device is busier with multiple simultaneous connections and more packets to be processed.

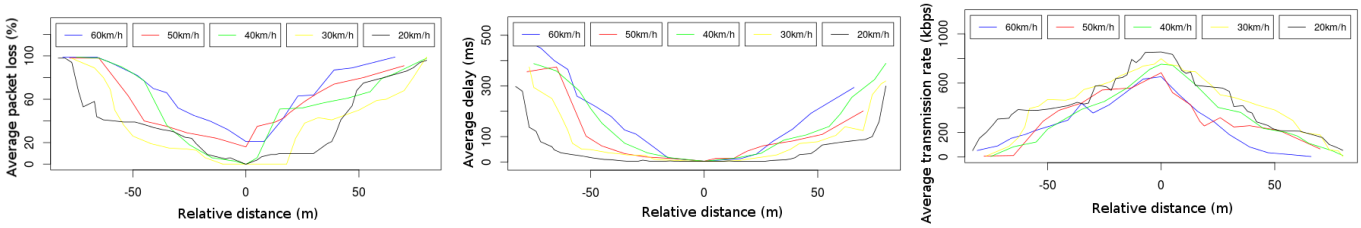
B. Black Box Application

The results were obtained from four repetitions for each experiment, and the scenario used is the one presented in Figure 5. The considered confidence interval was 95%, but it's not represented in the graph to facilitate the presentation of the information. All three experiments were evaluated at speeds of 60 km/h, 50 km/h, 40 km/h, 30 km/h and 20 km/h.



(a) Network formation time. (b) Packet delay time. (c) Packet transmission rate.

Figure 6. The results of the *Crowd Wi-Fi* application experiments.



(a) Average packet loss. (b) Average delay. (c) Average transmission rate.

Figure 7. The results of the *Black Box* application experiments.

In the graphs, the negative distance refers to the approach of the vehicle to the destination node and the positive distance refers to its distanciation.

The first experiment evaluated the packet loss rate. Figure 7a shows the loss rate. The data obtained at different speeds shows that the network behaves more robustly at lower speeds. It was possible to perform the transmission in a diameter of approximately 85 meters. The closer the vehicle is to the receiving node, the lower is the packet loss. When the nodes are at a relative distance up to 25 meters, packet loss was below 25%. The speed also impacted the packet loss, but not as much as the distance.

The second experiment evaluated the delay in the packet transmission. Figure 7b shows the delays. The delay was measured considering only the packets actually transmitted. The average delay was significantly different when considering the distance. The values obtained when the nodes were at distant points varied widely with respect to the delay obtained when the nodes were close. It was noticed that by increasing the speed, the delay in communication also suffers increase.

The third experiment evaluated the data transmission rate. Figure 7c shows the rates obtained in the communications. Data from the five experiments at different speeds showed that the average transmission rate varied over the distance.

VIII. Conclusion and Future Work

The content distribution architecture proposed in this paper successfully allowed the transparent communication in both applications developed within the Wi-Fi, showing its viability in both personal mobile devices and vehicles.

The results showed that the *Crowd Wi-Fi* application could achieve a low device association time and could also be scalable considering up to 9 devices in a communication network composed of mobile devices. Therefore, the application is feasible to be used in public places such as a restaurant, museum, or at an event where people can access content without the need for a data transmission technology.

Regarding the *Black Box* application, the results showed that below 30 km/h and at a maximum distance of 30 meters from the infrastructure, vehicles can communicate with a high transmission rate and low packet loss, making it feasible to be used for bus or truck flee companies.

In future works, besides the improvement of the applications, we intend to extend the studies, providing the *Crowd Wi-Fi* application for use in an event as well as installing the black box in a truck fleet company, with the purpose to deeply evaluate the behavior of the applications in production environments.

References

[1] P. Papadimitratos, A. L. Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular Communication Systems: Enabling Technologies, Applications, and Future Outlook on Intelligent Transportation", Communications Magazine IEEE, vol. 47, pp. 84–95, 2009.

[2] K. Dar, M. Bakhouya, J. Gaber, M. Wack, and P. Lorenz, "Wireless Communication Technologies for ITS Applications", Communications Magazine IEEE, vol. 48, pp. 156–162, 2010.

[3] N. D. Lane, E. Miluzzo, Hong Lu, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing", Communications Magazine IEEE, vol. 48, pp. 140–150, 2010.

[4] R. K. Ganti, Fan Ye, and Hui Lei, "Mobile crowdsensing: current state and future challenges", Communications Magazine IEEE, vol. 49, pp. 32–39, 2011.

[5] P. Datta and S. Kaushal, "Exploration and comparison of different 4G technologies implementations: A survey", Engineering and Computational Sciences, 2014, pp. 1–6.

[6] M. Dekker and C. Karsberg, "Annual Incident Reports 2013", Technical Report October, ENISA, 2013.

[7] M. Helft and D. Barboza, "Google Shuts China Site in Dispute over Censorship", The New York Times, 22 March, 2010.

[8] T. M. Chen, "Governments and the executive 'internet kill switch"', IEEE Netw, 25 (2), 2011, pp. 2–3.

[9] IEEE Group Std, "IEEE 802.11: Wireless LAN Medium Access Control and Physical Layer Specifications", IEEE Std. 802.11, 2007.

[10] O. R. Helgason, E. A. Yavuz, S. T. Kouyoumdjieva, L. Pajevic, and G. Karlsson, "A Mobile Peer-to-Peer System for Opportunistic Content-Centric Networking", Proc. of the ACM workshop on Networking, 2010, pp. 21–26.

[11] J. C. Haartsen, "The Bluetooth radio system", IEEE Personal Communications, 2000, pp. 28–36.

[12] A. K. Pietilainen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, "MobiClique: middleware for mobile social networking", Proc. of the ACM workshop on Online social networks, 2009, pp. 49–54.

[13] Wi-Fi Alliance P2P Technical Group, "The Wi-Fi Peer-to-Peer (P2P) Technical Specification v1.0", 2009.

[14] Jin Li, "On peer-to-peer (P2P) content delivery", Peer-to-Peer Networking and Applications, 2008, pp. 45–63.

[15] D. Boscovic, F. Vakil, S. Dautovic, and M. Tosic, "Pervasive wireless CDN for greening video streaming to mobile devices", MIPRO, Proc. of the 34th International Convention, 2011, pp. 629–636.

[16] N. Golrezaei, A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks", Communications (ICC), IEEE International Conference on, 2012, pp. 7077–7081.

[17] G. R. Hiertz, D. Denteneer, L. Stibor, Y. Zang, X. P. Costa, and B. Walke, "Device-to-device collaboration through distributed storage", IEEE Global Communications Conference, vol. 48, pp. 2397–2402, 2012.

[18] P. Sharma, et al., "Content and Host-Centric Information Dissemination in Delay-Tolerant Smartphone MANETs: An Architecture and Demonstration", Network Operations and Management Symposium, 2012, pp. 586–589.

 ISBN: 978-1-61208-411-4