ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΛΟΓΙΣΜΙΚΟΥ ΚΑΙ ΑΝΑΠΤΥΞΗΣ ΕΦΑΡΜΟΓΩΝ

Σχεδιασμός και Ανάπτυξη Αλγορίθμων που Αξιοποιούν τις Ιδιότητες Δικτύων Ανεκτικών σε Καθυστερήσεις και Διακοπές

DEMOCRITUS UNIVERSITY OF THRACE
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
SOFTWARE AND APPLICATION DEVELOPMENT SECTOR

Design and Implementation of Algorithms Exploiting Properties of the Delay-Disruption Tolerant Networks

PHD DISSERTATION

Dimitrios Vardalis

Xanthi, July 2015

Acknowledgments

The 5-year period towards this PhD dissertation has been an exciting road, albeit an especially tough one. Pushing through the obstacles and difficulties would not be possible without proper motivation and support. I hereby take the opportunity to acknowledge the people who have, directly or indirectly, contributed to the completion of the present work.

First of all, I would like to thank my advisor and friend Professor Vassilis Tsaousidis, both for giving me the opportunity of this PhD candidacy as well as for providing the necessary guidance and motivation during the course of my studies. Vassilis has been to me a role model for reasons that extend beyond his renowned academic excellence, including his determination to stand up for what is just, often at personal cost.

I would also like to thank the rest of my advisory committee members, Professors Paul Spirakis and Pavlos Efraimidis, for promptly attending to all necessary academic and administrative issues with respect to my PhD degree. Professor Efraimidis' contribution has been especially valuable both for supplying the inspiration for part of this work and for assuming a major portion of the administrative overhead. I would also like to thank Professor Vasilios Katos for the constructive collaboration, the outcome of which has found its way into this dissertation. Additionally, I would like to express my gratitude to the members of my dissertation committee, Professors Alexander Karakos, Eli Katsiri, Fotini-Niovi Pavlidou and Carlo Caini for the evaluation of my work.

Next I would like to thank my colleagues at DUTH: Sotiris-Angelos Lenas, Sotiris Diamantopoulos, and Sofianna Menesidou for their fruitful collaboration in our joint research efforts; Agapi Papakonstantinou for her tireless back-office support; and Yannis Komnios, Nikos Bezirgiannidis, George Papastergiou, and Lefteris Mamatas for frequently sharing their scientific ideas and outlooks.

Last, but certainly not least, I would like to thank my partner in life Vasso for her indispensable support during the stressful times and her companionship at the cheerful moments in the course of this work.

Table of Contents

1. Introduction 1

1.1. Subject of Study 1

1.2. Problem Definition 2

1.3. Solution Approach – Contribution 6

1.4. Chapter Preview 12

2. Research Background, Motivation and Related Work 13

2.1. Delay-Disruption Tolerant Networking (DTN) 13

2.1.1. DTN Motivation 13

2.1.2. Delay-Disruption Tolerant Architecture 15

2.1.3. Convergence Layers 19

2.2. Energy-Efficient Networking 21

2.2.1. The IEEE 802.11 Standard 22

2.2.2. Energy-Saving Protocols 26

2.2.3. Bandwidth Estimation 28

2.3. Space-Data Dissemination 30

2.3.1. LEO Missions Overview 30

2.3.2. Reliable Satellite Broadcasting 33

2.3.3. Space-Data Dissemination Schemes 33

2.4. Opportunistic DTN Routing 34

2.5. Simulation Environments 35

2.5.1. Simulators Overview 35

2.5.2. The ns-2 Simulator 36

2.5.3. Space-Networking Simulation Tools 37

3. Energy-Efficient Networking with DTN 40

3.1. Energy-Efficient DTN Overlay 40

3.1.1. Data Packaging 41

3.1.2. Overlay Routing 42

3.1.3. Cross-Layer operations 43

3.2. Buffering Energy-Saving Potential 44

3.2.1. Analysis Assumptions 44

3.2.2. Mathematical Formulation 45

3.2.3. Energy-Saving Performance 47

3.3. The Rendezvous Mechanism 48

3.3.1. Base Station Operation 48

3.3.2. Mobile Receiver Operation *50*
3.4. Bandwidth Estimation in 802.11 Wireless LANs 50
3.4.1. Medium State Monitoring *51*
3.4.2. Bandwidth Utilization Calculations *52*

4. Decentralized Space-Data Dissemination with DTN 53

4.1. Overview 53
4.2. Mission Requirements 54
4.3. Space-Link Models 56
4.4. Data Delivery Models 57
4.4.1. Network Elements *57*
4.4.2. Ground Distribution Models *57*
4.4.3. End-to-End Models *59*
4.5. Data Production and Transmission Scheduling 60
4.5.1. Application Data Production *60*
4.5.2. Transmission Scheduling *61*
4.5.3. Supporting Mechanisms *62*

5. Storage-Constrained Routing in Opportunistic DTNs 63

5.1. Network Topology 63
5.2. Storage-Constrained Routing 64
5.3. Bundle Acceptance Criteria 65

6. Experimental Methodology 67

6.1. Energy-Efficient DTN Overlay 67
6.1.1. Energy Expenditure Calculations *69*
6.1.2. Stage 1 (Proof-of-Concept) *71*
6.1.3. Stage 2 (Full DTN Overlay) *73*
6.1.4. Stage 3 (Simple DTN Overlay) *75*
6.2. Decentralized Space-Data Dissemination with DTN 76
6.2.1. Performance Metrics *77*
6.2.2. Physical Simulations *78*
6.2.3. Satellite Operation *78*
6.2.4. Ground Network *80*
6.3. Storage-Constrained Routing in Opportunistic DTNs 82
6.3.1. Simulation Setup *82*
6.3.2. Performance Metrics *83*
6.3.3. Test Plan *84*

7. Simulating Networking Protocols 85

7.1. Basic DTN Model 85

7.1.1. Overall Architecture 85

7.1.2. Bundle and Bundle Managers 87

7.1.3. Transport Agents 90

7.1.4. Connection Management 91

7.1.5. DTN Agent 92

7.1.6. Auxiliary TCL procedures 94

7.2. Simulation Model Energy Extensions 95

7.2.1. Rendezvous Mechanism 95

7.2.2. 802.11 Bandwidth Estimation 97

7.3. Simulation Model Space Extensions 98

7.3.1. Satellite Reliable Broadcasting 98

7.3.2. Ground Peer-to-Peer Multicasting 100

7.4. Other Simulation Model Extensions 103

7.4.1. DTN Storage-Based Routing Extension 103

7.4.2. DTN Cryptographic Operations 105

8. Experimental Results 109

8.1. Energy-Efficient DTN Overlay 109

8.1.1. Stage 1 (Proof-of-Concept) 109

8.1.2. Stage 2 (Full DTN Overlay) 117

8.1.3. Stage 3 (Simple DTN Overlay) 122

8.2. Decentralized Space-Data Dissemination with DTN 130

8.2.1. Small-Scale Simulations 130

8.2.2. Investigating System Dynamics 132

8.2.3. Broadcast Simulations 137

8.3. Storage-Constrained Routing in Opportunistic DTNs 140

8.3.1. Baseline Simulations 141

8.3.2. History-Based Simulations 142

9. Afterword 145

9.1. Energy-Efficient Networking with DTN 146

9.2. Decentralized Space-Data Dissemination with DTN 148

9.3. Storage-Constrained Routing in Opportunistic DTNs 150

9.4. DTN Simulation Model Development 150

References 153

List of Figures

Figure 2-1: An example DTN protocol stack including the TCP/IP layers and the reliability signals. Source [66]. 20
Figure 2-2: An example DCF CSMA/CA medium access. Source [4]. 24
Figure 2-3: An example frame exchange sequence including the RTS/CTS and ACK frames. Source [4]. 25
Figure 2-4: GEO relay satellite operation. 31
Figure 2-5: QB50p1 and QB50p2 of the QB50 precursor mission. Source: [109]. 32
Figure 3-1: Energy-efficient DTN overlay. 40
Figure 3-2: ADUs, bundles and segments. 41
Figure 3-3: Rhombus-like network topology. 42
Figure 3-4: Example state transitions of a 3*Buffer Size transfer when OutRate equals to 2*InRate. 46
Figure 3-5: Minimum overlay deployment at Source, Base Station and Mobile Receiver. 48
Figure 3-6: Bandwidth estimation utilization timeslots and corresponding weights. 52
Figure 4-1: Satellite data broadcasting with P2P multicast ground distribution. 54
Figure 4-2: Categorization of Earth Observation applications depending on the type of space-data they consume. 55
Figure 4-3: Example of a single type-of-service multicast distribution tree per ground station. 58
Figure 4-4: End-to-end architecture and indicative protocol stacks for space and ground. 60
Figure 4-5: Transmission scheduling example. 61
Figure 5-1: Partial mesh topology creating mutual reliance among top and bottom nodes. 63
Figure 6-1: Proof-of-concept topology. 72
Figure 6-2: Full DTN overlay topology. 73
Figure 6-3: Simple DTN overlay. 75
Figure 6-4: Simulation topology. 83
Figure 7-1: DTN Agent Class Diagram. 86
Figure 7-2: DtnAgent Class Message Receiving Function Pseudocode. 93
Figure 7-3: Rendezvous Mechanism Initial Data Reception at the BS Pseudocode. 95
Figure 7-4: Selection of the Next Bundle for Transmission Pseudocode. 100
Figure 7-5: Ground Network Route Setup Pseudocode 101
Figure 7-6: Two-tier Ground Network Route Setup Pseudocode 102
Figure 7-7: Alternative Data Transfer Route from Mart to Earth. Source 105
Figure 7-8: Stages of the Data Transfer from Source to Destination. 106
Figure 7-9: Incoming Data Control Flow at the DTN Agent. 107
Figure 7-10: Crypto-timer Expiration Control Flow at the DTN Agent. 107
Figure 8-1: FTP, Varying Bandwidth, No Competing Flow Energy Expenditure (E2E, Split) and Potential Energy Expenditure (P-E2E, P-Split). 109
Figure 8-2: FTP, Varying Bandwidth, Competing Flow Energy Expenditure (E2E, Split) and Potential Energy Expenditure (P-E2E, P-Split). 111
Figure 8-3: FTP, 2Mbps Bandwidth, Competing Flow State Transitions. 112
Figure 8-4: FTP, Varying Wireless PER, No Competing Flow Duration. 112
Figure 8-5: FTP, Varying Wireless PER, No Competing Flow Energy Consumption. 113
Figure 8-6: FTP, Varying Wireless PER, Competing Flow, Energy Consumption. 113

Figure 8-7: FTP, Varying Target Buffer, FTP, With and Without a Competing Flow, Energy Consumption. 114
Figure 8-8: Actual Buffer Occupancy at 60 KB Target Buffer Occupancy. 115
Figure 8-9: CBR, Varying Buffering Threshold Energy Consumption. 116
Figure 8-10: Actual Buffer Occupancy at 60 KB Target Buffer Occupancy. 116
Figure 8-11: Duration of a Single-bundle Transfer with Varied Bandwidth. 118
Figure 8-12: Energy Expenditure of a Single-bundle Transfer with Varied Bandwidth. 118
Figure 8-13: Single-bundle Transfer with Varied Target Buffer Occupancy, Energy Expenditure. 119
Figure 8-14: Single-bundle Transfer with TBO of 30 and 60 KB, Actual Buffer Occupancy. 119
Figure 8-15: Single-bundle Transfer with TBO of 30 and 60 KB, Sleep/Active state transitions. 120
Figure 8-16: Multi-bundle Transfer with Varied Bandwidth, Duration. 120
Figure 8-17: Multi-bundle Transfer, Varied DTN Storage Space, Duration. 122
Figure 8-18: FTP with no competing traffic energy and delay vs. the TBO. 123
Figure 8-19: FTP energy expenditure with all types of competing traffic. 124
Figure 8-20: CBR with no competing traffic energy and delay vs. the TBO. 126
Figure 8-21: CBR energy expenditure with all types of competing traffic. 126
Figure 8-22: HTTP with no competing traffic energy and delay vs. the TBO. 128
Figure 8-23: HTTP energy expenditure with all types of competing traffic. 129
Figure 8-24: Delivery ratio and average delivery latency vs. ADU size. 131
Figure 8-25: Delivery ratio and average delivery latency vs. transmission interval. 132
Figure 8-26: Direct point-to-point delivery ratio and latency varying data TTL. 135
Figure 8-27: Direct point-to-point delivery ratio and data volume varying the daily data production. 135
Figure 8-28: Low-cost broadcast delivery ratio and latency varying data TTL. 136
Figure 8-29: Low-cost broadcast delivery ratio and data volume varying the daily data production. 136
Figure 8-30: Low-cost broadcast delivery ratio and data volume varying 137
Figure 8-31: Low-cost broadcast delivery ratio and delivery latency varying the number of ground stations. 138
Figure 8-32: Low-cost broadcast delivery ratio and data volume varying the number of satellites. 138
Figure 8-33: Delivery ratio for broadcast plain, acknowledged and limited transmission varying the daily data production. 139
Figure 8-34: Delivery latency for broadcast plain, acknowledged and limited transmission delivery latency varying the daily data production. 139
Figure 8-35: Low-cost broadcast centralized vs. P2P ground data distribution delivery ratio and delivery latency, varying the daily data production. 140
Figure 8-36: Delivery ratio for all nodes in the deterministic bundle acceptance simulations. 141
Figure 8-37: Delivery ratio for all nodes in the non-deterministic bundle acceptance simulations. 142
Figure 8-38: Delivery ratio for the rejected bundle acceptance mechanisms. 143
Figure 8-39: Delivery ratio for the accepted bundle acceptance mechanism. 144

List of Tables

Table 6-1: Effect of bundle size and type-of-error on the delivery ratio. 79
Table 6-2: Node degree distribution 81
Table 6-3: Topology depth distribution 81
Table 8-1: Varying Bandwidth, No Competing Flow Transfer Duration. 109
Table 8-2: FTP, Varying Bandwidth, Competing Flow Transfer Duration. 111
Table 8-3: No Competing Flow, Buffer Occupancy Values (KB) 114
Table 8-4: FTP with no competing traffic, sleep and transition information. 123
Table 8-5: FTP delay with all types of competing traffic. 124
Table 8-6: CBR with no competing traffic, sleep and transition information. 126
Table 8-7: CBR delay with all types of competing traffic. 126
Table 8-8: HTTP with no competing traffic, sleep and transition information. 128
Table 8-9: HTTP delay with all types of competing traffic. 129
Table 8-10: Effect of the bundle size 131
Table 8-11: Contact Opportunities Statistics 133
Table 8-12: Performance metrics in the deterministic bundle acceptance simulations. 141
Table 8-13: Performance metrics in the non-deterministic bundle acceptance simulations. 142
Table 8-14: Performance metrics for the rejected bundle acceptance mechanisms. 143
Table 8-15: Performance metrics for the accepted bundle acceptance mechanism. 144

List of Abbreviations

ADU	Application Data Unit
AP	Access Point
BAR	Bundle Acceptance Ratio
BE	Bandwidth Estimation
BER	Bit Error Rate
BP	Bundle Protocol
BS	Base Station
BSR	Bundle Status Report
BxD	Bandwidth-Delay Product
CBR	Constant Bit Rate
CFDP	CCSDS File Delivery Protocol
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CT	Cross-Traffic
CW	Contention Window
DCF	Distributed Coordination Function
DIFS	Distributed InterFrame Space
DTN	Delay-Disruption Tolerant Networking
E2E	End-to-End
EID	Endpoint Identifier
EU	End-user
FTP	File Transfer Protocol
GEO	Geosynchronous Equatorial Orbit
GS	Ground Station
HTTP	Hypertext Transfer Protocol
IFS	InterFrame Space
IP	Internet Protocol
IPN	InterPlanetary Networking
LAN	Local Area Network
LEO	Low Earth Orbit
LTP	Licklider Transmission Protocol
MAC	Medium Access Control
MANET	Mobile Ad-hoc Network
MCC	Mission Control Center
MEO	Medium Earth Orbit
MR	Mobile Receiver
MRG	Minimum Reception Group
NAV	Network Allocation Vector
P2P	Peer-to-Peer
PDU	Protocol Data Unit
PER	Packet Error Rate
PI	Principal Investigator
PSM	Power-Save Mode
RTS/CTS	Request-To-Send/Clear-To-Send
RTT	Round-Trip Time
SIFS	Short InterFrame Space
TBO	Target Buffer Occupancy
TCP	Transmission Control Protocol

TCPCL	TCP Convergence Layer
TIM	Traffic Indication Map
TM/TC	Telemetry/Telecommand
ToS	Type-of-Service
TTL	Time-to-Live
UDP	User Datagram Protocol
UDPCL	UDP Convergence Layer
UGS	Uplink Ground Station
URI	Uniform Resource Identifier
WLAN	Wireless Local Area Network
WNIC	Wireless Network Interface Card
WSN	Wireless Sensor Network

1. Introduction

1.1. Subject of Study

In the present work towards a PhD degree we study Delay-Disruption Tolerant Networking (DTN) [1] applications and algorithms, addressing problems mainly related to traditional wired and wired-cum-wireless infrastructure networks. The DTN computer networking architecture was designed to cope with long propagation delays and lack of continuous end-to-end connectivity, as well as to provide the necessary framework for interconnecting heterogeneous computer networks. Despite its original conception as a space-communications architecture, DTN has been proposed for applications in various settings, including wired or wired-cum-wireless networks on the edges of the well-connected Internet. In this context, we extend DTN and apply the resulting architecture on a number of distinct use-cases, while investing significant effort in developing detailed simulation models necessary for evaluating application feasibility and performance. The proposed solutions share the characteristic of deploying DTN as an overlay over traditional IP [2] networks, engaging well-known protocols, such as the Internet transport protocols TCP [3] and UDP [4] and, in certain cases, interacting with the de facto standard for wireless LANs 802.11[5].

