ARIMA épisciences

Building a publish/subscribe information dissemination platform for hybrid mobile ad-hoc social networks over android devices

Martin Xavier TCHEMBE[*1] **and Maurice TCHOUPE TCHENDJI**[1,2]

[1]Department of Mathematics and Computer Science, University of Dschang, Cameroon
[2]FUCHSIA Team, LIRIMA

*E-mail : martinxaviertchembe@gmail.com

DOI : 10.46298/arima.9136
Submitted on February 25, 2022 - Published on November 23, 2022
Volume : **36** - Year : **2022**
Special Issue : **Special issue CRI 2021**
Editors : *Eric Badouel, Maurice Tchuenté, Paulin Melatagia, René Ndoundam*

Abstract
Mobile ad-hoc social networks (MASNs) have been the subject of several research studies over the past two decades. They allow stations located in a small geographical area to be connected without the need for a network infrastructure and offer them the possibility to communicate any time anywhere. To communicate, stations regularly broadcast their interests in the form of keywords. Stations with a high degree of similarity among their keywords can communicate with each other. However, the coverage of MASNs is limited to a small geographical area, due to the limited communication range of mobile ad-hoc networks (MANET) stations. In this paper, we present an architecture and implementation of hybrid mobile ad-hoc social networks (MASNs coupled to infrastructure networks) of Android mobile devices for information dissemination. Stations can use the infrastructure network to communicate and rely on the mobile ad-hoc network when the infrastructure is not available. Rather than communicating synchronously as this is the case in the similar works found in the literature, in our approach, the stations communicate using a publish/subscribe communication protocol, which is perfectly suited to this type of network thanks to the decoupling in time and space it provides.

Keywords
Hybrid Mobile Ad-hoc Social Networks, Publish/subscribe, Topics, Information dissemination, Nearby Connections

---

I INTRODUCTION

Online social networks such as Facebook, Twitter and LinkedIn have become very popular nowadays and have indeed become part of our everyday lives. These social networks accessible via the Internet allow people from different backgrounds to connect. However, they have their limitations when it comes to connecting people who are physically close and only need temporary social connections. Indeed, these social networks are accessible through an Internet connection, but people who are physically close, because of their spatial proximity to each other, do not need to use this remote connection, especially as it may be unavailable for various reasons (high cost of internet access, lack of network infrastructure, etc.).

Mobile Ad-hoc Social Networks (MASNs) are combination of Social Networks and Mobile Ad-hoc Networks (MANETs). They are perfectly suited to manage connectivity among individuals who are geographically close to each other. The use of these networks is made possible by increasingly sophisticated and inexpensive mobile devices. Most of these devices are equipped with multiple wireless interfaces such as Bluetooth and Wi-Fi, allowing them to communicate in an ad-hoc manner, without the need for any pre-existing network infrastructure.

MASNs can be used to discover friends and share contents in social locations such as travel stations, football stadiums, hospitals, school and university campuses, hypermarkets, fairs, etc. They can be used in scientific conferences for example to allow participants to communicate and exchange information such as personal notes, the conference program, or even important information given by the moderators (in case a participant arrived late or was absent for a while for example). Many other application areas of MASNs can be found in [17]. The contributions of this paper are the following:

Architecture for building MASNs. We propose an architecture for building MASNs coupled with infrastructure networks for information dissemination. Coupling MASNs with infrastructure networks has the advantage not only to extend the range of MASNs but also to extend the coverage of the infrastructure network as stated by Mohammad Al Mojamed in [15]. This enables stations to communicate via the infrastructure network or via the mobile ad-hoc network if they do not have access to the infrastructure. The communication model we consider is the topic-based publish/subscribe communication model. The use of a publish/subscribe communication model allows targeted dissemination of information. Indeed, this communication model suggests that each station subscribe to a certain number of topics, thus defining its interests and hence the types of information it wish to receive. Stations can then be notified when information belonging to one of their interests is available.

Implementation of a MASN over Android devices. We also propose in this paper an implementation of a MASN based on our approach, by prototyping an Android application for the dissemination of information among participants of a scientific conference. We used a software architecture designed for topic-based publish/subscribe communication among mobile devices and the Google's Nearby Connections API [19] to handle device-to-device connectivity among these devices.

The rest of this paper is organised as follows: section II presents similar works found in the literature, section III briefly presents the information dissemination protocol we used; we present the architecture and the implementation we propose in sections IV and V; the results we obtained are presented and discussed in section VI; section VII concludes this paper.

II RELATED WORKS

The issue of building MASNs on top of android devices has been addressed by several authors. Among the contributions found in the literature, efforts have been made to provide either an architecture to facilitate the development of mobile social computing applications, and/or to provide prototypes of such applications.

In the following works, authors proposed architectures to build a mobile ad-hoc network of mobile devices. In [7], authors proposed a mechanism to build an ad-hoc social network by creating user profiles automatically based on collected data from a mobile device. In [13], authors presented an architecture and implementation of social networks on commercially available mobile devices, that allow broadcasting name and a limited number of keywords representing users' interests, without any connection, in a nearby region. The application connects users to form a group based on their profile or interests using the peer-to-peer communication mode without using any centralized networking or profile-matching infrastructure.

