MDTN: Mobile Delay/Disruption Tolerant Network

C. E. Palazzi, A. Bujari, S. Bonetta
Department of Pure and Applied Mathematics
University of Padua, Padua – Italy
{cpalazzi | abujari}@math.unipd.it

G. Marfia, M. Roccetti, A. Amoroso
Deparment of Computer Science
University of Bologna, Bologna – Italy
{marfia | roccetti | amoroso}@cs.unibo.it

Abstract— **Infrastructure networks like Internet have some coverage gaps (e.g. rural areas) where no service connectivity is provided. Cellular networks could be used in alternative but they usually associate their services with a usage cost. Nowadays, the growing number of mobile devices equipped with a wireless interface and the end-user trend to shift toward wireless technology has opened new possibilities for networking. Opportunistic networking seems a feasible solution in filling in these coverage gaps by providing service connectivity where it is not sustained by an infrastructure. In this context we provide a delay tolerant platform that ensures service connectivity where it is absent. Our approach follows the Delay/Disruption Tolerant Networking (DTN) paradigm by implementing a store-and-forward communication model between mobile users and *carrier entities* (e.g. buses), where a user delegates the carrier a task (*bundle*), the carrier stores it locally and whenever service connectivity is available (e.g. wi-fi at the bus station) tries to accomplish it, successively notifying the user of the task output next time they encounter again.**

Keywords-component; mobile; opportunistic networking; DTN; multi-hop;

I. Introduction

Infrastructure networks like Internet lack of total coverage of the territory, they suffer from what is called the *last mile* problem [1]. Moreover, in underdevelopment countries this problem is broader and persistent. Even though solutions exist they are considered expensive or at least they require time to be actuated [2]. Meanwhile, with the growing number of mobile devices equipped with wireless interface, mobile users increasingly find themselves in different types of potential networking environments. These networks, span from infrastructure connected networks like cellular networks to the entirely disconnected networks of stand-alone mobile devices. Cellular networks could be a valid alternative in providing Internet access but they usually associate their services to a usage cost (e.g. UMTS [3]).

Opportunistic networking in terms of involving multi-hop Delay/Disruption Tolerant Networking (DTN [4]) represents an interesting alternative as it has no usage cost and could provide service coverage where it is not sustained by an infrastructure. Recent trends show that it has gain a lot of popularity in interconnecting mobile users in an ad-hoc fashion and provide service access by leveraging on node mobility to reach infrastructure end-points [5, 6].

Indeed, DTNs support interoperability of networks by accommodating long delays between and within them, and by translating between network communication characteristics. In providing these functions, DTNs can accommodate the mobility and limited power evolving wireless communication devices. Our objective is to implement and study a DTN build by the Public Transportation System (PTS). To this aim we have considered a scenario where carriers are public buses with on board Wi-Fi Access Points (AP), with routes going to/from rural areas from/to city center and users are mobile entities owning a handheld device, both of them are running and operating our software.

In this context we propose *Mobile Delay/Disruption Tolerant Network* (MDTN), a delay tolerant application platform which can be used to provide service coverage where Internet is not available (e.g. rural areas). Our software achieves this by implementing a store-and-forward communication model, DTN-like, where a mobile user delegates a task (*bundle*) to a carrier entity, the carrier stores it locally for later execution and the next time they encounter notifies the user of its output. Carriers are mobile entities that could gain Internet access further in time and act as a mule between us and the Internet. By delegation is meant that the system locally encodes the task and forwards it to the remote carrier for execution. MDTN offers the end-user the possibility of accessing push/pull services like *email on the go* and content retrieval like web page or multimedia file downloading.

The remainder of this paper is organized as follows. Section II summarizes background information needed to comprehend the applicative domain. Section III gives some insights about MDTN, its modus operandi and architectural design. In Section V we make a comparison between our work and others residing in the same applicative domain as MDTN. Finally, in Section IV concluding remarks are provided along with some possible future direction for our work.

II. Background

In this section we provide some insights on DTNs, their characteristics, purpose and communication paradigm.