In a typical deployment, DTN can be setup as an overlay on top of existing networks (i.e. IP networks) and provide delay-disruption tolerant services among different segments of the underlying network. Resilience to long delays and connectivity interruptions is ensured by permanently storing protocol state and in-flight protocol data, enabling seamless operation across machine restarts. Responsibility for reliably delivering data is passed among nodes along the end-to-end path via the custody mechanism, a coarse-grained retransmission capability provided by DTN. Through the described core mechanisms, the restrictions imposed by traditional networking protocols are relaxed and new application possibilities are created in a variety of circumstances, far beyond the area of space communications DTN was originally designed for.

The DTN architecture breaks the synchronous, source-to-destination connectivity constraint, imposed by traditional networking protocols, and provides asynchronous services, utilizing any of a wide range of data transportation methods. A prominent such example of asynchronous service provision is the use of DTN to bring Internet connectivity to regions lacking the required infrastructure, thus, reducing the digital divide between developed and developing areas. The DakNet [6] network provides a solution to this problem by physically transferring data using vehicles equipped with Wi-Fi devices and permanent storage. Vehicles visit disconnected areas lacking access to certain governmental services, collect data (e.g. administrative applications), and travel back to locations with Internet connectivity where they unload the collected data. In a similar scenario, temporary connectivity may be provided via satellite links, only for certain pre-determined intervals. Significant attention from the research community has also been drawn by DTN applications related to vehicular communications [7][8][9] and pocket-switched networks [10], which target at providing Internet connectivity to commuters and mobile users respectively, avoiding the use of cellular data connections. More recently, DTN has also been combined with resource pooling techniques

that share private broadband connections in order to provide Less-than-Best-Effort, free Internet services to all users [11].

Similarly to the applications described in the previous paragraph, we propose solutions for contemporary networking problems, exploiting the inherent capability of DTN to form networking overlays and provide non-volatile data storage. The first DTN application in our work targets at improving energy efficiency of mobile devices in an infrastructure, wired-cum-wireless Internet setting with a last-hop 802.11 connection. The inherent store-and-forward functionality of DTN is utilized in order to buffer incoming data at the base station and allow the mobile device to conserve energy by switching its network interface to sleep mode, exploiting possible excess capacity of the wireless as compared to the broadband link.

In the second application DTN is utilized for disseminating space-data in a decentralized manner to end-users connected to the Internet. Satellites transmit data through a best-effort, broadcasting mechanism of varying reliability, which can be adjusted according to the mission requirements. On the ground, space-data are disseminated over a DTN overlay, following a peer-to-peer multicast approach. Speedy delivery is achieved due to both the creation of shortest routes to the interested recipients, as well as due to offloading the network from duplicate data transmission.

The third application involves a preliminary study of storage-constrained routing in an opportunistic DTN setting, where persistent storage for accepting custody of relay bundles is scarce. The aim of the routing scheme is to achieve efficient and fair resource allocation, in a distributed setting, without the need for a central authority. The mechanisms employed by the routing scheme are inspired from Game Theory and routing is considered as a game, where nodes seek to achieve delivery for locally created data.

The simulation tools necessary to support development of the above-mentioned solutions have been implemented in great detail in the ns-2 network simulation environment [12]. The simulation model implements functionality across the entire protocol stack, namely at the physical, link, MAC and transport layers. The developed simulation framework has also been employed in order to support peripheral experiments such as bandwidth estimation in 802.11 wireless LANs and cryptographic operations in DTN.

1.2. Problem Definition

The original concepts behind DTN [13] were part of the work for the InterPlanetary Networking (IPN) and stemmed from the space-communications requirements of tolerating long delays and operating over certain, previously-known, contact patterns. The IPN later evolved into DTN, with a clear aim to encompass other types of networks, especially terrestrial wireless networks. DTN can be viewed as a natural outcome of the evolution in the Computer Networking field and, due to that, has enjoyed wide recognition from the research community as the networking technology of the future.

Prior to the appearance of DTN, a significant portion of the research effort in the computer networking field was devoted in the areas of Mobile Ad-hoc Networks (MANETs) and Wireless Sensor Networks (WSNs). The problems that researchers attempted to solve were mainly related to routing and energy conservation. Occasionally, the wireless networks would include a “data mule” [14] that physically carried

data from one isolated network to another, overcoming the problem of a possible lack of end-to-end connectivity and facilitating asynchronous data transfers. The DTN architecture incorporated opportunistic networking (e.g. MANETs and WSNs) and inherently supported the "data mule" concept, providing an architectural "umbrella" able to unify disparate networking islands. Similar unification under the DTN term has been, to a great extent, realized in the related research fields.

The basic message service of DTN is specified by the Bundle Protocol (BP) [15], the Protocol Data Unit (PDU) of which is appropriately termed the bundle. The BP describes the end-to-end protocol, block formats, and abstract service description for exchanging messages in DTN, and provides a store-and-forward overlay service, similar to that of an IP network, but with an emphasis on storing, rather that forwarding data. The requirement of permanent protocol storage adds an extra dimension to the routing problem in DTNs, that of the storage availability.

The responsibility for reliable delivery of a bundle is delegated through the custody transfer mechanism, which typically involves moving the bundle nearer the destination along with the responsibility for retransmission in case of loss. DTN facilitates bridging of heterogeneous networks via specialized convergence layers, allowing nodes to operate as gateways and offering routing and interoperability services among different networks.

DTN has been eagerly embraced by the computer networking research community due to its intrinsic characteristic of interconnecting heterogeneous networks and coping with long delays and interrupted connectivity. Inspired by these qualities, we employ DTN in order to address a number of problems only broadly related to the original role of the architecture as a space communications technology. The rest of this section includes the problem definitions along with basic background information and related work regarding the main research directions of the present study.

Energy-Efficiency in Mobile Networking Devices

Mobile devices have become a powerful tool for communication, storage and entertainment, and often undergo several hours of daily use, which stretches their energy storing capabilities. They are equipped with increasingly more powerful CPUs, larger memory capacities, faster wireless network adapters, and a set of applications that require internetworking capabilities. However, advances in battery technology have not been able to match the increased energy demand [16].

The wireless networking subsystem of a mobile device has been recognized as a major culprit in draining energy, accounting for up to 60% of the overall device energy consumption for network intensive applications [17]. Thus, energy efficiency of network protocols throughout the protocol stack has drawn significant attention from the research community [18]. The ubiquitous 802.11 standard [5] confronts the energy efficiency problem by providing a mechanism that buffers incoming data at the access point, allowing the mobile devices to temporarily switch their wireless interfaces to the sleep state in the meantime. However, the energy conservation potential of this mechanism is limited by the probabilistic

nature of incoming data as well as the relatively small buffer space at the access point. This has led many researchers into looking for alternative methods based, however, on the same core principle. The solutions proposed in the related literature typically involve buffering at a "base station", rather than an "access point", highlighting that the device operation extends in higher network layers (the device also assumes the role of a router). We respect this distinction and generally use the term base station, with the exception of references to the 802.11 specifications where the device operation is confined within the Physical and Data link layers of the OSI model.

Our solution to the energy-efficiency problem demonstrates the potential of DTN to shape internetwork traffic in a manner that allows mobile devices to balance their energy expenditure with minimal cost on throughput. In this context, we design and deploy an energy-efficient, internetworking overlay that exploits two major DTN properties: (i) to store packets for as long as it is necessary, regardless of disruptions in communications; and (ii) to enhance the edge nodes with functionality to wait for sufficient amount of packets to arrive, prior to transmitting to the end node. We then develop scenarios and weigh our expectations using a broad set of experiments that reflect potential variations in traffic characteristics, error probabilities and disruptions.

Efficient Space-Data Dissemination

Space agencies and research institutes organizing Earth observation missions worldwide focus on employing Low Earth Orbit (LEO) satellites, due to their fine observation capabilities provided by their close proximity to the Earth surface. Major space agencies typically employ high-end LEO satellites and corresponding ground stations, which communicate over high-speed, bidirectional links. Scientific data are usually collected at a central location and then distributed to the end-users. In special cases, scientific data may be additionally broadcast directly to end-users with suitable equipment. Some indicative examples of typical high-end LEO satellite missions, operated nowadays by major space agencies, include NASA's EOS Terra, Aqua and NPP missions [19] and ESA's Sentinels [20].

Recent trends in aerospace engineering include the development of smaller, low-cost satellites, enabling institutes with limited budget to build their own spacecraft and, also, allowing for the deployment of highly populated constellations (termed "satellite swarms") [21]. Likewise, corresponding ground communication equipment becomes inexpensive and easy to setup, allowing for extensive deployment that leads to more frequent contact opportunities, albeit of considerably lower capacity. In this spirit, the QB50 mission envisages a low-cost, distributed LEO satellite network [22], allowing universities and research institutes to deploy their own scientific equipment in space.

Emerging space technologies are bound to extend space-data access to a much wider audience [23], but also, considering the complex network topology and the vast amounts of scientific data produced by modern satellite equipment, significantly perplex the space-data transmission and ground management. These new challenges have inspired the European Space Agency (ESA) to explore alternative space-data

dissemination paradigms for low-cost, dense satellite networks that will be able to cope with the immense amounts of data collected by modern scientific equipment.

Our original work in this topic had been conducted in response to ESA's statement of work [24] with the collaboration of Telespazzio Vega [25] as our project partner. The study consisted of a theoretical part, which included a feasibility analysis on protocol applicability, as well as an experimental part, which included the development of detailed simulation tools and the evaluation of the proposed solutions. The outcome of the study, including detailed documentation of the simulation tools, was demonstrated and delivered to ESA for future use. After the ESA study was concluded, we extended the original work, independently of the project partner, producing the complete proposal as it is presented in this dissertation.

Our proposal builds on a constellation of small, inexpensive satellites, broadcasting scientific data to low-cost ground terminals. We suggest a broadcast, peer-to-peer (P2P) return model for satellite data based on the DTN architecture, and an end-to-end, from space to the end-user, simulation model for evaluating the proposed solution. The proposed architecture exploits low-cost space and ground assets to deliver space-data to interested end-users. The versatile transmission scheduling mechanism is appropriate for both unidirectional, as well as bidirectional links, and also, supports asynchronous delivery feedback through designated ground stations.

Routing in Opportunistic DTNs

Routing in opportunistic DTNs, formed by mobile devices in an ad-hoc fashion, is a popular topic among researchers in the networking community. Due to the absence of infrastructure, individual nodes are usually required to contribute their limited resources to support the overall network operation. Individual resources that may need to be contributed are usually related to energy, bandwidth and storage. Recently, seminal work has emerged on studying DTN routing from a game theoretic standpoint. The authors in [26] attempt to model node contacts in a DTN as a Game Theory problem and derive forwarding decisions assuming different game type categories (strategic game, extensive game etc.). In [27] Shevade et al. suggest employing mechanisms from Game Theory in order to achieve efficient and fair routing in DTNs.

Our work expands on this seminal research by experimentally studying the application of mechanisms inspired from Game Theory on DTN routing, where node storage space is scarce. Due to the preliminary nature of our work on this topic, the relevant simulation study is conducted on a fixed topology. However, the proposed mechanisms are applicable in opportunistic settings as well. The motivation behind employing these mechanisms is to achieve efficient and fair network resource allocation, without the need for a central coordination authority.

Simulator Inadequacy for DTN over Internet Protocols

In order to facilitate the development of the previously described DTN applications it was necessary to employ a set of network simulation tools. Due to the nature of our proposed solutions, a detail simulation of the entire protocol stack was vital for the validity of our results. However, most of the available, at the time of writing, DTN-related simulators focus on routing issues and provide only coarse implementations of the underlying protocols.

In our quest for suitable network simulators three were the main candidates that were initially identified: ns-2 [12], Opportunistic Network Environment simulator (ONE) [28] and OMNet++ [29]. Ns-2 and OpNet++ are both general-purpose tools, whereas ONE is a dedicated simulator for DTN-related protocols in opportunistic network environments. ONE mostly focuses on the routing aspects of DTNs and provides only coarse abstractions of lower layer protocols.

Our decision was to pursue an in-house implementation of the store-and-forward functionality of DTN in the ns-2 network simulation environment. The selection of ns-2 was founded on the fact that traditional IP networks are the primary target for deploying our DTN-based solutions and that ns-2 has a well-established reputation in these environments. Also, ns-2 provides detailed implementations of the physical and MAC layers of the 802.11 protocols, a crucial requirement for the thorough implementation of the energy-efficient DTN overlay and the accurate measuring of the energy expenditure at the wireless 802.11 interface.

1.3. Solution Approach – Contribution

Each of the problems defined in the previous section was tackled through a combination of a bottom-up confrontation of the practical challenges, and a top-down study of the overall context. The common denominator in all proposed solutions has been the implementation of detailed simulation models in order to attest the applicability of our proposal in real-world situations. Crucial to this premise has been the use of the ns-2 simulation environment, which, due to elaborately implementing the entire protocol stack, encourages detailed modeling, highlighting the practical issues present on real systems. The following subsection includes an overview of the core DTN simulation model, while the rest of the subsections expand on the solutions to the problems described in section 1.2.

DTN Simulation Model Development

The solutions proposed in this dissertation were developed with the help of detailed simulation experiments. As there was no established DTN implementation in ns-2, we developed our own DTN simulation model, which addresses design and implementation issues beyond a simple proof-of-concept demonstration. The model provides great control over the simulated functionality and enables the study of various issues related to DTN, including custody acceptance and rejection reporting, data storage

distribution, bundle sizing, retransmission strategy selection, and routing strategy selection. Additionally, the model facilitates experimentation with storage management and storage occupancy distribution along the network path. To the best of our knowledge, the developed DTN model is, at the time of writing, the only available simulator, implementing the bundle layer over detailed models of the underlying protocols. The DTN model is publicly available to the research community from this link [30].

The DTN simulation model comprises a set of classes implemented in the ns-2 network simulator framework. The model assumes a static topology for the wired part of the network and a dynamic, last-hop link that may be enabled or disabled during simulation, accordingly affecting the overlay routing. Initial experimentation during the development of the model showed that cross-layer interaction between DTN and the underlying transport protocol could significantly enhance protocol efficiency, especially under stressed network conditions. This observation led to the creation of a number of cross-layer query and callback mechanisms, mostly setup between DTN and TCP. The central entity of our simulation model is the DTN agent, which acts as a store-and-forward module between the application and the transport layers. A single type of a DTN agent is used for all purposes (i.e. as a source, relay, or destination), supporting both TCP and UDP transport between adjacent overlay nodes. Depending on the individual application needs the agent may interact with simulator entities at various networking layers, such as the WNIC of a mobile device (physical layer), the queue of an outgoing link (link layer), the 802.11 agent (MAC layer), the underlying transport agent (transport layer).

Energy-Efficient DTN Overlay

During operation of a mobile device the 802.11 Wireless Network Interface Card (WNIC) can be, at any given moment, in one of five states, each with different power requirements: *transmit*, *receive*, *idle*, *sleep* and *power-off*. The former three states (i.e. transmit, receive and idle) are the active states, whereas the latter two states (i.e. sleep and power-off) are the inactive states. The power requirements of the inactive states are much lower than those of the active states and, therefore, the challenge for energy-saving strategies translates into how to maximize the duration that the WNIC spends in these states. From the two inactive states, power-off is unsuitable for frequent transitions due to the considerable time required to re-enable the WNIC after a complete shutdown (requires system-level actions and possibly rejoining the Wireless LAN (WLAN) [31]). Therefore, energy-conserving protocols are concerned with utilizing the sleep state rather than the power-off state during operation.

The key to limiting the energy expenditure during data transfers is to maximize the sleep duration of the WNIC on a Mobile Receiver (MR) by buffering incoming data at the Base Station (BS), essentially shaping traffic in bursts and prolonging periods of inactivity. This method targets incoming traffic, which constitutes the vast majority of traffic in WLANs, exploiting possible excess capacity of the wireless link compared to the broadband Internet connection (a typical scenario in 802.11 deployments). Due to the limited buffer capacity at the BS, excessive buffering may cause packet drops and costly retransmissions

leading to throughput degradation. Also, buffering increases the data delivery latency, which may hinder operation in certain types of applications. Thus, a major consideration during the design of the proposed energy-efficient internetworking overlay has been to limit the performance penalty imposed by the buffering mechanism.