Middlewares and/or prototypes applications have been proposed in the following works. In [2], authors presented MobiSoC, a middleware that enables mobile social computing applications development and provides a common platform for capturing, managing, and sharing the social state of physical communities. In [6], authors designed and constructed a multi-hop networking system called MoNet based on WiFi. On top of that, they Implemented a privacy-aware geosocial networking service called WiFace. They also designed a distributed content sharing protocol in order to overcome the unavailability of online services. In [10], authors proposed a framework to rapidly build local mobile ad-hoc social networks on top of the Android platform. They used it to build prototype application which shows its flexibility to expedite various services. In [11], authors presented a middleware architecture for MASNs that provides software developers a platform for developing mobile apps that enable social connections.

Efforts have been made in the literature to address the issue of integrating MASNs with infrastructure networks. In [3], the authors present the challenges related to this integration and review the strategies used in the literature. The solutions proposed in [15], [4] and [12] consist in extending a routing protocol used in MANETs to provide bi-directional connectivity between ad-hoc stations and hosts in infrastructure networks. The common principle of these solutions is to use a set of ad-hoc hosts known as Gateways to act as bridges between the two networks. In [18], the authors propose a secure publish/subscribe system based on smart contract in autonomous vehicle networks using vehicles which are part of a truck platoon system as brokers of the publish/subscribe system. A cloud server is used to establish a contract between subscribers and brokers.

The authors in [5] propose an information dissemination protocol based on content-based publish/subscribe in a hybrid VANET. The protocol uses the infrastructure (consisting of a set of fixed information stations distributed in the city and able to communicate with vehicles via Wi-fi) if existing, and vehicle-to-vehicle communications. The approach presented by authors is highly dependent on GPS, navigation system, digital city map, etc. and requires very sophisticated vehicles equipped with these technologies. This is the observation made in [8] where the authors also present a publish/subscribe system for hybrid VANET environments consisting of fixed information stations and mobile vehicles. In their approach, communication among vehicles and from vehicle to information stations is done by flooding, which can be a real bottleneck and lead to heavy power losses for network stations. Moreover, while vehicles are not greatly affected by the limited energy constraint characteristic of MANETs in general, many devices such as mobile phones that could constitute a MANET network still suffer from this limitation, making it difficult to generalise this solution to MASNs.

III INFORMATION DISSEMINATION OVER MANET STATIONS: SOCIALMANET

This section gives a brief description of SocialMANET, a protocol for information dissemination over MANET. For more information on it, interested reader should rely on [14].

SocialMANET is an information dissemination protocol using a topic-based publish/subscribe communication model in MANETs; it is subdivided into three sub-protocols which are: (1) the subscription sub-protocol, (2) the publication sub-protocol and (3) the dissemination sub-protocol. The first two give a hint of what they consist of by their rather indicative names and are relatively simple. The third sub-protocol is the core of this protocol and allows for the dissemination of publications from one network station to other stations based on their interests and/or altruistic nature. It consists of two phases carried out by all stations in the network independently of each other: the *needs detection phase* and the *publications dissemination phase*.

SocialMANET stations periodically broadcast an announcement message named *QUERY* (format illustrated in Figure 1) during the *needs detection phase*. This message contains the identifiers of the topics they have subscribed to, and for each of them the identifiers of the publications they have on these topics. The reception of this message triggers the *publications dissemination phase* at the receiving stations. During this phase, the information contained in the *QUERY* message is used by the receiving stations to calculate which publications they can provide to the sending station, according to their common interests in terms of topics they have subscribed to. Once this calculation has been made, and if they find that they store at least one such publication, they broadcast the publications they own according to the recent calculation. When a station receives a publication on a topic it has subscribed to and does not yet own, it stores it and is therefore able to broadcast it as well.

```
ID
[
    TOPIC-1 (KEY-1-1, KEY-1-2, ..., KEY-1-N1)
    TOPIC-2 (KEY-2-1, KEY-2-2, ..., KEY-2-N2)
     ...
    TOPIC-M (KEY-M-1, KEY-M-2, ..., KEY-M-NM)
]
```

Figure 1: QUERY announcement message format

IV ENABLING COMMUNICATION WITHIN A HYBRID MASN

As stated earlier, hybrid MASNs are MASNs coupled to infrastructure networks. In order to enable hybrid MASNs' stations to communicate in an ad-hoc way, we made and used an extension of the protocol for information dissemination in MANET called SocialMANET. This extension aims to allow the SocialMANET protocol to be executed in the network structure we are considering, i.e, made of an infrastructure network in which we have fixed stations that can be contacted by mobile stations in a mobile ad-hoc network and communicate with them (Fig. 2). To do that, we extended the dissemination sub-protocol of the SocialMANET protocol by introducing opportunistic exchanges[1] between the mobile stations of the MANET and the fixed stations of the infrastructure network.