A. *Delay/Disruption Tolerant Network (DTN)*

Each network is adapted to particular communication characteristics; the TCP/IP protocol suite [7] has been a great success at interconnecting communication devices across the Internet where connectivity relies primarily on wired links continuously connected in end-to-end, low-delay paths between the sources and destinations. Nowadays, many evolving and potential networks do not confirm to the Internet underlying assumptions. These networks are characterized by:

- **Intermittent connectivity**: connection links are transient and short in time due to mobility;
- **Long or variable delays**: Propagation delays between nodes and variable queuing delays at nodes contribute to end-to-end delays that can defeat Internet protocols;
- **Asymmetric data rates**: The Internet supports moderate asymmetries of bidirectional data rate for users with cable TV or asymmetric DSL access. But if asymmetries are large, they defeat actual protocols;
- **High error rates**: Bit errors on links require correction or retransmission of the entire packet. For a given link-error rate, fewer retransmissions are needed for hop-by-hop than for end-to-end retransmission.

A DTN [5] originally thought and conceived for communication in outer space, is an overlay of networks, including the Internet. DTNs achieve interoperability by interfacing between different communication networks characteristics. In providing these functions, DTNs accommodate the mobility and limited power evolving wireless communication devices. They overcome the aforementioned problems by employing a store-and-forward message switching (Fig. 1). Whole messages or pieces of such messages are forwarded from a storage place on one node to a storage place on another node, along a path that eventually reaches the destination. To do so, DTN routers need persistent storage for their queues as opposed to Internet routers that use short-term storage provided by memory chips. This for the following reasons:

- A communication link to the next hop may not be available for a long time.
- One node in a communicating pair may send or receive data much faster or more reliably than the other node.
- A message, once transmitted, may need to be retransmitted if an error occurs at an upstream link, or if an upstream node declines acceptance of a forwarded message.

All this functionalities are provided by the DTNs by adding a new protocol layer called the Bundle Layer (BL [8]). Our idea is to build a DTN between mobile users and carriers who communicate through Wi-Fi instead of UMTS or other related technologies whose usage is associated to a cost. We leverage on carrier mobility to reach infrastructure end-points which have Internet connectivity.

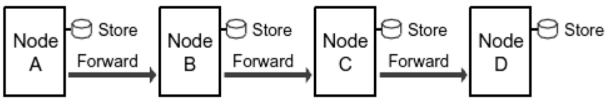


Figure 1 – DTNs store-and-forward communication model.

B. *Bundle Layer and Bundle Protocol*

The Bundle Layer (BL) sits over the specific Transport Layer of a network. The aim of this new layer is to link together different and heterogeneous networks, allowing message transmissions (*bundles*) between and among them (Fig. 2). Every bundle consists of three parts: (i) bundle header, (ii) control information, and (iii) source-application data. This means that a bundle is just an additional encapsulation step and could work properly over different networks. BL communicates using a unique and specific non-conversational protocol named Bundle Protocol (BP [7]) that minimizes the number of round trips required to confirm transmissions, making the acknowledgements optional.

BP provides end-to-end communication over performance-challenged networks while allowing interconnection of highly heterogeneous networks such as IP does; however, they make use of special layer adapters [8]. BP uses each local network preferred transport protocol and becomes more flexible than IP by allowing for interconnecting greater differing network types than IP as are common in space communication. It is worth noting that BP does not replace IP but in fact uses a TCP/IP convergence layer adapter when connection is needed on an IP-based network (Fig. 3).

III. MDTN

In this section we introduce our MDTN software by outlining the entities taking part in the system. Initially we extend the general case scenario where software could be operable and then we proceed by reviewing the implemented scenario. We also show some implementation details along with a real usage case are shown.

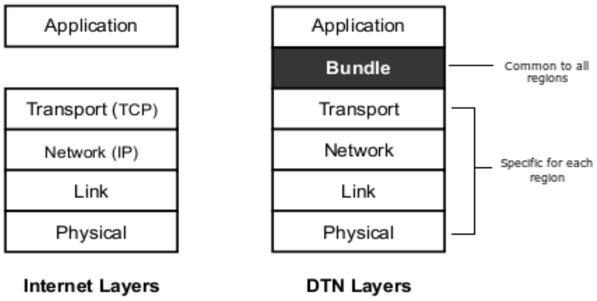


Figure 2 – DTNs Bundle Layer.