Storing relay traffic for as long as necessary is an inherent quality of DTN, making it an ideal match for implementing such a buffering scheme. We achieve energy conservation for mobile devices with the deployment of a DTN internetworking overlay, which includes, at a minimum, the source, the BS and the MR. The DTN functionality has been extended with a rendezvous mechanism employed between the BS and MR DTN nodes, for data transfers that originate on the Internet and target a mobile device on an 802.11 LAN. Incoming data is buffered at the DTN agent residing on the BS and flushed at certain rendezvous times, allowing the receiver to switch its wireless interface to sleep mode in the meantime. At each rendezvous the BS calculates the time for the next rendezvous and communicates this information as part of the BP header in the last bundle of the flushed data burst. The MR can safely switch its WNIC to the sleep state until the next rendezvous, without running the risk of missing any incoming data. Optionally, the DTN overlay may include additional nodes along the network path, so that data can be advanced towards their final destination at the MR even in case of buffer shortage at the BS, thus exploiting the collective network storage availability.

The amount of data buffered between the edge node and the receiver can be externally affected through the Target Buffer Occupancy (TBO) setting of the rendezvous mechanism. The rendezvous mechanism measures the incoming data rate and calculates the next rendezvous time so that the amount of collected data converges with the TBO. Due to the statistical nature of Internet traffic, the incoming data rate can be highly variable (especially when congestion controlling transport protocols, such as TCP, are employed), causing fluctuations to the actual amount of buffered data. The devised rendezvous mechanism achieves adequate convergence with the TBO, allowing for effective buffer planning, and suffering only occasional peaks of the actual buffered data amount (mostly during TCP slow start).

In order to develop and test the proposed solution, the DTN simulation model was extended with the required rendezvous functionality between the BS and the MR. Implementation of the required functionality at the BS involved pausing data relaying during sleep intervals and flushing at certain rendezvous times. At the MR, required functionality involved switching the WNIC to the sleep mode and back to active mode when necessary. In both cases, interactions with the transport layers were essential. The energy expenditure on the wireless node was monitored using the provided ns-2 energy model.

Overall, the simulated scenarios have shown that for large file transfers the energy-saving mechanism can conserve as much as 70% of the required energy introducing only minor delays, when compared to the traditional end-to-end TCP scheme. The transfer duration in these cases may even decrease under heavy congestion, as a side-effect of splitting the TCP connection into multiple segments. For other types of traffic, such as CBR or HTTP, the conserved energy may reach well above 50%, introducing, however, considerable packet-level delays of 50-100%.

Bandwidth Estimation in 802.11 Wireless LANs

During development of our energy-efficient internetworking overlay we experimented with mechanisms that, even though did not make it to the final design, could be of use in their own right, or even incorporated in a future version of the overlay. One such mechanism, worth describing in this dissertation, is a Bandwidth Estimation (BE) technique for 802.11 WLANs that gives greater visibility to the BS, allowing for more efficient scheduling decisions. This BE mechanism passively monitors the wireless network activity at the BS and calculates the channel utilization, taking into account the inner workings of the 802.11 MAC protocol.

Bandwidth estimation techniques can be broadly categorized as active and passive. Active BE injects probing traffic into the network, measures the network response and uses these measured quantities in order to calculate available bandwidth on the desired network path. Passive BE relies on passively listening to network traffic in order to calculate the available bandwidth. Passive techniques in wireless networks, such as the 802.11, take advantage of the broadcast nature of the communication by snooping on in-range transmissions and calculating idle periods for the medium.

Our BE mechanism records busy periods for the medium, as well as idle periods that are part of the collision avoidance strategy of the 802.11 MAC protocol. More specifically, periods of inactivity that are part of the Virtual Carrier Sense mechanism of the 802.11 MAC, or certain InterFrame Spaces (IFSs) (i.e. protocol imposed idle time between frames) are considered as busy intervals. The busy and total duration amounts are stored in appropriate data structures that allow for weighted contribution of the measured utilization depending on the relative age of the measurement. Higher layer entities in the protocol stack may query the 802.11 layer and receive channel utilization information, which, combined with the nominal channel capacity yields the available residual bandwidth.

Decentralized Space-Data Dissemination with DTN

Building on the concept described in ESA's statement of work [24] we proposed a broadcast, peer-to-peer (P2P) return model for satellite data based on the DTN architecture, and created an end-to-end, from space to the end-user, simulation model for evaluating the proposed solution. The main characteristics of such a network paradigm are two-fold. Firstly, due to satellite and/or ground station hardware limitations many, if not all, contacts are of a unidirectional, space-to-ground, nature. Only certain designated ground stations, in control of the satellite operation, may provide uplink functionality through which, data delivery feedback can be uploaded. Secondly, a large number of end-users scattered all over the globe may be subscribed to receiving the collected space-data, some of who may be operating their own ground stations. In such a setting, it would be more efficient for end-users to receive the collected data directly from the receiving ground stations, rather than via some centrally located repository, as it is normally the case with current space-data dissemination schemes.

Our proposed paradigm includes a best-effort, broadcast transmission mechanism with optional delivery feedback that can be deployed over well-known space-data transmission protocols [32]. Data acquisition rate is assumed to be less than or equal to the available bandwidth, so that all data receive at least one chance for transmission. The transmission scheduling algorithm exploits possible excess bandwidth for selectively retransmitting data by employing a randomized retransmission mechanism. This transmission pattern allows for adjusting the balance between the volume and the ratio of data received on the ground, even in the absence of delivery feedback. The transmission scheduling can be adjusted according to the application requirements with respect to temporal criticality (i.e. real-time vs. off-line services) and reliability (i.e. bulk vs. reliable data). Optional delivery feedback may enhance transmission efficiency by allowing the sending DTN node on the satellite to discard data that have already been received on the ground, avoiding unnecessary retransmissions. Additionally, optional information on contact opportunities may be used in order to restrict transmission during contacts, both improving transport efficiency as well as conserving valuable satellite energy.

Once on the ground, space-data are forwarded to the end-users on the Internet over a P2P multicast overlay network, which resembles a push version of the BitTorrent protocol [33]. Data are tagged with a Type-of-Service (ToS) identifier, allowing users to subscribe to the desired ToS in order to receive only relevant data, possibly produced by the scientific instruments of their preference. The subscribed nodes in each ToS form a P2P multicast network, creating source-based multicast distribution trees where the sources are the ground stations receiving the space-data.

Modeling the proposed space-data dissemination scheme was done in two stages: the physical simulation and the network simulation. The physical simulation was conducted by our project partners in Telespazzio Vega using the commercial modeling environment Systems Tool Kit (STK) by AGI corporation [34]. The network simulations were made possible via a set of extensions to our DTN model in ns-2. The new features implemented on the simulation model were quite extensive and included, among others, the transmission scheduling algorithm on the satellite, the broadcasting mechanism, the setup and teardown of the space links and the multicast routing on the ground. The simulation model was also extended to support the currently used space-data dissemination pattern, which involves bidirectional point-to-point links with immediate reception acknowledgment and centralized distribution on the ground. In the network simulator ns-2 we imported the contact characteristics, produced by the physical simulations in STK, and modeled the end-to-end topology.

Simulation experiments showed that, indeed, our versatile transmission scheduling mechanism is able to adjust data reliability even in the absence of delivery feedback, while it can utilize such feedback for improving performance when available. Furthermore, the P2P ground distribution improves both the data reliability and timeliness over a centralized method, especially benefitting applications involving high volumes of time-sensitive data. Through simulation experiments we were also able to quantify the tradeoff between data reliability and data volume, as well as the performance increase resulted from such mechanisms as delivery acknowledgments and transmission restriction during contacts.

Storage-Constrained Routing in Opportunistic DTNs

Routing algorithms in opportunistic DTNs attempt to balance the tradeoff between the benefit of the individual node vs. that of the overall network, and achieve fair and efficient resource allocation. A multitude of routing algorithms have been proposed for solving this problem; some taking a centralized (i.e. involving a central authority) and others a decentralized approach. We study the DTN routing problem from a storage availability perspective assuming a decentralized setting, in a fixed topology where nodes need to forward bundles via their peers. Even though the topology is fixed, the developed mechanisms can be applicable to opportunistic settings as well.

The development of the routing algorithm has been supported by extending our DTN agent with functionality to keep track of the acceptance history for a node's peers as well as to make the necessary bundle acceptance decisions. When a certain node receives a bundle forwarding request, it can either accept the request and store incoming data or reject the request and discard the data. The decision on bundle forwarding acceptance depends on the storage availability, the acceptance history of the origin node and a randomly selected factor. The simulation is setup as a game, where the players' strategy may be:

- *Impartial*: a bundle is accepted if enough space is available.
- *Selfish*: a bundle is accepted with a probability that depends on the available space, favoring locally created bundles over relay bundles.
- *History-based*: bundle acceptance depends on the acceptance history of the originating node.

The simulation experiments report on metrics such as the delivery time, delivery ratio and fairness when all nodes follow the same strategy or groups of nodes follow different strategy. The experiments highlighted a tradeoff between a strategy's efficiency when all nodes behave normally and the strategy's fairness when some nodes exhibit a selfish behavior. Based on the outcome of the simulations we were able to develop a bundle acceptance criterion that performs adequately well under a wide range of scenarios, providing efficiency and guaranteeing fairness when selfish nodes are present.

Simulating DTN Cryptographic Operations

Additionally to the proposed solutions, our DTN simulation model has also been employed as part of an automated key exchange protocol evaluation framework for securing DTNs. At the time of writing this dissertation, the corresponding work has been submitted for publication and is pending review (an older piece of related work can be found here [35]). A high-level tool provides a set of functions for describing key exchange protocol characteristics such as message length, number of passes and network topology. The tool checks for the protocol sanity and then creates the corresponding ns-2 simulation script, which deploys the DTN agent on a topology appropriate for the described key exchange sequence. In order to support the cryptographic operations the DTN agent was expanded to include encryption/decryption delay,

encryption/decryption size augmentation/reduction, and multi-way communications for key exchange (i.e. the reception of a message automatically produces a new message destined to the original message source).

1.4. Chapter Preview

The rest of this dissertation is structured according to the following description. In chapter 2 we present the background and motivation for our research, as well as the work that has been done in related fields by fellow researchers. In chapter 3 we present our energy-efficient DTN overlay, analyze its energy-saving potential and lay out the details of the rendezvous mechanism. Chapter 4 contains the description of the proposed decentralized space-data dissemination paradigm based on DTN, focusing on the individual components such as the transmission scheduling mechanism and the ground distribution scheme. Chapter 4 also includes an indicative categorization of various satellite missions based on their data requirements. In chapter 5 we present our preliminary study of storage-constrained routing in opportunistic DTN settings, where Game Theory-inspired mechanisms are employed. Chapter 6 contains the description of our experimental methodology including the simulator environment, the metrics and the simulated scenarios, stressing the need for the development of additional, specialized simulation tools. In chapter 7 we give detailed description of the architecture and functionality of our simulation tools. Firstly, the core of our DTN model is described and, secondly, the extensions supporting the development of each individual solution are presented. In chapter 8 we present the results of our simulation experiments with respect to the three main DTN-based solutions proposed (i.e. energy-efficient overlay, space-data dissemination and storage-constrained routing) and, finally, in chapter 9 we conclude this dissertation.

2. Research Background, Motivation and Related Work

This chapter contains background material related to the computer networking problems studied in this dissertation. Gaining deep understanding of the related fields enabled us to design solutions according to the current "state of the art" and develop simulation tools that accurately model already existing protocols. Since DTN constitutes the basis for most of this PhD work, we begin the presentation of the relevant material with the description of the DTN concepts in the immediately following section. The subsequent three sections include background information related to each of the three proposed solutions in the areas of energy-saving networking, space-data dissemination and opportunistic routing. The final section gives an overview of the simulation environments that have been considered for use or have been actually used for the development of the simulation models.

2.1. Delay-Disruption Tolerant Networking (DTN)

Delay-Disruption Tolerant Networking (DTN) [1] is a computer networking architecture that aims to address the technical issues present in challenged networking environments, as well as specify the necessary components for interconnecting heterogeneous networks. The original work towards the DTN architecture was part of a design for the InterPlanetary Networking (IPN), which addressed networking challenges in deep-space communications. The IPN design was soon extended to also encompass challenged terrestrial networks. One of the first such efforts appeared in the seminal work by Kevin Fall in 2003 [36], where the DTN term was for the first time officially coined.

This section includes a high-level description of the main DTN characteristics that are relevant to our work. Firstly, we present the motivation behind the DTN design, which has provided us with the inspiration for the proposed applications. Subsequently, we focus on certain technical aspects of DTN that are central to the implementation of the proposed solutions, namely, the overall DTN architecture and the DTN convergence layers. These technical aspects of DTN coincide, to a great extent, with the main features implemented in the developed simulation models.

2.1.1. DTN Motivation

The main networking challenges addressed by DTN are related to long link propagation delays and intermittent connectivity, implying the lack of a continuous, end-to-end path. Under such networking conditions, traditional Internet protocols performed poorly or, in many cases, completely failed [37], creating the need for a new networking paradigm. As per the relevant RFC 4838 [1], the DTN architecture may be applied to a number of network types such as sensor-based networks [38][39][40], terrestrial wireless networks lacking continuous end-to-end connectivity [41][42], satellite networks with periodic connectivity [43][44], and underwater acoustic networks [45][46].

The DTN architecture defines the "bundle layer", which is an end-to-end, message-oriented overlay between the transport and the application layers. In order to cope with network interruptions, the bundle

layer utilizes persistent storage, a hop-by-hop transfer of delivery responsibility (i.e. custody transfer), and an optional end-to-end acknowledgment. Both DTN and the Internet Protocol (IP) were designed with network interoperability in mind. However, in contrast to IP, DTN does not require nodes to: use a common network layer host identifier (i.e. the IP address), adhere to a common packet format, or assume a connected topology graph in order to support routing [13]. Also, even though both the bundle and the IP layers provide a store-and-forward forwarding service, DTN employs persistent storage, respecting the importance of storing vs. forwarding data, when connectivity cannot be taken for granted.

Traditional networking protocols make some fundamental assumptions that severely hinder their operation under certain circumstances [1][36][47]. These assumptions have become the motivation for the design of DTN; the ones that are more closely related to our work can be summarized as follows:

- There exists an end-to-end path between source and destination for the entire communication session.
- Errors may be effectively repaired by retransmitting data based on timely and stable feedback from data receivers.
- There is a relatively small end-to-end data loss.
- The participating nodes support the TCP/IP protocols.
- Switching fixed-size packets is an appropriate routing method for achieving interoperability and performance.
- Selection of a single route between sender and receiver yields acceptable communication performance.

The DTN architecture was set to relax the above assumptions employing a number of design principles that have been identified as follows [1][36]:

- Use potentially long messages of variable-length (as opposed to streams or small-sized packets) in order to enhance performance by allowing the network to make good transmission scheduling and/or path selection decisions.
- Employ a naming syntax that is general enough to support a wide variety of naming patterns in order to facilitate interoperability.
- Employ in-network storage enabling store-and-forward operation over multiple routes and possibly very long time intervals. This way, operation can be supported even in cases where end-to-end connectivity never exists, and end-to-end reliability is not required.
- Include features such as coarse-grained classes of services, delivery options, and data lifetime. These features allow the applications to express their requirements so that the network can better serve their needs.

Interoperability of heterogeneous networks is supported by a flexible addressing method based on Uniform Resource Identifiers (URIs) [48]. The use of URIs enables encapsulation of different naming patterns under the same general naming syntax. In case space communications are the sole purpose of a DTN, the naming scheme can be simplified in order to achieve higher efficiency. The motivation behind

using a URI-based naming scheme as well as possible naming simplifications that may be applied in specialized networks are discussed by Clare et al. in [49].

2.1.2. Delay-Disruption Tolerant Architecture

DTN applications exchange messages of arbitrary length, called Application Date Units (ADUs), whose reception order relative to their creation time may not be preserved by the network. An ADU corresponds to the minimum amount of information that is meaningful for the application layer. The bundle layer receives ADUs and creates, for each ADU, one or more Protocol Data Units (PDUs), which are called "bundles", and are ultimately exchanged by DTN nodes. The format of a bundle is specified by the Bundle Protocol (BP) [15].

A bundle may be split into multiple fragments, which are bundles in their own right, and may be further fragmented into, yet more, bundles. Bundle fragments may be reassembled anywhere in the network forming either longer fragments, or the complete original ADU. The use of arbitrarily long bundles in a store-and-forward overlay and the requirement for seamless protocol recovery after system restarts create the need for sufficient DTN node persistent storage so that bundles are retained until the next available communication opportunity (i.e. contact). Therefore, smooth protocol operation assumes adequate storage availability, well-distributed over the network.