[1] Opportunistic exchanges are initiated by mobile stations when they discover that they have access to the infrastructure network.

4.1 Network Architecture

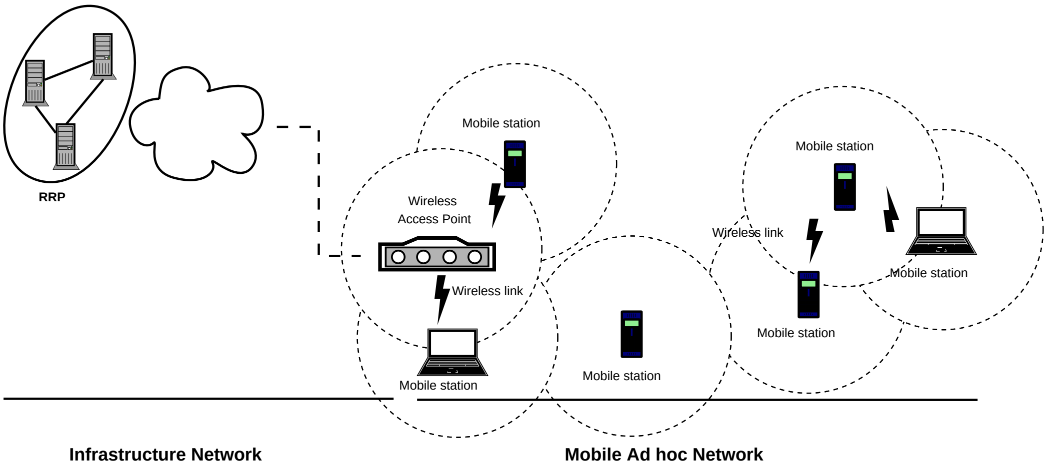

Figure 2: Network architecture

Figure 2 shows the general architecture of the network we are considering. We have mobile stations that communicate with each other via radio waves, using technologies such as Bluetooth or WiFi. The infrastructure network can be Internet or any IP network with fixed and perfectly addressable stations. The communication between the MANETs mobile stations and those of the infrastructure network can be done through the cellular network (3G/4G) or through a Wifi access point for example. Each MANET station is able to contact a station in the infrastructure network if it has access to it. In the case of an Android smartphone, for example, access to the infrastructure network can be detected by a change in the status of the Wi-Fi indicating the connection to an access point when the Wi-Fi is activated, or when the user activates the mobile data connection (3G/4G) for example.

In the infrastructure network, we consider the existence of an entity called Remote Rendezvous Point (RRP). It can be a station known and accessible by all MANET stations having access to the infrastructure network. It can also be a set of fixed stations collaborating in a distributed way to ensure the storage and dissemination of publications, and linked together by a Chord [1] network over Internet for example. The RRP is considered to be subscribed to all topics in the publish/subscribe system. It is considered to have an unlimited amount of storage space and energy, so that it can store all publications coming from any MANET station.

4.2 Interacting with the Remote Rendezvous Point (RRP)

The communication between the MASN station and the RRP is illustrated by the figure 3(a). When a MASN station discovers that it has access to an infrastructure network, it instantly initiates a communication with the RRP located on this network by sending a message to it, identical to the classic *QUERY* announcement message of the SocialMANET protocol (Fig. 1). Processing this message allows the RRP to deduce two pieces of information (Fig 3(b)):

1. the identifiers of the publications to be sent to the MASN station; those are the publications from the RRP's publications set which are not included in the publications set held by both the RRP and the mobile station, as far as each of them belongs to one of the topics specified in the query sent by the mobile station;

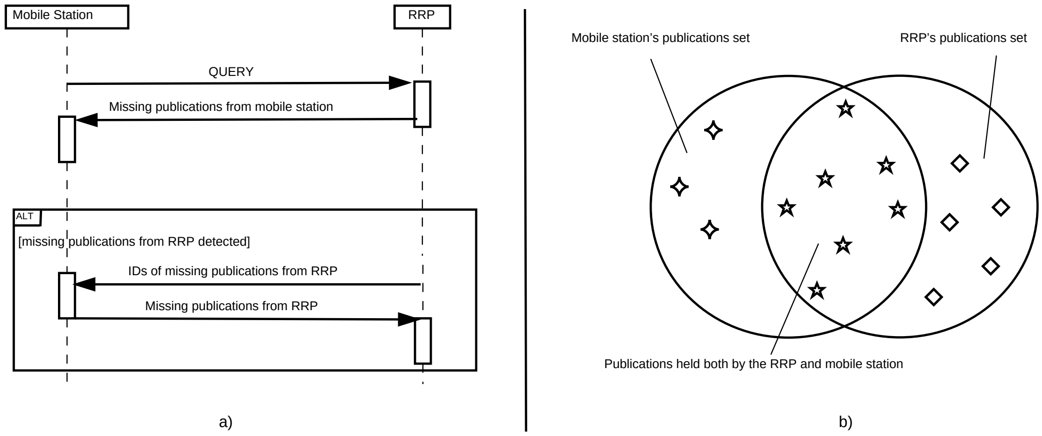

Figure 3: Communication between RRP and Mobile station

2. the identifiers of the publications to be retrieved from the MASN station; those are the publications from the mobile station's publications set which are not included in the publications set held by both the RRP and the mobile station

In response, the RRP sends back to the MASN station the list of publications which it knows that it needs, after interpretation of the *QUERY* message received. It also sends the identifiers of the publications to be requested if available. The MASN station then sends these publications to the RRP and thus the RRP and the MASN station are all up to date with each other.