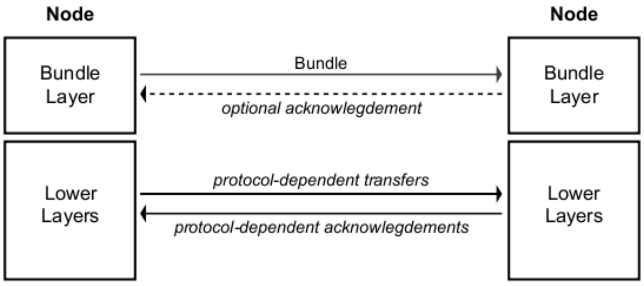


Figure 3 – DTNs Bundle Protocol stack.

A. *Modus operandi*

We now explain the architecture of MDTN by fixing the entities involved in the communication. Referring to Fig. 4 we can see that there are two main entities involved: (i) the MDTN-client or a mobile user operating a smartphone and (ii) the MDTN-server or the bundle carrier which locally stores users requests, tries to accomplish them when Internet connection is available (e.g. bus stop or station).

Once established a connection with the server the client forwards it a task along with a list of potential carriers (busses) the user is going to take on later. The server could try to accomplish the task by itself or delegate it to other servers along its route. Once the task is accomplished its output is forwarded toward the requested user destination/s (carrier list) and available from there for user retrieval.

In this scenario we can delineate different possibilities which we aim to investigate:

- User gets on the bus, makes a requests and demands that task output is forwarded to another bus he might take later on;
- User gets on the bus, makes a requests and demands that task output is forwarded to a potential list of buses he might take later on;
- The server to whom user delegated the task accomplishes it by itself and forwards the output to the demanded destination/s through an efficient routing algorithm;
- The carrier to whom user delegated the task delegates the task toward another carrier which can achieve task accomplishment in a timely fashion and later on forward the output toward the demanded destinations.

All these scenarios involve multi-hop routing between carriers. Since the PTS has a global view of its network, efficient and time critical (deadline) routing could be possible.

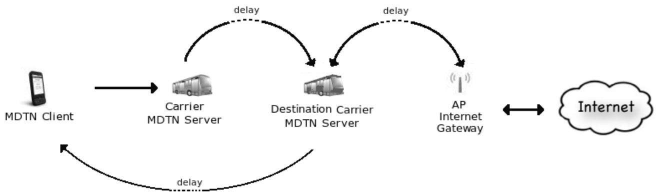


Figure 4 – MDTN complex scenario.

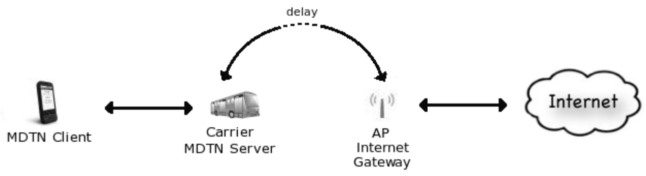


Figure 5 – MDTN simple scenario.

The aforementioned modus-operandi is the general case scenario. In the current implementation the client delegates a task to the server and this one accomplishes it and the next time they encounter (e.g. user on board) the output is ready for retrieval (Fig. 5). This process is entirely client-transparent; the client has an established connection with the server and is notified for the task output when/if the task has been accomplished. Obviously, the client can disconnect/connect from/to the server at any time and the task output will be forwarded next time they pair with each other. The implemented scenario compared to the general case could seem trivial at first glance but it serves as a proof of concept and as a baseline for future developments.

B. *MDTN protocol*

The MDTN-client can be in the following two possible states:

- **online**: when he is connected to the MDTN server;
- **offline**: when he is disconnected from the service.

Instead the MDTN server can be in the following two operating modes:

- **gathering**: server does not have Internet connectivity but he is actively receiving and storing user requests eventually forwarding them stored information previously accomplished for other users that requested them;
- **digesting**: server reaches Internet connectivity and is able to accomplish (digest) pending user requests (gathering mode is still available).