Nodes and Endpoints

An entity capable of sending and receiving bundles, implementing the bundle layer, is referred to as a DTN node. A node may belong to one or more groups called "DTN endpoints". Endpoints are identified by Endpoint Identifiers (EIDs). The source and destination of a bundle are specified by the EIDs of the corresponding DTN endpoints, related to the original sender and the final destination respectively. Additionally, a bundle contains a "report-to" EID, which may include one or more DTN nodes as the recipients of diagnostic output, in case such service is requested.

A bundle is considered as being delivered if it has been received by a minimum subset of the nodes belonging to the DTN endpoint it is destined to. This minimum subset of nodes in the DTN terminology is called a Minimum Reception Group (MRG). Depending on the endpoint group membership and the semantics of MRG, DTN can support different types of data transmission such as:

- Unicast (a single node in the endpoint, which receives all data).
- Anycast (multiple nodes in the endpoint, any of which can receive the data).
- Multicast and broadcast (multiple nodes in the endpoint, all of which receive the data).

Through the EID the bundle layer determines the MRG of a certain DTN endpoint. It is possible that a certain DTN node may belong to the MRG of multiple endpoints. However, for each node there must be at least one endpoint that solely includes this node so that the node is uniquely identified by the corresponding EID.

The native support of multicasting in DTNs naming and addressing scheme is especially important for our proposed space-data dissemination paradigm. Interested end-users may subscribe to receiving certain types of scientific data gathered in space by joining the appropriate EID (i.e. each EID corresponds to a certain data type). Data are then transferred to the subscribed end-users through multicasting. Multicasting in DTNs has received significant attention from the research community, leading to the development of a wide range of multicast strategies, a taxonomy of which can be found in [50]. Researchers in [51] propose various DTN multicast models each with a different policy with respect to the number of copies of the same message allowed in the network. Srinivasan et al. explore reliability issues with DTN multicasting and propose their own Reliable Anonymous Multicasting in DTNs [52]. DTN multicasting has also been studied from a social networking perspective, where data forwarding decisions are based on the social network analysis concepts of centrality and communities [53][54].

Reliability Mechanisms

The DTN architecture defines eight options for delivering bundles, any combination of which can be requested by an application sending an ADU. Four of these options are designed for ordinary use and the rest are designed for diagnostic purposes. The diagnostic bundle delivery options are out of the scope of our work so we only include descriptions of the four delivery options for ordinary use. These are the following:

- "Custody Transfer Requested": Requests that reliability for the sent bundles will be enhanced using the custody transfer mechanism. The bundle layer will transmit bundles using reliable transport protocols (if such protocols are available), and the responsibility for delivery reliability will be transferred to one or more "custodians" on the network path.
- "Source Node Custody Acceptance Required": Provides a means for the application to demand that custody for the sent bundle be accepted by the source node. Enabling this option implies that the bundle in question must be permanently stored by the bundle layer at the originating node; otherwise the transmission request will fail. Therefore, there may be storage space contention among local and peer applications, especially evident in opportunistic DTNs.
- "Report When Bundle Delivered": Requests a bundle delivery report to be sent by the recipient node, when the ADU in question is successfully delivered.
- "Report When Bundle Acknowledged by Application": Requests a bundle delivery report to be sent by the receiving application, when the ADU in question is successfully delivered.

Of these four delivery options for ordinary use the custody transfer mechanism is the main reliability method employed in our proposed DTN-based solutions. The custody transfer mechanism allows the source to hand over retransmission responsibility for a bundle to another node and release resources related to the retransmission process. It is not mandatory for all nodes in the network to accept bundle custody and, consequently, custody transfer may not be a strict hop-by-hop mechanism. Accepting custody for a bundle requires that the receiving node has sufficient non-volatile storage so that the bundle can be retained until

custody for it is passed-on, or until it is successfully delivered to its destination. However, a node may refuse custody, despite having sufficient storage availability, in cases other resources, such as available bandwidth and power, are scarce.

Custody transfer allows tracking of the bundle delivery progress, but also provides a coarse mechanism for enhancing the reliability of bundle delivery, additionally to the reliability provided by the underlying transport layer. In cases where custody transfer is requested, the custodian node may initiate retransmission of transmitted bundles employing a retransmission timer at the bundle layer.

The delivery options briefly described in the previous paragraphs are facilitated by a number of signaling bundles, automatically produced by the network in response to certain events. These administrative records are categorized as Bundle Status Reports (BSRs) and Custody Signals. The available BSRs are: Bundle Reception, Custody Acceptance, Bundle Forwarded, Bundle Deletion, Bundle Delivery, Acknowledged by Application. A Boolean Custody Signal is also available to indicate whether custody has been successfully transferred or not. The difference between BSRs and the Custody Signal is that BSRs are sent to the designated "Report-to" node (most likely the source of the bundle), whereas Custody Signals are sent to the current custodian of the bundle. In that sense, the source node can be still notified about the delivery progress of a certain bundle even if custody has been delegated to some other node.

Reliability of finer grain than the custody mechanism may be additionally provided on a hop-by-hop basis by the underlying transport protocols and their related mechanisms. The transport solution originally coupled with DTN for deep-space links with long delays is the Licklider Transmission Protocol (LTP) [55], a point-to-point protocol providing different reliability levels according to the importance of transported data (i.e. important file header vs. less important individual payload values). Researchers have proposed a large number of transport protocols geared towards DTN, including the DS-TP [56] and DTTP [57] protocols. A comprehensive survey of available transport protocols for deep-space communications by Sarkar et al. can be found in [58].

When propagation delays are shorter, the hop-by-hop transport may also utilize the established Internet transport protocol TCP. Caini et al. study the performance of various TCP retransmission algorithms in a space DTN deployment of moderate delay and channel disruptions in [59]. A similar study, exploring the use of TCP as the DTN transport for links of longer propagation delay (i.e. cislunar communication), is presented by Wang et al. in [60]. This work also includes a comparative evaluation of the performance of TCP vs. LTP as the underlying DTN transports.

Bundle Fields and Identification

Each bundle consists of a sequence of at least two blocks. The first block is always a "primary block" and corresponds to the main header for the bundle. Following the primary block may be at most one payload block and/or one or more extension blocks, supporting extensions of the bundle protocol. Although our developed simulation model does not include all the bundle fields one-by-one, it does implement the

most important features of the BP through a subset of the fields and some complementary functionality. The main primary block fields that are relevant to our work are:

- Creation Timestamp: The bundle creation time combined with a sequence number, such that for a certain source node no two bundles share the same creation timestamp.
- Lifespan: The time when the bundle expires and is not useful anymore. When the Lifespan of a bundle is reached the network can safely discard the bundle.
- Source EID: The EID of the original sender.
- Destination EID: The EID of the final destination.
- Report-To EID: The EID of the node that will receive the BSRs for the bundle. May or may not be the same as the source node.
- Custodian EID: The EID of the bundle custodian, if available.
- Fragment Offset (optional): For bundle fragments, this field indicates the offset from the beginning of the original ADU that the first payload byte of this fragment corresponds to.
- Total Application Data Unit Length (optional): For bundle fragments, this field indicates the total length of the original ADU that this bundle belongs to.

Bundles belonging to the same original ADU share the same creation timestamp, which is unique to the source node EID. A bundle is uniquely identified by the combination of the source EID, the creation timestamp and the data offset/length information. Bundle identification information is used during custody transfer as well as for reassembling bundle fragments.

Administrative records such as BSRs and Custody signals are sent as part of a payload block. A flag within the primary block signifies whether the payload block must be interpreted as an administrative record by the receiving nodes.

Bundle Routing and Fragmentation

Routing in the DTN architecture is conducted on the basis of “contacts” or “contact opportunities”. Contacts are periods of time when two DTN nodes are connected via a link of constant capacity and delay. The product of the capacity and the contact duration represents the “volume” of the contact (i.e. the data amount that can be transmitted during the contact). Depending on their type, the contacts may be characterized as persistent, on-demand, intermittent-scheduled, intermittent-opportunistic, and intermittent-predicted. Out of the five contact types defined in the DTN architecture, our proposed solutions we will be mostly utilizing the following three:

- Persistent: Contacts that are always available.
- Intermittent-scheduled: Previously-known contacts of a limited duration.
- Intermittent-opportunistic: Previously-unknown contacts of a limited duration.

Routing schemes may involve multiple types of contacts ranging from persistent to fully opportunistic. A categorization and evaluation of routing algorithms of various contact types is provided by Jain et al. in [61].

Depending on the contact pattern, bundles may be fragmented by the bundle layer in order to enhance efficiency. Fragmentation of a bundle produces two bundles, containing the leading and the trailing bytes of the original bundle respectively, such that the aggregate size equals the original bundle length. Fragments may be further fragmented so that the original bundle is replaced by more than two fragments. It is noted that fragments resulted from different fragmentation events in different network locations may contain overlapping parts of the original bundle.

Fragmentation may be either proactive or reactive. In proactive fragmentation, a DTN node splits the application data contained in a bundle and transmits each of the resulting data blocks as an independent bundle. The bundle recipients are then responsible for reassembling the fragments into larger bundles, eventually reconstructing the entire ADU. In reactive fragmentation two adjacent DTN nodes may jointly fragment a bundle if it is only partially transferred as a result of connection interruption. The receiving bundle layer will re-package the received portion of the bundle and forward it to the next hop as a new fragment. The transmitting bundle layer will be notified for the fragmentation and resume forwarding of the remaining part of the original bundle as another fragment, possibly via a different route.

Selecting a message fragmentation strategy appropriate for the prevailing network settings may significantly improve network performance. Dias et al. [62] present a performance assessment of fragmentation mechanisms in vehicular DTNs attempting to better exploit limited contact opportunities. In [63] the authors present a similar study in a topology containing both vehicular as well as human nodes.

2.1.3. Convergence Layers

Due to the wide variety of networks aspired to be supported by DTN, it soon became clear that a middle layer between the bundle and the transport layers would be a valuable architectural addition, in order to reconcile disparate features and functionality of the underlying protocols. This additional layer, the *convergence layer*, adapts or extends an underlying protocol or protocol family, so that a consistent interface is presented to the bundle layer above [1]. Depending on the type of the underlying protocols and the service required by the bundle layer, a convergence layer may be quite simple (as in the case of TCP/IP) or considerably complex (as in the case where a reliable transport protocol is not available and has to be implemented at the convergence layer). In our proposed solutions we mainly deploy DTN over Internet protocols and, therefore, we will briefly describe the TCP and UDP convergence layers.

The DTN TCP Convergence Layer (TCPCL) is a connection oriented protocol that resides between the Bundle Protocol and the TCP layers in the protocol stack, providing a set of features that optimize DTN operation over TCP. The development of TCPCL is work in progress described in the Internet Draft by Demmer et al. in [64]. A TCPCL connection is established with the initiation of a corresponding TCP connection (i.e. there is a one-to-one correspondence between TCP and TCPCL connections). Each bundle may be transmitted over the TCP connection in a single or multiple TCPCL segments. In the case of

multiple segments, the first and last segments of a certain bundle are tagged, in order to signify the start and end of the bundle respectively. Optionally, the receiving TCPCL entity may acknowledge received segments as they arrive, allowing for reactive fragmentation in case of connection interruption. Furthermore, the receiver may at any point issue a negative acknowledgment (a REFUSE_BUNDLE signal), which notifies the sender to stop transmission of the current bundle. This feature allows for cancelling the transmission of a bundle that may have already been received, thus, saving link capacity. In one of our proposed designs we utilize the negative acknowledgment to reject a bundle in case custody transfer is required and the receiving node does not have sufficient available storage. In such a case, the receiver examines the bundle length in the primary bundle block and may decide to cancel the remaining transmission if the bundle cannot be accommodated for storage.

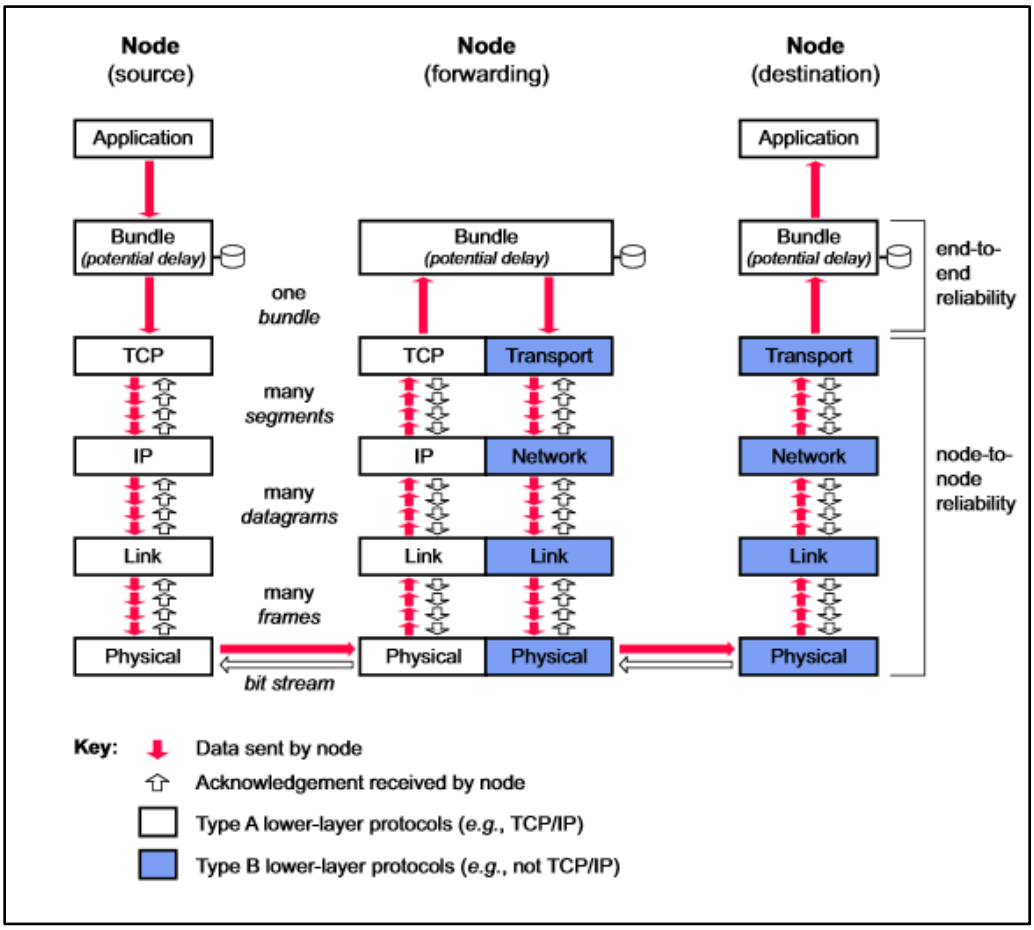

Figure 2-1: An example DTN protocol stack including the TCP/IP layers and the reliability signals. Source [67].

Additionally to the TCPCL, DTN includes provision for datagram, connectionless-oriented protocols (i.e. UDP [4] and DCCP [65]), as described in the Internet Draft by Kruse et al. [66]. Even though the authors of the document discourage the use of a datagram convergence layer, there may be scenarios where employing TCP or LTP introduces unreasonably high overhead and, thus, a simpler protocol may be more

appropriate. Each bundle in UDPCL is encapsulated in a single UDP packet and the packet size is set to the total bundle size. If the bundle size, including the bundle header, is larger than the maximum packet size for UDP, bundle transmission is aborted. The datagram size (and consequently the bundle size) must be carefully selected so that fragmentation at lower layers is limited. Excessive fragmentation may lead to severe efficiency degradations both due to increasing the loss probability for a bundle, as well as due to inducing additional computational effort for IP reassembly. The entire protocol stack when deploying the BP over TCP/IP protocol is depicted on the diagram of Figure 2-1.

2.2. Energy-Efficient Networking

Mobile devices, capable of connecting to the Internet through a wireless 802.11 LAN, have become commonplace, even with non-technologically-savvy users. The sophistication of these devices, along with the need for longer operating hours, creates an ever-increasing energy demand that cannot be matched by recent advances in battery technology [68]. The main sources of energy expenditure have been identified in the various subsystems of a mobile device [69], namely: CPU, memory, display, audio and wireless networking. For applications that heavily rely on the networking subsystem, the related functions may account for a substantial portion of the total energy consumption of the device.

In [17], Acquaviva et al. show that for networking-intensive applications the power requirements of the network interface may account for as much as 60% of the total power necessary for the mobile device operation. Consequently, improving the energy-efficiency of the networking subsystem has drawn significant attention from the research community, and has led to the implementation of energy-saving features in a number of wireless networking applications. The research literature on energy-efficient networking for mobile devices includes two broad categories of solutions that focus on the device hardware and the communication software (i.e. networking protocols) respectively. In our work, we study energy-efficiency of the networking subsystem from a communication software perspective.

The importance of energy-efficiency for wireless networking protocols has been readily identified by the wireless networking research community. In [18] Jones et al. provide a comprehensive survey of energy-efficient protocols for wireless networks and summarize the design principles for achieving energy-efficiency. The available solutions are studied by the authors on a per-network layer basis across the entire protocol stack, and the corresponding design principles are categorized with respect to the network layer they are applicable to (i.e. physical, data link/MAC, network, transport, OS/middleware and application).