V SOFTWARE ARCHITECTURE AND IMPLEMENTATION

In this section we present the software architecture we proposed, and an implementation of a prototype mobile application for the Android platform. The goal of the application is the sharing of information using a publish/subscribe communication model within a MASN formed by the participants of a scientific conference. The information to be disseminated is grouped into topics. Topics' names reflect the categories in which the participants of the conference can be classified: speakers (plenary speaker, session speaker, and talker), the organizing committee (secretariat, protocol, and logistics), chairs (plenary chair, session chair), guests, etc. Users have to subscribe to a given topic in order to receive related publications.

We simulate device-to-device connectivity in ad-hoc mode using Google's Nearby Connections API [19]. Indeed, although direct device-to-device connectivity via the ad-hoc mode was available in IEEE 802.11, it is still not widely available in the devices [13]. Researchers in the literature simulate it by using Wifi Direct or Wifi P2P which allows devices to connect to each other and form groups [10, 11, 13]. Nearby Connections follows the same logic and offers interfaces that facilitate the discovery of nearby devices and communication with them.

5.1 Software Architecture

The software architecture we proposed is the one illustrated in figure 4. It has two main components. The first one represents a mobile station and is based on the architecture defined in [16]. The second one represents the **RRP**, the remote and fixed entity located on the infrastructure network, able to store all publications and able to communicate with all mobile stations that have access to it.

The first component has three main layers which are:

1. **Service Layer**: it offers users a set of services allowing them to subscribe to a topic, to publish information and even to be notified when a new publication is received from a neighbouring station;
2. **Application Layer**: through its Publications Manager and Subscription Manager components, it manages the user's topics, publications and subscriptions;
3. **Communication Layer**: it is responsible of functions such as discovering and updating information about neighbouring stations, and communicating with them. This layer also is responsible for detecting the connection status of the mobile station to an infrastructure network and for communicating with the RRP in that network.

In this architecture, the *IHM* layer represents a set of UI components allowing the user to interact with the software. The storage is where topics and publications are stored. The *RRP* has a three layers structure; those are:

1. **Controller**: this is the entry point of all the requests sent to the *RRP*. The Controller layer access the Service layer to handle the request according to the data sent by the mobile station, then return back the response returned by the Service layer to the mobile station;
2. **Service**: this is where the business logic of the *RRP* is implemented. The service layer is responsible of analysing the data sent by the mobile station in order to determine which publication it can send back and which publication it can request back to the mobile station. It communicates with the Repository layer to store and load publications and topics;
3. **Repository**: the logics to access publications and topics are implemented here. The Repository layer is responsible of accessing the storage to save and retrieve publications and topics.

5.2 Discovering neighbouring devices

The *Nearby Connections* API of Google allows to advertise, discover and connect nearby devices in a completely off-line peer-to-peer manner. Connections among devices are high-bandwidth, low-latency and fully encrypted to enable fast and secure data transfers. Under the hood, the API uses a combination of Bluetooth, BLE and Wifi hotspots, taking advantage of the strengths of each while working around their respective weaknesses. For convenience, users are not prompted to enable Bluetooth or Wifi. Nearby Connections activates these features as they are needed and returns the device to its previous state once the application has finished using the API.

Nearby Connections uses three advertisement and discovery policies. The *P2P_POINT_TO_POINT* policy supports a 1-to-1 connection topology; it does not allow more than one connection at a time. The *P2P_STAR* policy supports a 1-to-N, or star, connection topology. The *P2P_CLUSTER* strategy supports an M-to-N, or cluster, connection topology. We used the latter because it allows devices to accept multiple incoming connections and simultaneously initiate multiple outgoing connections to other neighbouring devices.