Both these entities need to implement and use the same communication protocol (Fig. 6) which is a slightly modified version of the Bundle Protocol. It is important to note that both of them implement a BP-Application Layer as it is required and application specific. Also, inside a DTN every node that is able to send and receive bundles is called *bundle node*, which is characterized by the presence of three fundamental components:

- BPA (BP-Application Layer): is the Bundle Layer services supplier, which allows higher levels to communicate through the DTN;
- CLA (Convergence Layer Adapter): is the adapter which allows the Bundle Overlay to be placed over various physical networks that can work with different Transport Protocols (like TCP). There can be more than one adapter for each node;
- AA (Application Agent): is the component that uses the BPA for communications purpose.

The bundle format used is slightly different respect to RFC specification, since some fields have been added. Every bundle is composed by (at least) two blocks: the Primary Block (Fig. 7), which contains control information, and the Payload Block (Fig. 8), that consist of useful user-data.

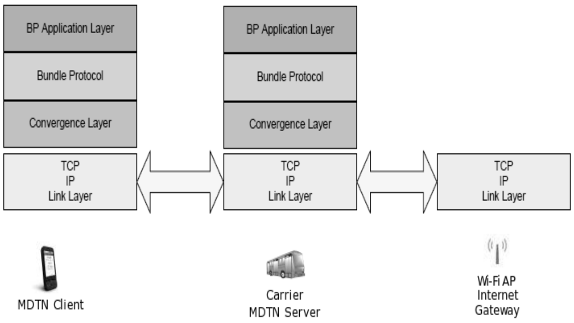
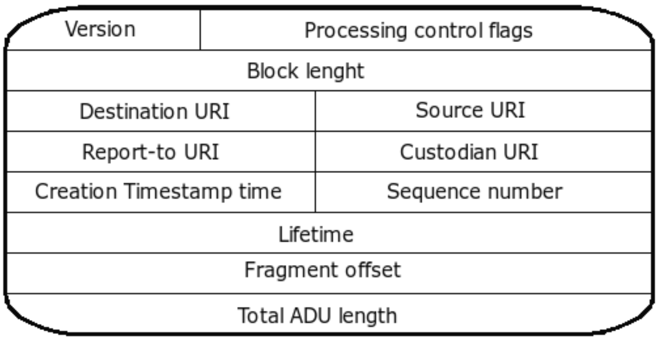
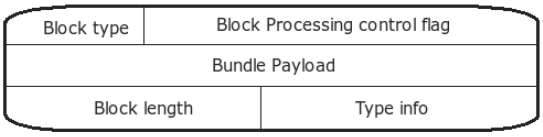


Figure 6 – MDTN protocol stack.



Version	Processing control flags
Block lenght	
Destination URI	Source URI
Report-to URI	Custodian URI
Creation Timestamp time	Sequence number
Lifetime	
Fragment offset	
Total ADU length	

Figure 7 – Bundle, primary block.



Block type	Block Processing control flag
Bundle Payload	
Block length	Type info

Figure 8 – Bundle, payload block.

C. *MDTN Implementation*

Our MDTN is a Java based application built for Android capable devices and tested on HTC Hero, HTC Desire, HTC Evo 4G and HTC Tatoo. It consists of a package composed by a client and a server both sharing the same core module handling the DTN communication. The client consists of a tabbed interface allowing the user to manage and easily interact with the MDTN services. These services are:

1. MDTN-Status: handle service connection and logs;
2. MDTN-Email: send e-mail;
3. MDTN-Files: require and download Internet resources (e.g. web page, multimedia document).

At this point we show by means of an experiment a concrete usage of MDTN. The map in Fig. 9 shows the interactions taking place between the carrier and the user respectively operating MDTN-server and client. In this demo the carrier is a car with a predetermined route and an on board Wi-Fi AP while the user is mobile and operating MDTN-client on an Android enabled device.