At the lower network layers, such as the physical and data link/MAC, substantial effort in the energy-efficiency research field involves the ubiquitous WLAN standard 802.11[5]. The 802.11 protocol provides an energy-saving mechanism that buffers incoming data at the access point, allowing the mobile devices to temporarily switch their wireless interfaces to the sleep state in the meantime. The basic operation of 802.11 along with the provided energy conservation features, termed Power-Save Mode (PSM), will be further described in upcoming sections. Due to the probabilistic nature of the incoming data pattern and the relatively small buffer space at the access point, the energy conservation potential of the inherent 802.11

energy-saving mechanism is limited. This observation has led many researchers into examining alternative methods based on the same core principle [70].

In the following sections we provide a high-level description of the IEEE 802.11 standard including the PSM mechanism that a large number of energy-conserving solutions are based upon. The description of the standard is followed by a survey of energy-saving network protocols and, finally, by a short review of the available bandwidth estimation techniques. Within the context of our work we mostly focus on bandwidth estimation related to WLANs, such as the 802.11.

2.2.1. The IEEE 802.11 Standard

The IEEE 802.11 [5] lies in the center of almost every network protocol solution seeking to improve energy efficiency for mobile devices. Thus, it is deemed appropriate to provide a brief description of the basic protocol functionality along with the related energy-saving features. The 802.11 corresponds to a large family of protocols described in a specification document of over 1200 pages and, therefore, a comprehensive description of the entire content is beyond the scope of this dissertation. Instead, our intent is to first set the general context and then focus on specific parts that are mostly related to our work and, also, more meaningful from a practical perspective (e.g. more widely implemented by vendors).

802.11 comprises a set of specifications for the physical (PHY) and Medium Access Control (MAC) layers that facilitate communication in a Wireless Local Area Network (WLAN). There is a large number of physical standards available under the general 802.11 umbrella, denoted by a small letter such as a, b, g, n, etc. The most widely adopted incarnations of the physical layer specifications (i.e. b, g, n) use Direct-Sequence Spread Spectrum (DSSS) [71] or Orthogonal Frequency-Division Multiplexing (OFDM) [72] and provide maximum nominal bandwidth of 600 Mbps. However, the vast majority of home 802.11 access points nowadays offer a nominal bandwidth of 54 Mbps. Most physical layer standards operate around either the 2.4 or the 5 GHz frequencies and provide a selection of 14 separate channels that allow network administrators to limit interference among multiple overlapping 802.11 networks. In the following paragraphs the focus is set on the MAC layer of 802.11, which is where the energy-saving features are implemented.

The 802.11 specification describes two basic building-blocks fundamental to its operation: A Basic Service Set (BSS), consisting of a single Access Point (AP) and the associated mobile terminals (i.e. infrastructure mode), and an Independent Basic Service Set (IBSS), consisting of mobile terminals forming an ad-hoc network without the presence of an AP (ad-hoc mode). The ad-hoc operation of 802.11 is outside the scope of the present work, so we will focus on the infrastructure mode where mobile terminals are associated with a controlling AP. The infrastructure mode of operation supports the integration of multiple BSS components (multiple areas, each covered by a single AP) into a single Extended Service Set (ESS), within witch the mobile terminals can roam, switching their associated AP transparently to the user.

The 802.11 specifications support the following three main MAC techniques:

- Distributed Coordination Function (DCF): The default MAC technique used in both the infrastructure and the ad-hoc modes of operation.
- Point Coordination Function (PCF): A centrally coordinated technique which uses the AP as a coordinator managing channel access. PCF depends on DCF and is used in the infrastructure mode of operation only.
- Hybrid Coordination Function (HCF): A technique that is, similarly to PCF, centrally coordinated (i.e. used in infrastructure mode only) and provides additional support for traffic prioritization with respect to certain QoS requirements.

The Wi-Fi Alliance [73], which certifies 802.11 compatible products, does not include the PCF and HCF MAC techniques into their requirements. As a result, most home 802.11 products (i.e. Access Points and WNICs) only implement DCF and, for this reason, the next few paragraphs will be devoted to the most widely used DCF function. Please note that applicability of the proposed solutions has been central in our work. Therefore, it was essential to explore the practical issues of a possible real deployment, including the implications of deploying the energy-efficient DTN overlay over 802.11 and the incorporation of an effective bandwidth estimation mechanism into the 802.11 MAC.

The Distributed Coordination Function

At the heart of the 802.11 DCF is a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) medium access method with exponential backoff [5]. In order for a 802.11 station to commence transmission, it must first listen to the wireless medium for a time interval named DCF Interframe Space (DIFS). If the medium is found to be idle for the specified duration then the station may start transmitting frames, otherwise the station waits for an idle DIFS and then defers transmission for an additional randomly calculated duration selected within a certain Contention Window (CW). For every unsuccessful attempt to transmit a frame the station doubles the CW up to a specified maximum value. The random backoff mechanism limits collisions in case multiple stations try to transmit frames simultaneously. The CW is reset to its initial value when a frame has been successfully transmitted.

The diagram of Figure 2-2 shows an example frame exchange and backoff activity of five mobile stations contending for medium access. It can be seen that backoff time does not elapse during DIFS intervals nor during busy medium intervals. For instance, Station B selects a backoff time, which is paused during the various DIFS intervals as well as during transmission of the frames by C, D, and E. The sequence of the created frames in the example is A, D, B, C and E, whereas the sequence of the actual transmission due DCF backoff and deferral is A, C, D, E, B. After each successful transmission it can be seen how the CW is reset to the minimum value.

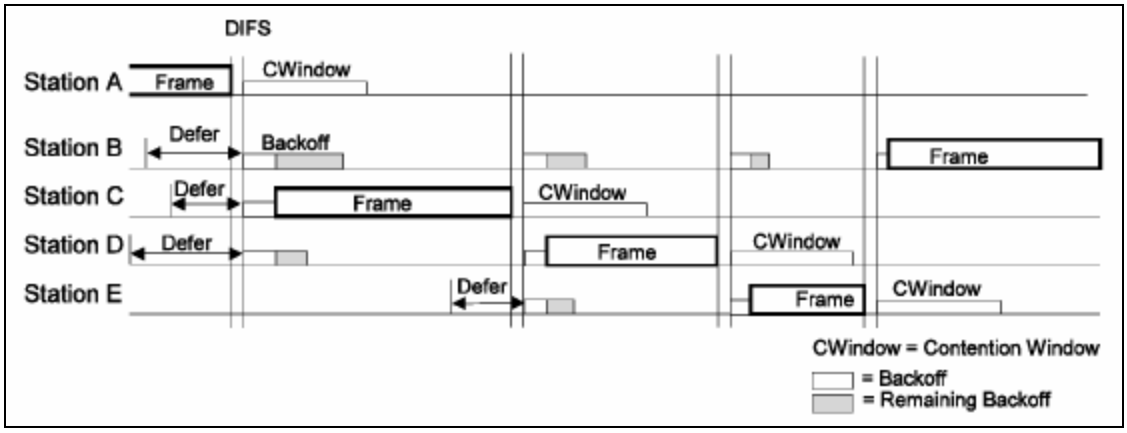

Figure 2-2: An example DCF CSMA/CA medium access. Source [5].

Additionally to physical carrier sense DCF also employs a virtual carrier sense mechanism, which relies on distribution of medium reservation information included in the transmitted frames. One means of distributing this information is through the Request-To-Send/Clear-To-Send (RTS/CTS) frames exchanged respectively by the transmitter and receiver, prior to the actual data frames. The RTS/CTS frames include a duration field that contains the period of time the medium will be reserved for the transmission. The RTS/CTS mechanism also helps limit to a great extent the well-known *hidden terminal* problem [74] that manifests itself in wireless networks. For instance, a wireless node outside the range of the transmitting node (the transmitting node issues the RTS), but within the range of the receiving node (the receiving node issues the CTS), will be informed about the medium reservation from the CTS, even though it will not be able to sense the originating node's transmission. Duration information is also included in individually transmitted frames when the RTS/CTS sequence is not exchanged. The virtual carrier sensing procedures of DCF are referred to as the Network Allocation Vector (NAV). Each station maintains a NAV, which is updated with the information received from medium reservation messages and corresponds to the predictions for future traffic. A station will wait for the duration specified by the NAV to expire before attempting to actually sense the medium.

Reliability in DCF is enhanced by a positive acknowledgment mechanism, which dictates that the receiving station must issue an ACK frame for the sender upon successful reception of a data frame. The ACK frame is issued by the destination a Short Interframe Space (SIFS) after the reception of the data frame. SIFS is shorter than DIFS and is used so that control frames gain medium access with a higher priority than data frames. SIFS is also used between RTS/CTS and data frames. The 802.11 MAC specification contains a number of additional Interframe Spaces outside the scope of our work.

Having a clear picture of the IFSs, the backoff mechanism and the various signaling frames (e.g. RTS/CTS, ACK) is crucial for correctly assessing bandwidth availability. For example, an idle medium does not necessarily imply excess network capacity; rather, the participating nodes may be backing off as part of the collision avoidance mechanism or allowing the necessary IFSs determined by the protocol.

Obviously, these seemingly idle intervals must not be considered as idle intervals for the WLAN in its entirety.

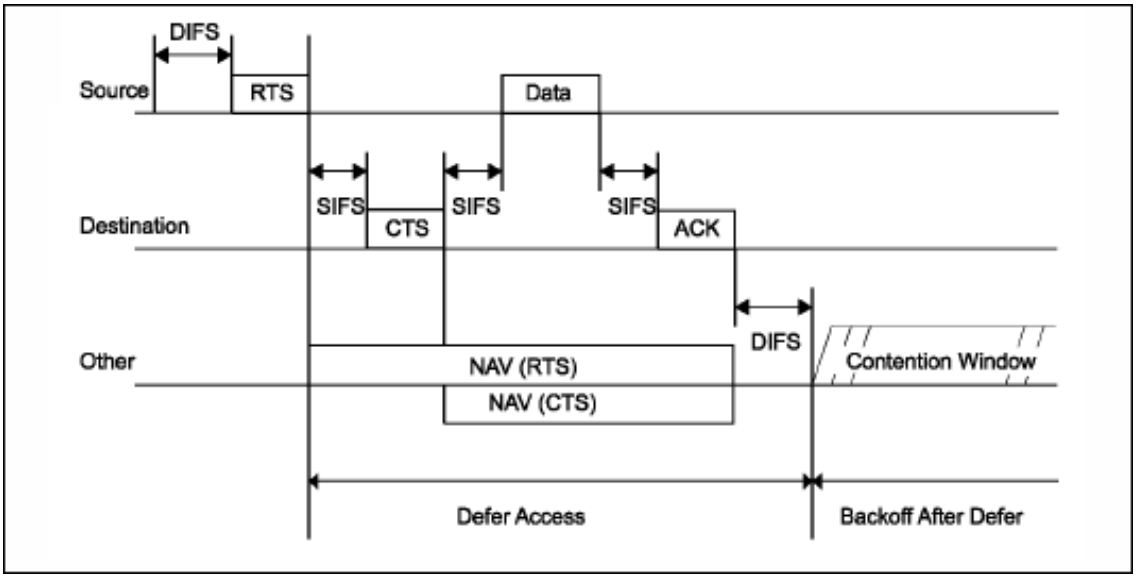

Figure 2-3: An example frame exchange sequence including the RTS/CTS and ACK frames. Source [5].

The diagram in Figure 2-3 gives an example of a frame exchange sequence including the RTS/CTS and the ACK frames. The source defers until a DIFS idle time is detected and then issues an RTS frame. Upon reception of the RTS frame the destination defers for SIFS and responds with a CTS frame. The Data and ACK frames are also transmitted using SIFS intervals. The reasoning behind this design is that the entire sequence starting with the RTS and concluding with the ACK frame must be completed before any other transmission commences. Other mobile stations defer according to the NAV values updated from the RTS/CTS (as depicted on the diagram), wait for an additional DIFS, and then exercise the backoff mechanism.

Power-Save Mode in Infrastructure 802.11 Networks

Key to the operation of an infrastructure 802.11 network (such as those commonly used in home settings) is the AP that the mobile terminals need to associate with in order to gain access to the wired network. The AP advertises the WLAN to interested clients by transmitting a Beacon frame every some predefined time interval, the Beacon interval, whose default duration on most home APs is 100ms. Besides advertising the WLAN, the Beacon frame is also used to coordinate the operation of already associated stations with respect to a number of different functions, including power management.

A station participating in an infrastructure 802.11 network can be in any of the *Active* or *Power-Save* power management modes. Accordingly, with respect to its network interface a station can be in the *Awake* and *Sleep* (or *Doze*) power state. When the mobile terminal wishes to switch power management mode it

must inform the AP using certain fields of the transmitted frame header. While in the Power-Save Mode (PSM) the station can switch its network interface to the Sleep power state in order to save energy. The AP maintains a list of stations in PSM and refrains from transmitting frames to these stations, buffering incoming data for when the stations switch back to Active. Each Beacon transmitted by the AP contains a Traffic Indication Map (TIM), listing the stations in PSM for which buffered data await delivery. A station in PSM may periodically wake-up, examine the TIM of the next Beacon frame and, in case buffered data are available, request delivery. Otherwise, the station may decide to switch back to PSM.

The energy conservation potential of the 802.11 PSM is limited by the relatively small buffer space at the BS and the lack of visibility at higher network layers, leading many researchers into examining alternative methods based on the same core principle. Some notable examples of such methods are described in the following sections.

2.2.2. Energy-Saving Protocols

In [75] Chandra and Vahdat highlight the limitations of the 802.11 PSM and propose an application-specific traffic shaping mechanism for multimedia streaming that can be implemented either on the server (source of the stream) or at the access point, in the same LAN as the receiving client. The authors describe how their mechanism achieves energy savings by allowing the client to sleep for periods of time longer than those achieved by the 802.11 PSM. The system architecture consists of a client side proxy and a server-side or a local proxy. The server-side or local proxy informs the client proxy of the next data arrival, based on a predetermined delay value. The client-side proxy is then responsible for transitioning the client to a low power sleep state.

In [76], Adams and Muntean propose an Adaptive-Buffer Power Save Mechanism (AB-PSM), again for multimedia streaming, that buffers data at higher network layers, essentially hiding them from the BS and allowing for longer idle intervals. The proposed AB-PSM is targeted at mobile multimedia streaming and improves the inherent 802.11 PSM without making any modifications to it. AB-PSM introduces an additional buffer at the Application Layer of the server, complementing the Access Point Buffer specified by the basic PSM. Depending on the application requirements for data latency, the additional buffering introduced by AB-PSM can be set to the desired multiple of beacon intervals. Energy-saving strategies when streaming from a single server to multiple clients are explored by Acquaviva et al. in [77]. The authors explore the energy expenditure impact of a close-loop strategy (the client triggers the transmission of data) and two open-loop strategies (the server has full control of the data transmission).

Zhu and Cao in [78] expand on the proxy idea by introducing a scheduler service at the base station and a proxy at the mobile terminal. The scheduler incorporates an algorithm, called Priority-based Bulk Scheduling (PBS), that employs data buffering on the client side in order to save power, respecting the QoS requirements set by the application. PBS considers multiple mobile terminals with active incoming data flows and schedules pre-fetching of the buffered data so as to avoid simultaneous client requests. The proposed solution is geared towards multimedia streaming, and, more specifically, audio-on-demand, even though the authors also use web traces for their simulation experiments.

In [79] Anastasi et al. propose a power-aware multimedia streaming protocol for mobile users. The proxy idea is again implemented at the base station introducing the proposed Real-Time Power-Saving protocol (RT_PS) for the communication between the base station and the mobile node. The RT_PS protocol includes idle interval information so that the client can put the WNIC into sleep mode when applicable. In order to mitigate possible congestion on the wired part of the network, the authors suggest fetching data from the server using the TCP-Friendly Rate Control protocol.

The buffering technique that is common to most of the described power-saving solutions for streaming protocols tends to create bursts of transmitted data at high peak rates, thus, rendering the WLAN prone to congestion. The authors in [80] present a theoretical and empirical study of the impact of the transmission peak rate and the burst length on observed network performance characteristics, with or without the use of a proxy service. Based on their results, they propose an adaptive, energy-saving streaming mechanism that accordingly adjusts the burst length so that network congestion is avoided.

In order to prolong sleep intervals on the mobile client side, Balasubramanian et al. [81] introduce a protocol that delays transmissions (for delay-tolerant applications, such as email) and pre-fetches data (for applications like web search). They also perform detailed measurements on the energy expenditure of mobile devices when using 3G, GSM and 802.11 as their network interface. The same core principle that allows for longer sleep intervals is exercised by the authors in [82] who apply power-saving strategies at the transport layer (i.e. TCP protocol). In this work, the researchers consider an indirect TCP model (such as the one originally presented in [83]) by introducing a base station proxy service, which splits the end-to-end TCP connection into two connections; one between the mobile host and the base station, and the other between the base station and the wired server. The former connection employs their novel proposed Power-Saving Transmission Protocol (PS-TP) and the latter employs regular TCP. The proxy service downloads all data requested by the mobile host, while the mobile host remains in the sleep state. When the requested data becomes available, they are all downloaded at once by the receiver.