Once the devices are connected, they can communicate. The information transmitted is encapsulated in the Java class *com.google.android.gms.nearby.connection.Payload*. Three types of payloads are available:

- **BYTE**, allow to send simple data such as text messages,
- **FILE**, allow to send files such as photos and videos,

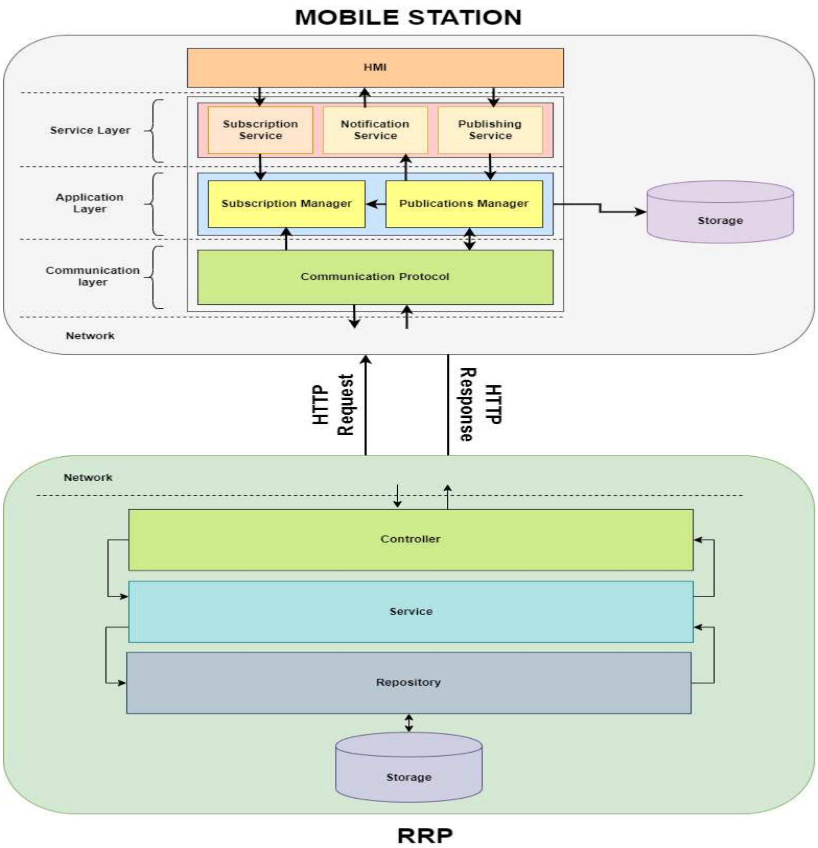

Figure 4: The proposed software architecture

- **STREAM**, suitable for large amounts of data generated on the fly, such as an audio stream.

5.3 Needs calculation

As stated in the SocialMANET protocol [14], stations periodically broadcast *QUERY* announcement messages to signal their presence and these messages are used to calculate their respective needs in terms of missing publications. Since the discovery of nearby devices in our implementation is entrusted to the Nearby Connections API, the *QUERY* message is sent to devices as soon as a connection is established with them. However, as specified in the SocialMANET protocol, any new publication made by a device is immediately broadcast to all other nearby connected devices. The construction of the *QUERY* message is done by the method *computeQuery* (Algorithm 1). A new entry[2] is added to the query for each topic the current station has subscribed to (line 4). Then that entry is further updated by adding the IDs of the publications the current station has on that topic (lines 5-8).

When a device receives the *QUERY* message, it updates the information about the devices it is connected to, and then builds a list of publications to send and a list to inquire from the device that sent the *QUERY* message. These operations are respectively performed by the methods *computePubsToSend* and *computePubsToAsk* defined in the main activity Java class

[2] A query entry is made up by a topic ID and a collection of IDs of publications the current station has on that topic

Algorithm 1: Preparing the QUERY message

```
private Query computeQuery() {
  Query query = new Query();
  for (Topic topic : topicsManager.getSubscriptions()) {
    query.addEntry(new QueryEntry(topic.getId()));
    for (Publication pub:
      publicationManager.getPublications(topic)) {
      query.addPublication(topic.getId(), pub.getPublicationId());
    }
  }
  return query;
}
```

MainActivity.java. The sources of this mobile application can be found at the following link: https://bitbucket.org/martinxt/socialmanet.

5.4 Communication with the RRP

In our implementation, the RRP is a REST API deployed on a server on the Internet. The full documentation of this API is available in [20]. It exposes the following endpoints:

1. **/rrp-api/v1/publications/get**: It is used to retrieve publications stored on the *RRP* that are missing from the station sending the request. Requests sent to this endpoint are made using the http *POST* method. The body of such a request is basically a JSON object with a unique property *entries*, which is a collection of query entries representing the part of the QUERY message of the SocialMANET protocol delimited by brackets (see Fig. 1). The content of the response to a request sent to this endpoint is a also JSON object. It has two properties named *publications* and *pubRequestIds*.The former is a set of publications matching the query received by the RRP as the request body. The later is a set of publication IDs that were present in the query received by the RRP and whose corresponding publication are unknown by it (see sub-section 4.2). An example of a request body and a response body for this endpoint is shown in figure 5.
2. **/rrp-api/v1/publications/save**: This endpoint is used to send publications to the *RRP* which will then store them. It is useful when a station publishes an information while connected to the Internet. The body of such a request represents the set of publications that are being sent to the RRP (Fig. 6).