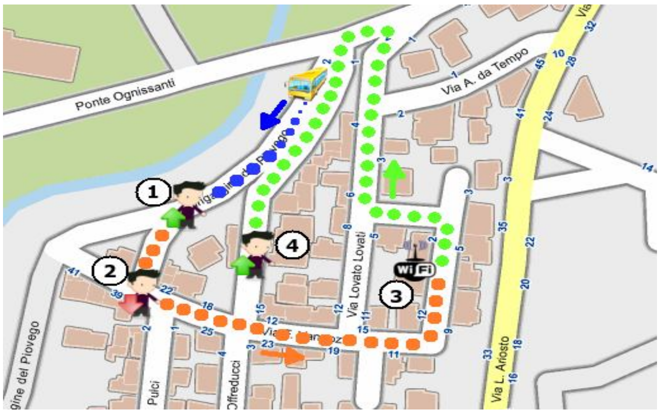


Figure 9 – Demo map showing the interactions along the route between carrier and user. The blue dots represent the car route with no pending tasks. The orange dots represent the carrier route where there are pending tasks, issued by the user at 2. Along the green dots path the carrier has satisfied user requests (at 3) and is ready to forward the output when user gets on board at 4.

The enumerated points in the map represent the following actions:

1. The user gets on the car and successively forwards MDTN-server a task;
2. The user gets off the car and continues his journey;
3. The carrier is nearby a wireless AP connected to the Internet (e.g. bus station), satisfies users pending tasks and stores the output locally;
4. The user gets on the car and once connected to the MDTN-server retrieves the required content.

The Figs. 10-14 show the aforementioned interactions taking place for each designated point of the map. In Fig. 10 the vehicle (carrier) leaves the station equipped with Internet connection and travels along the predetermined route shown in the map above. In Fig. 11 the user gets on the vehicle, establishes a connection to the MDTN-server and forwards some requests (Fig. 12). Once user has arrived at destination, he gets off the vehicle, disconnecting from the server meanwhile the carrier follows its route toward the station (Fig. 13). Left the station the carrier has satisfied user requests and stored their output locally. Fig. 14 shows the user getting on board of the vehicle, connecting to the MDTN-server in order to retrieve the requested contents.

Figure 10 – Car representing the carrier parked at the base station.

Figure 11 – User waiting at the car stop.

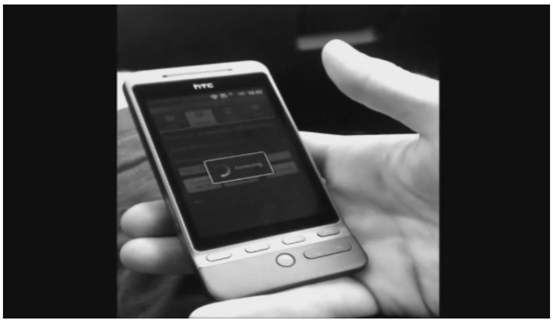


Figure 12 – User gets connected to the server and issues requests.

Figure 13 – Carrier has arrived at the station and begins satisfying user pending requests.

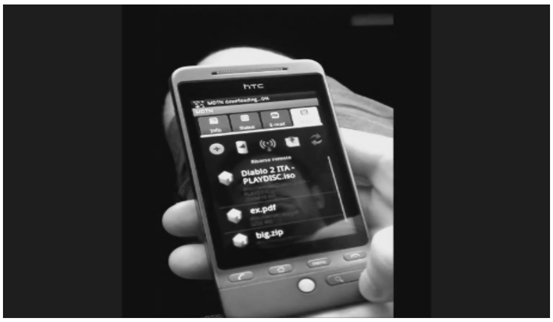


Figure 14 – User gets on board of the car, connects to the server and finds the desired requested content.

IV. Comparison with Related Work

Other DTN based solutions for smartphones exist in Internet and in scientific literature. Therefore, in Table I we compare our MDTN software with other related works by considering different technological aspects, spanning from the architectural design, to the communication style and entities involved.

TABLE I. Comparison with related work.

	MDTN	**Bytewalla**	**Maggiorini *et al.* [5]**
Infrastructure	Static gateway with Internet connection	Static proxy, gateway with Internet connection	Static gateway with Internet connection
Routing	None, interested entities pair with each other when in range	None, interested entities pair with each other when in range	Multi-hop between line carriers
Internet-user Interface	PTS carrier	Proxy	PTS carrier
Service Type	Pull/push style	Pull/push style	Push style

To this aim we have focused our attention in the Bytewalla [9] project whose purpose is to provide service connectivity in areas that lack of Internet coverage by using the DTN paradigm in Android enabled devices. Our project resembles Bytewalla in the means but the architectural design and applicative scenario are quite different. Bytewalla makes use of an infrastructure deployed entity called proxy which is static and acts as an interface between end-users and the Internet. MDTN does not require the deployment of any entity; it leverages the PTS infrastructure which is assumed to have pre-deployed Wi-Fi spots at bus stations (same as Bytewalla). These design choices result in quite different systems architecture and applicative scenario.