By the same token, Batsiolas and Nikolaidis [84] propose a modified TCP version, called Selective-Idling TCP (SI-TCP), which utilizes the round trip time estimates in order to drive a mechanism that selectively idles the transceiver, thus improving energy-efficiency. SI-TCP allows regulation of TCP acknowledgments so that the sender may predict with some accuracy when ACKs are expected, and enter sleep mode in the meantime.

The energy efficiency of transport protocols has also been the focus of investigation by Mamatas and Tsaoussidis in [85]. The authors examine various flavors of TCP from two perspectives: in terms of the energy-saving potential of the protocol and in terms of the risk for the protocol to receive unfair service due to its energy-saving features. The above qualities are quantified with the help of two new metrics introduced by the authors, the Energy Potential and the Risk Index. The presented experiments monitor the state transitions of the WNIC and evaluate the energy-related characteristics of different transport strategies.

It is clear from the above paragraphs that multiple energy-conserving solutions, aiming at prolonging the sleep intervals of a WNIC in 802.11 WLANs, have been proposed in the literature. Most of the described solutions target at streaming applications and involve the installation of specialized components, such as proxies, schedulers and local services in order to operate. The deployment of the solutions found in the related literature usually involves implementing additional functionality at the mobile terminal, the base station and, possibly, the source of the data transfer. The network *per se* does not participate in the effort for energy-efficiency.

In our work, we use the inherent capabilities of DTN for shaping internetwork traffic in a manner that allows mobile receivers to suspend their WNICs with minimal cost on throughput. Using DTN, it is the network itself, as opposed to some application proxy, that realizes the necessary traffic shaping. A minimal overlay would involve deploying DTN on a mere three nodes (source, base station and mobile receiver), but in the general case the overlay may contain multiple intermediate nodes as well. The bundle protocol is enhanced with a rendezvous mechanism implemented between the base station and the mobile receiver. The rest of the DTN nodes in the overlay do not need to be aware of the additional functionality necessary to support the rendezvous mechanism.

Exploiting the custody feature of DTN, storage distribution over the network path can be automatically balanced, shifting data towards the destination when storage is available. Through the simulation experiments presented in a later chapter, it becomes apparent that traffic shaping in the internetworking overlay significantly smoothes the probabilistic nature of Internet traffic, allowing for enhancing the energy-efficiency of the wireless devices connected at the edges of the network. The storage provided by the bundle layer is the key feature that we exploit in our quest for enhancing energy-efficiency.

2.2.3. Bandwidth Estimation

Bandwidth availability information allows networking applications and protocols to adjust their behavior in response to the changing network conditions, significantly enhancing the efficiency of network-related operations. Information on available bandwidth can be of importance in a multitude of applications related to network resource management. Some applications that can benefit from available bandwidth knowledge are [86]:

- Setting up user-level links based on available bandwidth in peer-to-peer applications.
- Configuring routing tables based on available bandwidth in overlay networks.
- Monitoring Service-Level-Agreements (SLAs) metrics agreed upon between customer and provider.

The wide range of possible applications for such visibility into bandwidth availability (the main network resource) has drawn substantial attention from researchers to bandwidth estimation techniques for over two decades [68]. One of the first such efforts has been packet pair probing, proposed by S. Keshav in the early nineties [87].

Bandwidth estimation techniques usually belong to one of two broad categories: *active* and *passive*. Active bandwidth estimation techniques inject probing traffic into the network, take certain measurements related to the injected traffic, and use these measurements in order to calculate available bandwidth on a certain network path. Traditionally, active bandwidth estimation has been applied on broadband network connections, while, more recently, it has been applied on wireless 802.11 LANs as well ([88][89]). Active bandwidth estimation employs four major types of techniques as described in [86]: Variable Packet Size probing, Packet Pair/Train Dispersion, Self-Loading Periodic Streams, and Trains of Packet Pairs.

Passive bandwidth estimation relies on data passively collected from the network in order to calculate available bandwidth. In wireless networks, passive techniques take advantage of the broadcast nature of the communication in order to snoop on in-range transmissions and calculate idle periods of the medium. For single-hop, wireless networks such as the 802.11, many researchers suggest that passive methods are more pertinent than active ones ([90], [91], [92]). Generally, passive bandwidth estimation techniques are non-intrusive (i.e. do not burden the network with extra traffic), more responsive (i.e. no probing is required; channel utilization information is readily available), and more accurate than active techniques (i.e. active techniques rely on assumptions that do not hold for wireless 802.11 LANs [92]).

Sarr et al. in [90] present a method to evaluate the available bandwidth in ad hoc networks that uses both passive and active mechanisms. They use the carrier sense capability of wireless nodes combined with mechanisms like collision prediction in order to provide an evaluation mechanism that compromises accuracy and resource consumption. Their method focuses on ad hoc 802.11 settings, so it is not applicable to infrastructure WLANs. In a similar setting, Nielsen et al. present in [93] the results of a testbed-based comparison of an active versus a passive bandwidth estimation method for mobile ad-hoc networks, indicating the pros and cons of each approach.

In [91], Lee et al. adopt a strictly passive approach and propose the IdleGap method, which used the Network Allocation Vector (NAV) of 802.11 in order to provide bandwidth estimation for multimedia streaming services in WLANs. Their method involves adding an *idle module* in the MAC layer of every wireless node that needs access to bandwidth availability information. In [94], the authors propose a passive traffic load estimation for infrastructure WLANs, which is part of their proposed LAN traffic monitoring systems, Wireless Network Sentry (Wintry). The system relies on the deployment of multiple sentry nodes near the networks APs. Sentry nodes overhear in-range traffic and send the measurements to a central node, which, in turn, combines the received data and publishes bandwidth availability information through a management console.

Our solution to the bandwidth estimation problem in 802.11 LANs consists of a strictly passive (i.e. non-intrusive) approach and is deployed only on the AP of the WLAN. Due to our energy-efficiency concerns, deployment on mobile terminals was ruled out as it would require continuous reception of in-range traffic, forcing the WNIC to remain active and expend energy even during periods of inactivity. Additionally, the bandwidth availability estimate would be utilized at the AP in order to adjust the amount of buffered data and schedule transmission. In case certain mobile nodes also require access to the

bandwidth availability information the AP may communicate the corresponding value to them via a dedicated protocol header during the rendezvous process.

2.3. Space-Data Dissemination

Dissemination of collected data constitutes a major challenge in space missions, drawing significant attention by both space agencies and research institutions. The wide variety of orbital patterns, the high volume of collected data and the large number of interested parties render the design of effective transmission and distribution schemes for space-data a chief factor in successful missions. Nowadays, the vast majority of Earth observation missions employ Low Earth Orbit (LEO) satellites due to the close proximity of LEO to the Earth surface. LEO satellites operate at approximately 200-2000 km and provide much finer observation capabilities than their Medium Earth Orbit (MEO) and Geosynchronous Equatorial Orbit (GEO) counterparts, orbiting at approximately 20,000 km and 35,786 km respectively. MEO and GEO satellites have been mainly used for navigation and telecommunication applications, respectively [95].

Because of the lower altitude, the links between LEO satellites and designated ground stations on Earth exhibit much lower propagation delay than the higher orbit links, while transmission power requirements are relatively low. On the downside, ground contact opportunities with LEO satellites are brief and infrequent, a liability of their small footprint, high speed and variable orbital pattern. In order to compensate for the limited connection opportunity, LEO satellites are often deployed in groups (constellations) and may communicate with multiple ground stations per orbit.

2.3.1. LEO Missions Overview

The majority of operational LEO satellites are related to scientific missions, mainly employed by major space agencies and consortia. NASA operates its Earth Observing System [96] consisting of a series of satellites including the Aqua, Aura, and ClouSat, which belong to the Afternoon Constellation (A-Train) [97], the Terra, Landsat 5 and 6, which belong to the Morning Constellation [98], as well as a number of individual satellites. These satellites observe the land surface, biosphere, solid Earth, atmosphere and oceans. In Europe, ESA operates the Sentinels missions [20] consisting of Sentinel-1, which has already been launched, and a series of other Sentinels (i.e. 2, 3, 4, 5), which will follow in the upcoming years. These satellites will be equipped with multi-spectral imaging instruments for land, ocean and atmospheric monitoring. A consortium of the Algerian, Nigerian, Turkish, British and Chinese governments operates the Disaster Monitoring Constellation [99] providing emergency Earth imaging for disaster relief. Finally, a notable example of a commercial LEO constellation is the RapidEye [100], operated by BlackBridge, which consists of 5 satellites providing geo-image and geo-information services to interested parties.

A number of telecommunication LEO satellite constellations have also been deployed with moderate, however, commercial success [101]. The main representatives of such constellations are Iridium, Globalstar and OrbComm, which aim to provide voice and data communication services, and support a diverse range

of applications including geodesy, navigation, remote sensing, personal and asset tracking, data monitoring and broadband networking applications.

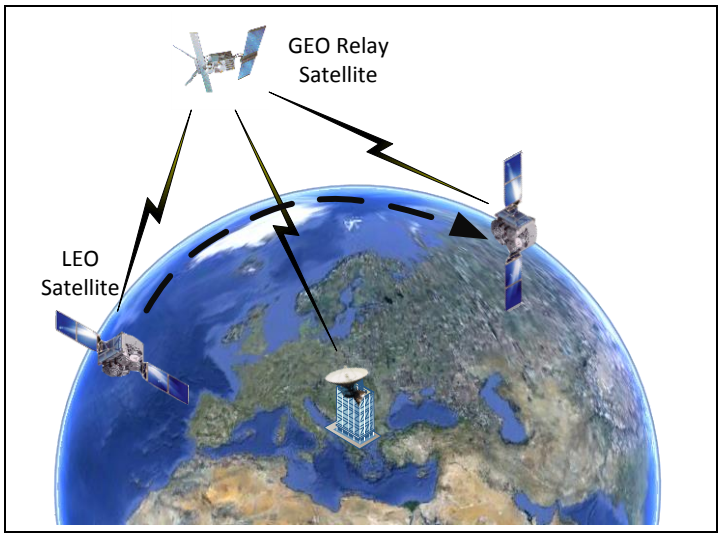

Figure 2-4: GEO relay satellite operation.

Typical LEO scientific missions rely on large, expensive satellites that include high-end communication subsystems, contacting stations on the ground over point-to-point, bidirectional links. Once the physical connection locks, the satellite downlinks the collected data in a reliable fashion. The downlink capacity of such typical LEO missions can be significantly increased by employing specialized GEO satellites as a means to relay data (e.g. NASA's Tracking and Data Relay System [19] and the future ESA's European Data Relay System [102]). The GEO relay satellite has continuous contact with the designated ground station and provides much larger contact windows with the LEO satellites as it can be seen in Figure 2-4.

In special cases, the main data transfer, which involves point-to-point, bidirectional links with designated ground stations, may be complemented by a broadcasting mechanism. In such missions, scientific data are broadcast as acquired so that they may be received by in-range ground terminals at any given orbit location. This mechanism enables interested parties to setup their own ground communication equipment and gain immediate access to scientific data collected in their vicinity. The collected data are also downlinked via the typical method when the satellite passes over the designated ground stations. Complementary broadcasting is implemented in NASA's Terra, Aqua and NPP satellites [19], employing the Direct Broadcast technology developed by NASA's Direct Readout Laboratory. Unlike complementary broadcasting technologies, our proposed solution employs broadcasting as the main means for transferring space-data to the ground, optionally involving occasional point-to-point connections.

Protocols commonly used in LEO missions are the Telemetry (TM)/Telecommand (TC) [103][104] or the Advanced Orbiting Systems (AOS) [105] Space Data Link Protocols at the data link layer and the CCSDS File Delivery Protocol (CFDP) [106] at the transport/application layer. Recently, DTN has also

been proposed as a solution for LEO satellite communication. In fact, the first real DTN deployment in space took place in the context of a test conducted on the Disaster Monitoring Constellation [99], the outcome of which is described in [107].

In contrast to the pattern of high-end, high-cost satellites described so far, the QB50 EU FP7 research project [22] focuses on the development of a low-cost, distributed LEO satellite network. The QB50 project has already deployed 50 Cubesats for multi-point, in-site measurements in the lower thermosphere and another 100-150 Cubesats will be ready for launch in the following few years. The Cubesats are deployed sequentially in a nearly circular orbit at 320 km altitude. Even though a single CubeSat is too small to carry sensors for significant scientific research, combining a large number of CubeSats with identical sensors into a network, can provide a substantial volume of scientific measurements [108]. An example of such network, the QB50 atmospheric research network, will consist of 40 double-unit CubeSats (10x10x20 cm), with one unit (the 'functional' unit) providing the satellite functions and the other unit (the 'science' unit) accommodating the necessary sensors for lower thermosphere and re-entry research. Figure 2-5 depicts the first two CubeSats deployed as part of the QB50 precursor mission by the Innovating Solutions In Space (ISIS) Corporation. Data downlink services for the QB50 mission will be provided by the Global Educational Network for Satellite Operations (GENSO) [109], a network of low-cost ground stations spread across the globe.

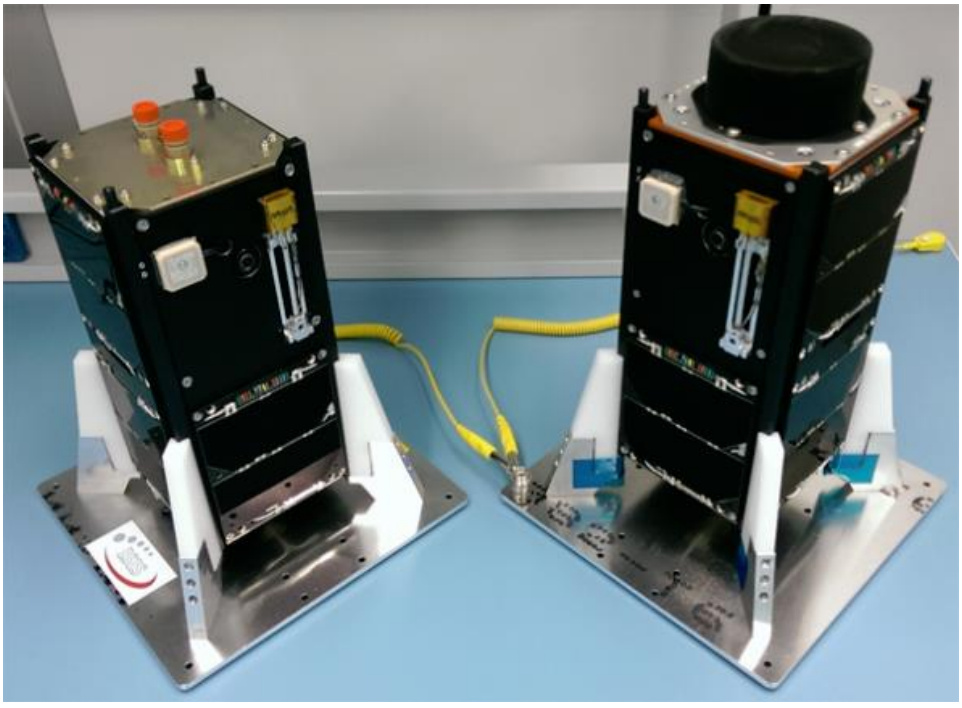

Figure 2-5: QB50p1 and QB50p2 of the QB50 precursor mission. Source: [110].

The special characteristics of such dense constellations of small satellites (i.e. short contacts of low bandwidth) have spurred research towards specialized MAC protocols for the downlink of data. One such

attempt is the ALOHAGP protocol deployed as a multiple access method for DTN presented by Bedon et al. in [111].

2.3.2. Reliable Satellite Broadcasting

Since broadcasting is the main means of transmitting space-data in our proposed solution it was necessary to explore ways to enhance reliability, even in cases where a return path does not exist. Reliable data broadcasting, possibly for satellite data, has been the focus of several networking protocols. File Delivery over Unidirectional Transport (FLUTE) [112] is a protocol for the unidirectional delivery of files over the Internet, tailored for multicast networks. FLUTE uses Asynchronous Layer Coding (ACL) [113], which, in turn, employs Forward Error Correction (FEC) techniques to achieve reliability. The sender generates encoding symbols based on the file that needs to be transferred and includes these symbols in the transmitted packets. The receivers need to wait for a sufficient number of packets to arrive in order for them to reliably reconstruct the object, alleviating the need for requesting lost packet retransmission. FLUTE is designed specifically for unidirectional transport and it does not include provisions for utilizing a return channel in case such a channel is available.