According to the Android platform, to ensure that an application is notified when a device is connected to a network, it has to subscribe to the *BroadcastReceiver*. This have been done in the main activity Java class of our implementation (*MainActivity.java*). When this happens, the *handleConnectionStatusChanged* method is executed. This method is defined in the Java class *MainActivity.java* and depending on the status (on-line or off-line) of the device, it initiates a communication with the *RRP* as described in section 4.2.

```
{
  "entries": [
    {
      "topicId": "topic-1",
      "publicationIds": "pub-1_._.pub-2_._.pub-5"
    }
  ]
}
```


Request body


```
{
  "publications": [
    {
      "publicationId": "pub-3",
      "topicId": "topic-1",
      "publishDate": 1663502695205,
      "validity": 3600000,
      "description": "sample description",
      "path": null
    }
  ],
  "pubRequestIds": ["pub-5"]
}
```


Response body

Figure 5: Fetching publications form RRP: Request and Response bodies

VI RESULTS AND DISCUSSIONS

6.1 Results

We implemented a prototype application for information sharing in a scientific conference. This application is made to run in Android smartphones. The source code is hosted on Bitbucket in a public repository at the following address: *https://bitbucket.org/martinxt/socialmanet/* (a README file which explains how to clone an run the code is available). The REST API that we proposed as the Remote Rendezvous Point (*RRP*) is hosted on Heroku and the URIs of the different endpoints and their descriptions are available in [20].

Figure 7 shows three screen shots of the application. The different essential parts that can be seen in figure 7(a) are labelled with numbers and described as follows:

1. Indicator which, when green, shows that the device is scanning the perimeter looking for nearby devices;
2. Indicator that when green, shows that the device is advertising to show its presence;
3. Navigation bar to access the list of publications, the list of topics (current tab) and the list of nearby devices respectively;
4. List of topics, headed by topics to which the user has subscribed (*Subscriptions*).

Figures 7(b) and 7(c) show a device having two devices available in its vicinity: *TDWT* and *O8KQ*. At 11:09, only *TDWT* is connected (Fig. 7(b)); one minute later, *O8KQ* is also connected (Fig. 7(c)).

Figure 8 demonstrate the capacity of the application of getting publications from nearby devices using Wi-fi Direct or Wi-fi P2P (Fig. 8 (a, b, c)) and from the RRP via Internet (Fig. 8 (d, e, f)). We have two devices that have both subscribed to the topics *Speaker* and *Protocol*. They can be distinguished by their battery level indicators (*WLU7* with a low battery level and *GBLT* with a high battery level).

Figure 8(a) shows the device *WLU7* publishing a message at 19:15. At 19:18, it gets near to the device *GBLT* and thanks to the communication protocol implemented in the application, *GBLT* will receive from *WLU7* the publication it made few minutes ago. Figure 8(b) shows *GBLT* receiving a notification indicating that a new publication has been received. Figures 8(d, e, f) show the same devices. They are not nearby each other. *GBLT* publishes a message at 19:20 while connected to Internet (Fig. 8(d) - notice the mobile data indicator). As a result,

```
[
  {
    "publicationId": "pub-6",
    "topicId": "topic-1",
    "publishDate": 1663502695205,
    "validity": 3600000,
    "description": "Sample publication",
    "path": null
  },
  {
    "publicationId": "pub-7",
    "topicId": "topic-1",
    "publishDate": 1663502697542,
    "validity": 240000,
    "description": "New publication",
    "path": null
  }
]
```


Figure 6: Pushing publications to RRP: Request/Response body

this publication is dispatched to the *RRP* and *WLU7* receives it few seconds later as it activates mobile data connection and gets connected to Internet (Fig. 8(e)).

6.2 Discussions

The approach we propose in this paper is based on the SocialMANET protocol. This is a publish/subscribe information dissemination protocol for MANETs presented in [14], which we extended to make it able to work in an MASN whose underlying MANET is coupled to an infrastructure network. However, it is worth to note that the same approach can be replicated using any other communication protocol similar to SocialMANET. It would simply be a matter of equipping that protocol with primitives enabling it to communicate with stations located in an infrastructure network using a message format that may or may not be identical to that used to communicate with mobile stations.

In the network architecture we considered, no MANET station acts as a gateway for other stations to access the infrastructure network. It is true that several solutions in the literature [4, 12, 15] employ this technique to allow even MANET stations without direct access to the infrastructure network to access it through other stations (gateways). However, this places an additional load on the gateway that acts as a relay between stations on the infrastructure network and those on the MANET that do not have access to the infrastructure network. This load is likely to lead to additional energy consumption of the gateway, which is why we avoid using mobile stations as gateways for other mobile stations. This choice also adheres to the principle of non-waste of energy used by SocialMANET [14], which consists of a station avoiding wasting energy to participate in the dissemination of messages that are not intended for it or for which it has no interest, in order to avoid running out of energy very quickly.

Contrary to other software architectures proposed in the literature, such as those presented in [9, 11, 13], our proposed software architecture is perfectly adapted to MASNs in which stations communicate by the publish/subscribe communication model. The other architectures, on the other hand, are designed for MASNs where stations communicate synchronously after having discovered a high match rate during a comparison among their interests.