We consider in our comparison also [5] by Maggiorini *et al.* Their project regards a routing algorithm for message forwarding from source to destination, by using the PTS as a backbone for delay tolerant communication. One of the main differences with our MDTN is the type of service provided to mobile end-users. The solution in [5] provides only a *push* service, like e-mail or microblogging on-the-go, whereas MDTN also provides the users with *pull* services like content retrieval (e.g. multimedia file or web pages).

V. Conclusion and Future Work

The evolution of device technology has led to a reduction in cost, complexity, processing capability and usability of mobile devices. Thus, opportunistic multi-hop networking seems a feasible solution in providing Internet connectivity where it is not it is not sustained by an infrastructure. Much more by replacing it when usage costs are applied (e.g. UMTS of cellular networks).

In this context we propose MDTN: a delay tolerant application platform that adopts the DTN paradigm enabling the communication between mobile users and carriers which are entities of the public transportation network. A user could delegate the carrier a task, the carrier accomplishes it and next time they encounter forwards the output back to the requestor. In the current implementation no routing is required, the client forwards a task to a carrier and is this carrier’s duty to accomplish it and successively forward back the output.

As stated in Section III-A there are some potential future directions worth exploring. For instance, all of them require multi-hop routing between carriers and since carriers are part of a public transportation network one could assume they might even have information of other carrier routes. This information could be exploited in order to add time considerations while routing requests. To this purpose, we are currently studying the feasibility of adopting THE_ONE simulation environment [10, 11], which would allow us to decide which is the best direction for deployment in real and every day usage scenarios.

REFERENCES

[1] J. Dedrick, V. Gurbaxani, K. Kraemer, Information technology and economic performance: "A critical review of the empirical evidence". ACM Computing Surveys, vol 35, March 2003.

[2] A. Fabrikant, E. Koutsoupias, C. Papadimitriou, "Heuristically Optimized Trade-offs". http://www.cs.berkeley.edu/ christos/.

[3] H. Holma, A. T. Eds., WCDMA for UMTS, 2nd ed.,Wiley, 2002.

[4] S. Burleigh, K. Fall, V. Cerf, R. Durst, K. Scott, H. Weiss, L. Torgerson, A. Hooke, "Delay Tolerant Networking – An Approach to Interplanetary Internet", IEEE Communications Magazine, vol 41, pp. 128-136, June 2003.

[5] L. Pelusi, A. Passarella, M. Conti, "Opportunistic Networking: Data Forwarding in Disconnected Mobile Ad Hoc Networks," IEEE Communication. Magazine., vol 44, pp. 134-141, December. 2006.

[6] S. Gaito, D. Maggiorini, E. Pagani, P. G. Rossi, "Distance Vector Routing for Public Transportation Vehicular Networks: Performance Evaluation on a Real Topology", in Proc of the 2nd IFIP/IEEE Wireless Days 2009, Paris, France, December 2009.

[7] W.R.Stevens, TCP/IP Illustrated, Volume 1- The Protocols, Addison-Wesley, April 2000.

[8] K. Scott, S. Burleigh, "Bundle Protocol Specification" draft-irtf-dtnrg-arch-02.txt, July 2004.

[9] KTH-ICT/Telecommunication Systems Lab, "Bytewalla Delay Tolerant Network on Android phones project". http://www.tslab.ssvl.kth.se/csd/projects/092106/home.

[10] A. Keränen, J. Ott, T. Kärkkäinen, "The ONE simulator for DTN protocol evaluation", Proceedings of the 2nd International Conference on Simulation Tools and Techniques, Rome, Italy, 2009.

[11] F. Ekman, A. Keränen, J. Karvo, J. Ott, "Working Day Movement Model", Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models, NY, USA, 2008.