Similarly to FLUTE, the NACK-Oriented Reliable Multicast (NORM) [114] protocol provides reliable multicasting services, depending, however, on a return channel and receiver-generated negative acknowledgments for achieving reliability. NORM provides end-to-end reliable transport of bulk data or streams over IP multicast. It uses a selective, negative acknowledgment mechanism for providing transport reliability, and allows utilization of the provided service by different types of applications. In the same spirit, but with a focus on unidirectional satellite transmissions, the authors in [115] propose a Reliable Multicast over Unidirectional Satellite link (RMUS) protocol, which provides error and congestion control to multicast transmission. RMUS achieves data reliability by having receivers detect packet losses and report them to the sender, over a narrow return path (ground-to-satellite link). Again, this solution relies on the existence of a return path through which the sender is informed of possible data losses.

Finally, the authors in [56] propose the Deep-Space Transport Protocol (DS-TP), a point-to-point protocol that utilizes a pre-emptive retransmission strategy to accelerate recovery of packet losses. Extremely long propagation delays of deep-space links favor the use of redundant data transmission as a complementary mechanism to data delivery feedback. Even though DS-TP is not directly related to satellite broadcasting, it does share with our proposed solution the characteristic of redundantly transmitting data in order to achieve better data reliability.

2.3.3. Space-Data Dissemination Schemes

The efficient distribution and management of space-data, once it has been successfully received on the ground, constitutes a major challenge in modern space missions, mainly due to the vast quantities of acquired data as well as the large number and diversity of interested end-users. Data are typically stored on central repositories and end-user access is ensured via online portals on a client-server basis over FTP or

HTTP downloads. “ESA Earth Online” [116] and “NASA Earth Observations” [117] are the corresponding portals providing access to ESA’s and NASA’s Earth observation scientific data respectively.

In the recent years, researchers have been exploring more sophisticated space-data dissemination schemes. In the Space-Data Routers FP7 EU research project, space-data are delivered to end-users via a DTN overlay including Ground Stations and Space Research Centers [118]. The project aims at unifying space and earth communication infrastructures and delivering a set of tools and protocols for space-data exploitation. Space-Data Routers includes an application that thematically integrates various missions on the basis of scientific topics. The application allows end-users to filter space-data across different missions through an online web interface.

2.4. Opportunistic DTN Routing

Opportunistic DTN routing has long been a very active topic within the networking research community. Routing protocols traditionally targeting ad hoc networks, such as AODV [119] and DSR [120], fail to create routes in DTNs due the lack of end-to-end connectivity. When continuous end-to-end paths cannot be guaranteed to exist, a “store-carry-forward” approach must be adopted by the routing algorithm in order to improve data delivery chances. Notable examples of routing algorithms for opportunistic DTNs that are popular with the research community are Epidemic Routing [121], PRoPHET [122], and Spray and Wait [123]. Accurately predicting mobility patterns can be crucial for improving the performance of opportunistic routing algorithms. Chen et al. [124] develop a model that actively captures mobility information and assists the routing algorithm in making better forwarding decisions. The literature also contains attempts to integrate ad-hoc and DTN routing in hybrid schemes so that the best of both worlds may be combined as in [125] and [126].

In general, opportunistic DTN routing protocols inject multiple copies of each packet in the network and, with the help of heuristics, attempt to improve the data delivery probability while limiting the amount of expended resources. Resources such as storage, energy and bandwidth are likely to be scarce in opportunistic DTN settings. The resource consumption vs. network performance tradeoff has drawn significant attention within the research community. The authors in [127] focus on the energy-delay and storage-delay aspects of these tradeoffs. The authors in [128] study the effects of storage congestion in DTNs and propose a routing scheme that utilizes neighboring storage availability in order to increase routing efficiency.

In the recent years, researchers have made attempts to approach opportunistic DTN routing as a Game Theory problem. In [26] the nodes are randomly moving and occasionally coming within radio range of each other, when they have to decide whether to accept forwarding a peer’s bundle or not. The decision made at node contact is in turn considered under the assumptions of a strategic, an extensive, a Bayesian, or an extensive with imperfect information game. Along the same lines, but in a considerably more extensive work, authors in [129] provide an analytical study of a 2-hop routing problem in a DTN. The network is sparse and consists of sources, relays and destinations. The nodes are moving randomly coming into

contact with each other. At each contact of a source with a relaying node, the relaying node decides whether it will participate in the transfer or not. The game is mathematically analyzed and simulated as an evolutionary game, studying the evolution of the population according to the corresponding strategies that are followed.

In [27] Shevade et al. employ mechanisms inspired from Game Theory in order to achieve fair and efficient routing in DTNs. The mechanisms provide incentives that reward collaborative nodes while punishing nodes that behave in a selfish fashion. The core mechanism is based on Tit-For-Tat [130], while "generosity for bootstrap" and "vendetta mitigation through contrition" are employed in order to boost good faith between nodes in the network. Similarly, Buttyan et al. [131] propose a mechanism that discourages selfish node behavior in DTNs and develop a game-theoretic model in order to show that enhanced cooperation among nodes is, indeed, achieved. In another piece of related work, this time using a central credit authority, the authors in [132] present a protocol involving a probabilistic acceptance mechanism that is based on whether the corresponding node is cooperative or not. Node credits are kept at a central repository, while node reputation is built among neighbors.

2.5. Simulation Environments

During the development of computer network protocols, network simulators play a prominent role in the hands of the protocol designer. In the early stages of our work we identified a number of candidate simulation tools for developing our proposed solutions. The next section includes a brief description of the network simulation tools available to the research community, the subsequent section focuses on the ns-2 network simulation environment, and, finally, the last section surveys space-related simulation tools that are either available as commercial products or have been custom-developed by researchers to meet certain research needs.

2.5.1. Simulators Overview

Ns-2 [12] is a general-purpose network simulator, offering detailed implementations of protocols throughout the network stack. Specifically, ns-2 allows exploration of protocol design issues that extend all the way to the MAC and physical layers of wireless 802.11 LANs, and provides detailed energy consumption models, especially useful for the evaluation of our proposed energy-efficient DTN overlay. Among others, ns-2 has been extensively used in the development of current Internet protocols and so it includes detailed models pertaining to these protocols. Ns-2 provides visualization of the topology and the network activity via an external tool, the Network Animator (Nam).

Ns-2 is in the process of being superseded by ns-3 [133], which has been under development for several years prior to the writing of this dissertation. Ns-3 has been designed from scratch in order to provide a simpler architecture and a more practical toolset than ns-2. Implementing functionality already present in ns-2 has been a slow process, as older code cannot be directly ported to the new platform and, so, ns-3 was not mature enough for adoption at the time the work towards this PhD degree commenced.

Ns-2 does not provide built-in capability for DTN-related protocols, although it does provide support for MANETs. Lakkakorpi et al. have developed DTN models which can be incorporated in both ns-2 [134] and its successor ns-3 [135]. These models allow the experimentation with TCP over the ad hoc routing protocol AODV [119], as well as a number of specialized opportunistic DTN routing protocols such as Epidemic Routing [121], PRoPHET [122], and Spray and Wait [123]. These extensions augment ns-2 with significant DTN functionality; however, they do not support employing TCP or UDP as a hop-by-hop transport method. Rather, the Bundle Protocol is employed directly over IP and reliability is provided by the DTN custody mechanism alone.

Contrary to ns-2, the ONE simulator, presented by Keränen et al. in [28], is a specialized network simulator focused on simulating DTN-related protocols in opportunistic network environments. ONE generates node movement using a number of different movement models (mobility patterns), handles the exchange of routing messages between nodes for a variety of DTN routing algorithms, and provides visualization tools for mobility and message passing in real-time. However, ONE provides only coarse abstractions of lower layer protocols, and, thus, it is not the recommended tool for studying cross-layer interactions in traditional TCP/IP networks.

Another simulation platform available for the networking community is OMNeT++ [29]. OMNeT++ is a component-based C++ simulation framework for building network simulators, providing domain-specific functionality for environments such as sensor networks, wireless ad-hoc networks, Internet protocols, etc. Although it is not a network simulator per se, it is gaining popularity as a network simulation platform in the scientific community. Similarly to ns-2, it provides general-purpose network simulation facilities and has recently been extended by Petz et al. [136] in order to accommodate certain delay tolerant experimental scenarios related to DTN routing.

Since the main target for deploying our proposed solutions was traditional IP networks we decided to adopt ns-2 as our major simulation tool due to its established status in such environments. Lack of DTN support was compensated by our own version of the Bundle Protocol (i.e. the DTN agent), inserted between the transport and application layers in the ns-2 protocol stack.

2.5.2. The ns-2 Simulator

Studying the inner workings of the selected simulator environment was a first necessary step in order to make a good design for our DTN simulation model. Ns-2 is a discrete event simulator in which actions are associated with events rather than time instances. In discrete-event simulators an event contains the execution time, a set of actions to be executed at that time and a reference to the next event. The events are linked into a chain on the simulation timeline. Simulation in ns-2 consists of two general phases. During phase 1 (the network configuration phase) the simulator creates an initial set of events according to the user-defined configuration script. In phase 2 the simulation runs and new events are created as a consequence of the initial set of created events. The simulation concludes when a certain event closes the output files and calls the exit function.

The ns-2 is architecturally split into two different domains represented by two languages, each playing a different role in the simulation process: C++ and Object-oriented Tool Command Language (OTcl). As described by Issariyakul and Hossain in their excellent ns-2 introductory book [137], C++ is used to define the internal mechanisms of the simulation objects, whereas OTcl creates more complex objects and sets up scenarios by configuring the objects and by scheduling discrete events. The C++ and the OTcl are linked together using TclCL, which maps C++ objects into OTcl variables, referred to as handles. Handles in the OTcl domain are represented as regular strings and contain no functionality whatsoever. The functionality is, instead, defined in the mapped C++ object.

The motivation behind splitting the functionality into two different domains stems from the different characteristics required from the backend vs. the frontend of a simulation tool. The backend functionality is required to execute efficiently and is not, normally, modified very often. On the other hand, the frontend functionality, which glues the backend functionality together, may frequently change (i.e. the simulation scenario) and has to be promptly translated for execution. The solution is to employ a fast-executing, compiled language, such as C++, for the backend and a fast-translating interpreted language, such as TCL for the frontend. A number of other simulation tools also follow a similar approach. For instance, OmNET++ uses C++ for the backend and the interpreted NEtwork Description language (NED) [138] for the frontend, while ns-3 uses C++ for the backend and Python for the frontend.

The majority of the work for our DTN model was invested in the backend domain of the ns-2 simulator (C++). The DTN agent itself was developed as a descendant of the Agent, a special entity available in the framework, and was placed along with the application classes of the simulator. The TCP and UDP protocols at the transport layer also had to be extended in order to support the Bundle Protocol. At the lower layers, modifications were made both to the wireless physical interface (related to energy saving features) as well as the MAC 802.11 layer (related to the bandwidth estimation mechanisms). A new type of a queue object was also implemented in order to provide feedback for actually transmitting a bundle, as opposed to simply enqueuing the bundle for future transmission.

A smaller, but still significant, part of the work for our DTN model was done in the Tcl domain. This work involved the implementation of certain types of applications (such as the space-data producing application and HTTP) and some gluing functionality connecting the DTN agent with TCP and UDP agents. For space-related simulations, additional functionality for data broadcasting and ground routing was also implemented in Tcl.

During the development of our DTN model we recognized the ns-2's architectural complexity and use of Tcl (an obsolete language with an idiosyncratic syntax) as the main disadvantages of ns-2. Presumably, these disadvantages led to the decision of the ns-3 team leaders to design ns-3 from scratch, using a modern scripting language (i.e. Python [139]) and an integrated visual tool PyViz.

2.5.3. Space-Networking Simulation Tools

Modeling of our proposed space-data dissemination scheme was done in two stages: the physical simulation and the network simulation. The physical simulation was conducted by our partners in

Telespazzio Vega [25], using the commercial modeling environment Systems Tool Kit (STK) by AGI corporation [34]. The network simulation was conducted by our team using an extended version of our DTN model in the ns-2 simulator. The following paragraphs include a brief description of STK as well as a concise report of research-oriented simulation tools for space networking available in the related scientific literature.

Systems Tool Kit (STK)

The Systems Tool Kit (formerly known as the Satellite Tool Kit) is a modeling environment for complex systems including aircraft, missiles, satellites and their sensors. STK can be used to analyze mission simulations and visualize dynamic datasets in four dimensions. STK is a high-end, commercial tool running in the Windows operating system family and is used by engineers, mission analysts, operators and decision-makers in a number of global organizations to model complex systems and evaluate their performance.

STK is appropriate for modeling the physical links between space and ground assets, including the capacity constraints of the telecommunications equipment (i.e. link budget). STK is used to model all relevant physical aspects of the mission, including the orbit geometry of the satellite, geographical locations and coverage of ground stations, as well as the physical layer of the space link communications. Mainly, these properties determine the temporal availability and performance characteristics of each space link contact.

The most important modeling capabilities of STK related to our simulation study include:

- Accurate Earth representation taking into account phenomena such as pole wander, nutation and sidereal time.
- Dynamic satellite position using a number of different orbit models such as two-body, J2 Perturbation etc.
- Dynamic vehicle orientation.
- Standard object database containing a wide range of already modeled off-the-shelf equipment.

Additionally, STK provides a wide range of analysis functionality such as line-of-sight calculations, constraints due to geometric limitation and multi-constraint satisfaction intervals, while it includes a comprehensive set of tools for visualizing the simulation model.

Custom Space-Networking Simulation Tools

Most currently available research-oriented simulation tools for space networking mainly focus, to the best of our knowledge, on the space segment, providing little support for modeling terrestrial data distribution entities. Due to these limitations, most such tools cannot be used for complete end-to-end, from space to the end-user, simulation modeling.

The authors in [140] present Galileo, a simulator written in java for the transmission of both connection-oriented and connectionless traffic in LEO and MEO satellite constellations. This particular tool allows for modeling of the satellite nodes and their respective ground stations and produces simulation results related mainly to routing and data relay performance. In [141], Smalarz combines STK with the well-known mathematical environment MATLAB [142] in order to evaluate CubeSat constellations performance, implementing a store-and-forward communications model (a feature that is currently not available in STK).

Henderson and Katz extended ns-2 with detailed packet-level simulation capabilities of various access methods employed in LEO satellite communications [143]. Simulation results, produced with the help of the presented extensions, allow for assessing LEO-specific communication link properties. Researchers in [144] and [145] model coverage ability and Inter-Satellite Link (ISL) connections within satellite constellations in STK, while simulating network characteristics, such as delay, throughput and jitter, in ns-2. The ns-2 extension presented in [144] is highly customized and, thus, suitable for simulating only specific LEO communication scenarios. Both solutions presented in [144] and [145] contain no support for store-and-forward data distributions as required by a DTN-based dissemination scheme.

Finally, Muri et al. present in [146] a hybrid platform, employing both simulation and emulation components, that aims to investigate the application of DTN communication principles in Cubesat constellations. The physical DTNs, including the data link layer, are modeled by an extension of 802.11g-2007 protocol in ns-3. Higher network layers are emulated both on real as well as virtual hardware nodes.

3. Energy-Efficient Networking with DTN

This chapter describes our work towards enhancing energy-efficiency for wireless networking devices within the context of the DTN architecture. In the immediately following section we give an account of the end-to-end operation for our proposed energy-efficient DTN overlay. In the second section, we develop a mathematical formulation for the energy-saving potential created from data buffering at the Base Station, as a function of the main system parameters. In the third section of this chapter a detailed description of the rendezvous mechanism is provided, and, finally, in the last section we describe our bandwidth estimation mechanism for 802.11 WLANs.

3.1. Energy-Efficient DTN Overlay

Our energy-efficient DTN overlay is deployed on a traditional internetworked wired-cum-wireless topology with a last-hop WLAN connection, aiming to improve energy efficiency for mobile devices. The proposed overlay exploits the inherent DTN capability for shaping internetwork traffic in a manner that allows mobile devices to balance their energy expenditure with minimal cost on throughput. More specifically, our proposed scheme attempts to increase energy efficiency by leveraging two major properties of the DTN paradigm:

- The ability to store packets for as long as it is necessary, regardless of communication disruptions.
- The ability to enhance the edge nodes with the functionality of collecting a sufficient amount of data, prior to transmitting to the end node.