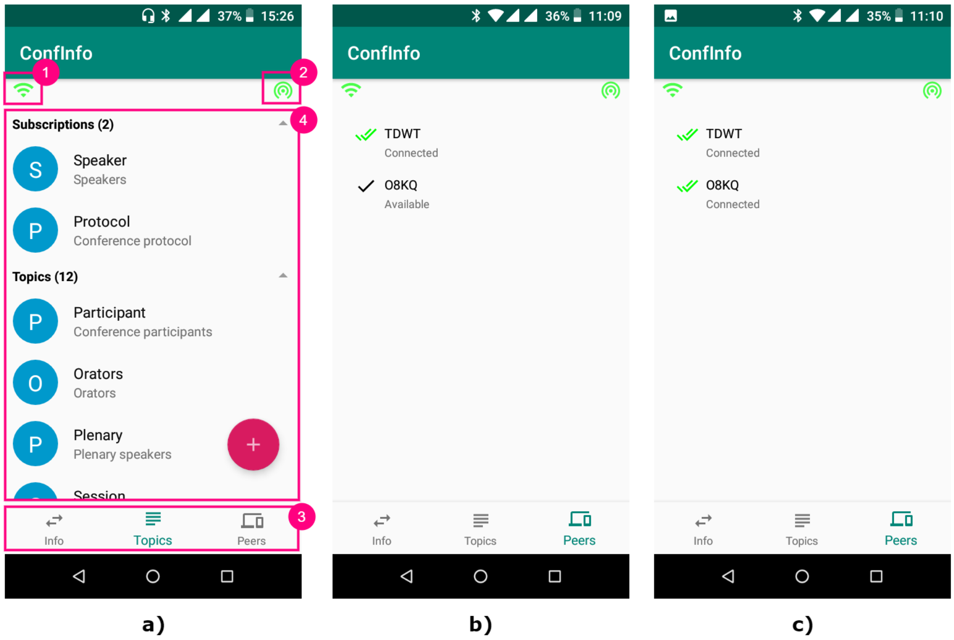

Figure 7: Application main parts and nearby devices

VII CONCLUSION

In this paper, we have presented an architecture for building hybrid mobile ad-hoc social networks for information dissemination using a publish/subscribe communication model. We also presented an implementation of such a network by prototyping an Android mobile application for information dissemination in a scientific conference. We used a communication protocol called SocialMANET which is a publish/subscribe communication protocol, and we have extended it to enable communication among MASNs and infrastructure networks' stations. The results obtained have enabled us to demonstrate the applicability of our solution. Nevertheless, it would be wise to continue with the validation of our solution by carrying out experiments in a real context in order to draw conclusions based on the results obtained in practice. This is what we are planing to do as a next iteration of this work.

VIII ACKNOWLEDGEMENTS

The authors would like to express their sincere thanks to the anonymous reviewers of the earlier version of this paper presented at the Conference CRI'2021, for the insightful comments and valuable suggestions they made, and which allowed us to improve its quality.

REFERENCES

[1] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications". In: *IEEE/ACM Trans. Netw.* 11.1 (Feb. 2003), pages 17–32. ISSN: 1063-6692.

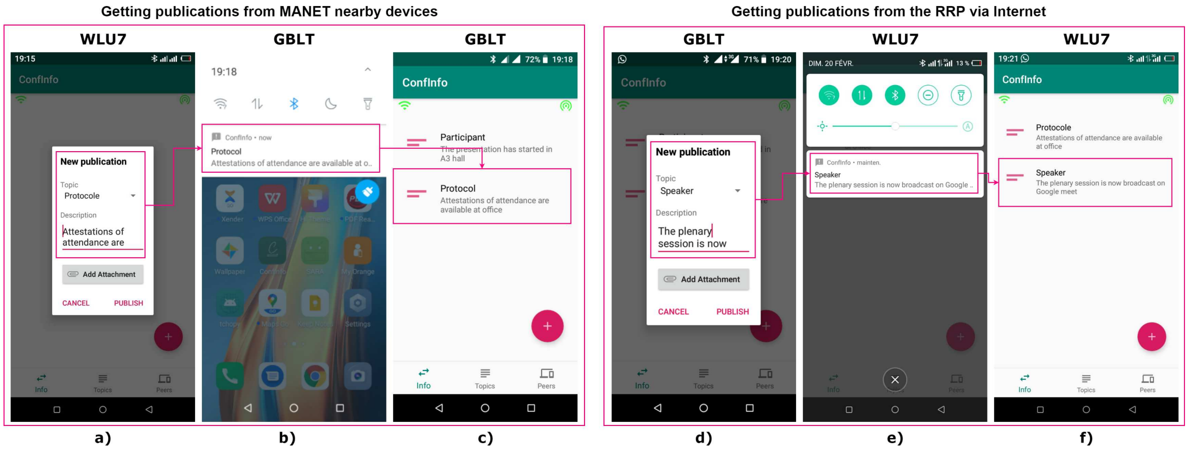

Figure 8: Publishing and dispatching publications

[2] C. Borcea, A. Gupta, A. Kalra, Q. Jones, and L. Iftode. "The MobiSoC Middleware for Mobile Social Computing: Challenges, Design, and Early Experiences". In: *Proceedings of the 1st International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*. MOBILWARE '08. Innsbruck, Austria: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008. ISBN: 9781595939845.

[3] K. U. Rahman Khan, R. U. Zaman, and A. V. G. Reddy. "Integrating Mobile Ad Hoc Networks and the Internet: challenges and a review of strategies". In: *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE '08)*. 2008, pages 536–543.

[4] K. U. R. Khan, A. V. Reddy, and R. U. Zaman. "An Efficient Integrated Routing Protocol for Interconnecting Mobile Ad Hoc Networks and the Internet". In: *Proceedings of the International Conference on Advances in Computing, Communication and Control*. ICAC3 '09. Mumbai, India: Association for Computing Machinery, 2009, pages 460–466. ISBN: 9781605583518.

[5] I. Leontiadis, P. Costa, and C. Mascolo. "A Hybrid Approach for Content-Based Publish/Subscribe in Vehicular Networks". In: 5.6 (Dec. 2009), pages 697–713. ISSN: 1574-1192.

[6] L. Zhang, X. Ding, Z. Wan, M. Gu, and X.-Y. Li. "WiFace: A Secure Geosocial Networking System Using WiFi-Based Multi-Hop MANET". In: *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. MCS '10. San Francisco, California: Association for Computing Machinery, 2010. ISBN: 9781450301558.

[7] J. Lee and C. S. Hong. "A mechanism for building Ad-hoc social network based on user's interest". In: *2011 13th Asia-Pacific Network Operations and Management Symposium*. 2011, pages 1–4.

[8] T. Mishra, D. Garg, and M. M. Gore. "A Publish/Subscribe Communication Infrastructure for VANET Applications". In: *2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications*. 2011, pages 442–446.

[9] L. V. Hoang and H. Ogawa. "A platform for building ad hoc social networks based on Wi-Fi direct". In: *2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE)*. 2014, pages 626–629.

[10] D. Zhang, D. Zhang, H. Xiong, C.-H. Hsu, and A. V. Vasilakos. "BASA: building mobile Ad-Hoc social networks on top of android". In: *IEEE Network* 28.1 (2014), pages 4–9.

[11] N. Aneja and S. Gambhir. "Middleware Architecture for Ad-hoc Social Network". In: *Research Journal of Applied Sciences, Engineering and Technology* 13.9 (2016), pages 690–695.

[12] J. A. Silva, J. Leitão, N. Preguiça, J. M. Lourenço, and H. Paulino. "Towards the Opportunistic Combination of Mobile Ad-Hoc Networks with Infrastructure Access". In: *Proceedings of the 1st Workshop on Middleware for Edge Clouds and Cloudlets*. MECC '16. Trento, Italy: Association for Computing Machinery, 2016. ISBN: 9781450346689.

[13] N. Aneja and S. Gambhir. "Profile-Based Ad Hoc Social Networking Using Wi-Fi Direct on the Top of Android". In: *Mobile Information Systems* 2018 (Oct. 2018), pages 1–7. ISSN: 1875-905X.

[14] M. T. Tchendji, M. X. Tchembé, and I. T. Necheu. "SocialMANET: A Publish/Subscribe Events Dissemination Protocol for Mobile Ad-Hoc Networks". In: *Journal of Computer Science* 15.9 (Sept. 2019), pages 1237–1255.

[15] M. Al mojamed and L. Dong. "Integrating Mobile Ad Hoc Networks with the Internet Based on OLSR". In: *Wirel. Commun. Mob. Comput.* 2020 (Jan. 2020). ISSN: 1530-8669.

[16] M. T. Tchendji, M. X. Tchembé, and A. L. M. Kakeu. "An ad-hoc social network generation approach". In: *CoRR* abs/2008.04618 (2020). arXiv: 2008.04618.

[17] N. Aneja and S. Gambhir. "Recent Advances in Ad-Hoc Social Networking: Key Techniques and Future Research Directions". In: *Wireless Pers Commun* 117 (2021), pages 1735–1753.

[18] R. Xing, Z. Su, Q. Xu, and A. Benslimane. "Truck Platooning Aided Secure Publish/Subscribe System Based on Smart Contract in Autonomous Vehicular Networks". In: *IEEE Transactions on Vehicular Technology* 70.1 (2021), pages 782–794.

[19] *Overview - Connections API*. https://developers.google.com/nearby/connections/overview. Accessed : 2022-07-28.

[20] *Remote Rendez-vous Point: OpenAPI definition*. http://kamvusoft.com:8085/swagger-ui/index.html. Accessed : 2022-07-28.