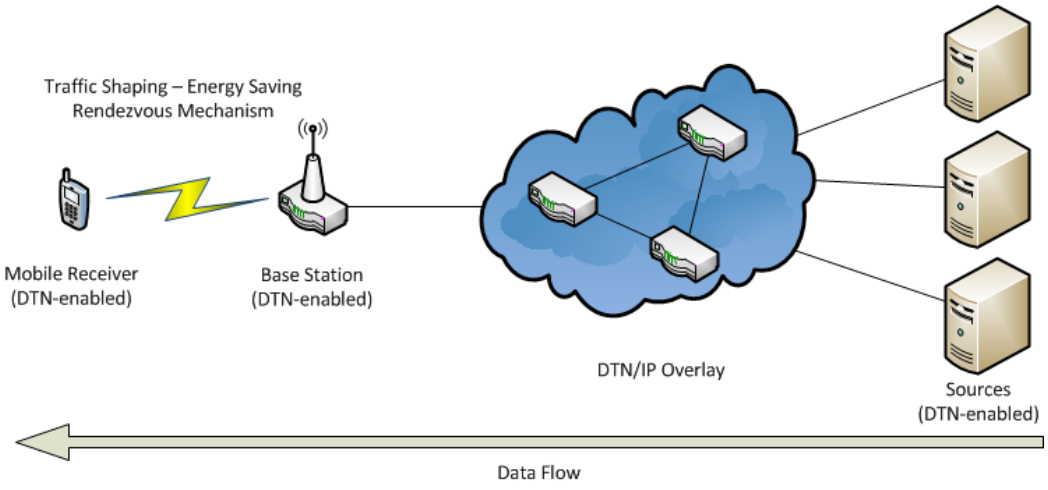

Figure 3-1: Energy-efficient DTN overlay.

A minimum deployment of the proposed energy-efficient DTN overlay involves employing DTN on three nodes: Source, Base Station (BS) and Mobile Receiver (MR). Optionally, any number of intermediate

DTN nodes can be deployed between the Source and the BS, possibly creating alternative routes that may be simultaneously active. Deployment of additional nodes allows for taking advantage of in-network storage for improving consistency of the data flow rate arriving at the BS.

The overlay supports hop-by-hop communication using any of the two main Internet transports TCP and UDP. TCP is used for applications requiring reliability such as file transfer (FTP) and web browsing (HTTP), whereas UDP is used for unreliable applications, such as audio or video streaming. The corresponding functionality is provided by the TCP and UDP convergence layers for DTN.

The proposed scheme implements our novel rendezvous mechanism into DTN. The rendezvous mechanism is enabled between the BS and the MR and exploits the traffic shaping capabilities of DTN. Based on the incoming data rate and the desired amount of buffered data the BS sets a transmission rendezvous with the MR at a future time, allowing the MR to switch its wireless interface to sleep mode, thus, achieving energy conservation. A graphical representation of the overlay is included in Figure 3-1. Our scheme was developed and evaluated with the help of our DTN simulation model in ns-2.

3.1.1. Data Packaging

At the source node of the data transfer the application layer requests the transmission of data in chunks, each of which constitutes an Application Data Unit (ADU); the smallest amount of data that can be delivered to the destination application. The bundle layer at the source node fragments ADUs with respect to the maximum bundle size and passes the resulting bundles to the transport layer. An ADU that is smaller than the maximum bundle size will not be fragmented and, thus, it will be transmitted as a single bundle. Fragmented ADUs must first be reassembled at the destination DTN node and then delivered to the destination application (i.e. an ADU cannot be partially delivered).

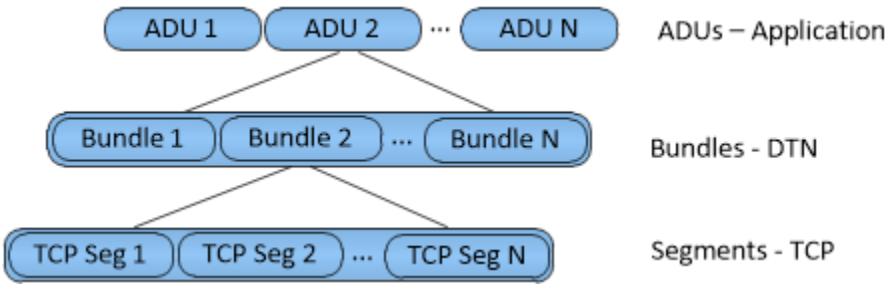

Figure 3-2: ADUs, bundles and segments.

In the case that the TCP transport is used the bundles received for transmission are further divided with respect to the maximum segment size creating multiple TCP segments. Therefore, an ADU may be contained in one or more bundles, while each of these bundles may be contained in one or more TCP segments (Figure 3-2). In case the UDP transport is used, each bundle is encapsulated in a single UDP datagram of size equal to that of the bundle.

3.1.2. Overlay Routing

Routing at the DTN overlay assumes a rhombus-like network topology, where each node maintains lists of the upstream and downstream peer DTN nodes. Bundles received from the upstream nodes are forwarded to the downstream nodes. In our prototype overlay the routes are considered to be predefined so that all downstream paths from the source lead to the desired final destination (as in Figure 3-3). Routing between any two DTN nodes in the overlay is taken care of by the underlying IP routing protocol.

Downstream nodes are used as destinations for data bundles and origins of custody signals, whereas upstream nodes are used as origins for data bundles and destinations for custody signals. Each node may have multiple downstream and multiple upstream peers, provided that the overall topology remains rhombus-like. When only a single downstream peer is registered with a certain DTN node, the incoming bundles are all routed through the single registered peer; otherwise bundles are forwarded to the downstream nodes in a round-robin fashion. When the data reach their final destination, the DTN node, instead of forwarding bundles downstream, merges incoming bundle fragments and delivers fully assembled ADUs to the attached application.

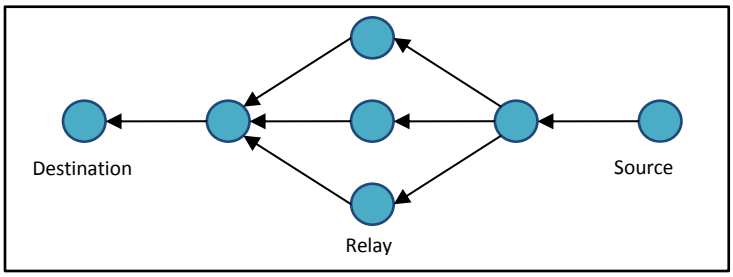

Figure 3-3: Rhombus-like network topology.

Fall et al. in [147] cite three possible strategies for routing bundles: reservation of persistent storage, in-memory store-and-forward, and cut-through in-memory routing. In cut-through routing a bundle does not need to be fully received before it is forwarded down the path, and, thus, throughput is significantly increased, especially when the bundles are of large size. Our overlay assumes hop-by-hop custody transfer for all bundles and adopts a hybrid strategy where a bundle is routed in a cut-through fashion only if there is enough space in non-volatile storage. A custody acceptance report can be viewed as a hop-by-hop acknowledgment on the DTN layer.

Each node is assumed to have a certain amount of non-volatile media storage for storing bundles it has accepted custody for. At the arrival of a new bundle the node checks for storage space availability. If adequate storage space is available, the bundle is stored and a positive custody transfer report is issued to the corresponding upstream peer. As soon as the upstream node receives the positive custody transfer report it deletes the corresponding bundle freeing storage space. In case of storage shortage, the bundle is discarded and a custody rejection report is issued to the upstream node. Subsequent bytes belonging to the

rejected bundle are also discarded. Upon receiving the custody rejection report the upstream node suspends bundle transmission and possibly schedules a future retransmission for the rejected bundle.

Applications that require reliability (e.g. FTP or HTTP) are supported by the use of TCP as the hop-by-hop transport and the implementation of a retransmission mechanism at the overlay level. The retransmission mechanism transfers custody one hop closer to the final destination by setting a timeout when transmitting a bundle. If a custody acceptance is not received within a certain period of time after bundle transmission, the bundle is resent. A timeout is also set if the downstream node rejects the forwarded bundle. The timeout in our overlay is manually set to 2 seconds, a value that was deemed appropriated for the underlying topology characteristics. However, determining an appropriate timeout value in similar DTN deployments is a topic that requires further study. For applications where reliability is not required, UDP is used as the hop-by-hop transport and both the custody transfer and the retransmission mechanisms are disabled.

3.1.3. Cross-Layer operations

Experimentation during the design phase of the energy-efficient overlay has shown that, in traditional wired TCP/IP networks, collaboration between DTN and TCP could be beneficial in order to avoid redundant retransmissions and network overload at the DTN layer. In that spirit, the overlay defines a number of cross-layer mechanisms between the bundle and the underlying transport layer that optimize network operation, especially under stressed network conditions.

A first observation leading to the development of cross-layer communication is that, under circumstances of heavy network congestion, the timeout may expire before the related bundle has actually been transmitted due to the delay introduced by the congestion/flow control mechanism of TCP. Under such circumstances the application is creating data at a rate higher than what the transport layer can accommodate, resulting in significant buffering delays. The solution to this problem is to set the retransmission timer at actual transmission time as opposed to the time when the bundle is enqueued at the transport layer. This is achieved with the help of a cross-layer *bundle transmission notification* callback through which, the transport layer notifies the DTN node that the bundle in question has been delivered to the network layer so that the retransmission timer can be accurately set.

A second observation related to the bundle-transport layer communication is that, again under conditions of heavy congestion, the retransmission timer at the bundle layer may expire while the bundle in question is in the TCP window (i.e. the bundle has been transmitted but has not yet been acknowledged). Obviously, employing the overlay reliability mechanism and retransmitting a bundle while the underlying TCP reliability mechanism with respect to that bundle is still in-progress would put unnecessary burden on the transmission process, without improving the bundle delivery probability. The solution to this problem is provided by another cross-layer mechanism the *bundle in-transit query*, which allows the bundle layer to query the underlying TCP agent about whether a certain bundle is in-transit. A bundle is retransmitted only if it is found not to be in-transit in TCP.

Our overlay includes an additional cross-layer mechanism, the *bundle transmission cancelation*, which allows the bundle layer to cancel pending TCP transmission for a certain bundle. This functionality can be employed in certain circumstances in order to offload the network from unnecessary traffic. Indicatively, the cancelling functionality can be beneficial in the following cases:

- Custody is rejected for a bundle, while TCP packets that belong to this bundle are still pending transmission. These packets can be removed from the TCP queue as they will be ultimately discarded by the receiving DTN node. This may be a common scenario, especially with large bundles, because the receiving node initiates the custody rejection report when the first part of the bundle is received. The report may reach the sending node before the entire bundle has been completely transmitted.
- Custody is accepted for a bundle, while TCP packets that belong to this bundle are still pending transmission. These packets can be removed from the TCP queue as they have already been received by the receiving DTN node. This situation may be reached if the bundle layer initiates the retransmission of a bundle and, subsequently, custody for the original bundle transmission is received. The TCP packets that contain the retransmitted bundle are redundant.

3.2. Buffering Energy-Saving Potential

In order to assess the potential of the energy-efficient overlay as well as identify the practical limitations of the proposed solution we developed a series of mathematical formulas describing the energy-saving potential of the buffering mechanism. The mathematical formulation is based on a number of assumptions that are presented in the following section. In the subsequent two sections we present the mathematical formulas and some key observations derived from these formulas. The derived observations help to better understand the experimental results presented in a later chapter.

3.2.1. Analysis Assumptions

The development of the mathematical formulation for the energy-saving potential of incoming data buffering at the BS relies on the following simplifying assumptions:

- The incoming data rate at the BS is assumed to be constant. Normally, it would fluctuate due to varying network congestion and the congestion/flow control mechanisms imposed by TCP (in case TCP was used). For streaming applications using the UDP transport, incoming data rate would present only mild fluctuations.
- The outgoing data rate at the BS is assumed to be constant. Again, the achieved data rate over the wireless link will normally fluctuate due to the operation of the underlying protocols (i.e. 802.11 MAC and TCP when applicable).
- Wireless traffic is one-way from BS to MR (i.e. activity of the 802.11 MAC protocol is not considered). Data is transmitted by the BS and received at the MR during the entire duration of a transfer at the specified data rate. In reality, the WNIC of the MR would also need to transmit

certain control frames, such as RTS/CTS or ACKs, as well as TCP acknowledgments. Therefore, the WLAN data rate is considered as the effective data rate achieved by the combination of the protocols in the network stack.

- The data to be transferred is of a certain limited amount (i.e. refers to a file rather than a data stream). Despite making this assumption, useful conclusions of the energy-saving potential analysis can be drawn for buffering in streaming applications as well.

In the formulation of the energy-saving potential of the buffering mechanism we took into account a number of parameters related to the characteristics of the data transfer, the buffering configuration and the specifications of the receiver's WNIC. These parameters are:

- *Duration*: The total duration until the entire data is received at the MR.
- *RecvTime*: The total time in the receive state for the WNIC of the MR.
- *SleepTime*: The total time in the sleep state for the WNIC of the MR.
- *Size*: The size of the transfer.
- *Buffer*: The amount of data to be buffered.
- *InRate*: The incoming data rate at the BS.
- *OutRate*: The outgoing data rate at the BS (WLAN effective data rate).
- *TransTime*: The transition time for the WNIC to switch from sleep to active or from active to sleep (the same amount of time is assumed to be required for both transitions).
- *TransPower*: The power required by the WNIC while transitioning between the Sleep and Active states.
- *RecvPower*: The power required by the WNIC while receiving data.
- *SleepPower*: The power required by the WNIC while in the sleep state.

3.2.2. Mathematical Formulation

The diagram of Figure 3-4 depicts an example scenario where $\mathrm{OutRate} = 2{*}\mathrm{InRate}$ and $\mathrm{Size} = 3{*}\mathrm{Buffer}$. Initially the WNIC is assumed to be active and immediately switched to the sleep state, allowing for the buffer at the BS to fill (i.e. for a time period of $\frac{\mathrm{Buffer}}{\mathrm{InRate}}$). The WNIC must commence the switch to the active state at time equal to $TransTime$ before the rendezvous time, so that it will be fully operational when the BS starts transmitting. Therefore, for the first sleep interval the actual sleep time is equal to $\frac{\mathrm{Buffer}}{\mathrm{InRate}}$-2*TransTime. The duration of the reception period immediately following the first rendezvous is $\frac{\mathrm{Buffer}}{\mathrm{OutRate}}$ and the duration of the subsequent sleep interval is $\frac{\mathrm{Buffer}}{\mathrm{InRate}}$-$\frac{\mathrm{Buffer}}{\mathrm{OutRate}}$-2*TransTime. The next receive/sleep period follows the same pattern; at the end, a lone reception period concludes the transfer.

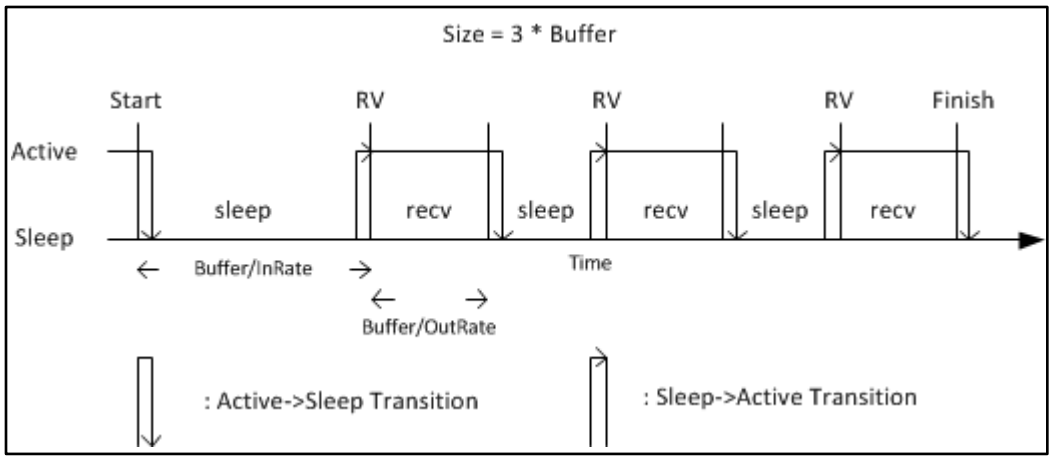

Figure 3-4: Example state transitions of a 3*Buffer Size transfer when OutRate equals to 2*InRate.

Based on the above example, the main measures in a general case can be summarized as follows:

- The total transfer duration is:

$$Duration = \frac{Size}{InRate} + \frac{Buffer}{OutRate} \qquad (3\text{-}1)$$

- The receiving time for the WNIC of the MR is:

$$RecvTime = \frac{Size}{OutRate} \qquad (3\text{-}2)$$

- Assuming instantaneous state transitions, the sleep time for WNIC during a data transfer is:

$$\frac{Size}{Buffer} * \left(\frac{Buffer}{InRate} - \frac{Buffer}{OutRate}\right) + \frac{Buffer}{OutRate} = \frac{Size}{InRate} - \frac{Size - Buffer}{OutRate}$$

- The actual sleep time is calculated by subtracting the transition time from the previous equation:

$$SleepTime = \frac{Size}{InRate} - \frac{Size - Buffer}{OutRate} - 2 * \frac{Size * TransTime}{Buffer}$$

Or,

$$SleepTime = Size\frac{OutRate - InRate}{InRate * OutRate} - \frac{Buffer}{OutRate} - 2 * \frac{Size * TransTime}{Buffer} \qquad (3\text{-}3)$$

- In order for the WNIC to be able to switch to the sleep state, the time to fill the buffer minus the time to empty the buffer must be greater than the transition time back and forth from the sleep state:

$$\frac{Buffer}{InRate} - \frac{Buffer}{OutRate} > 2 * TransTime \Leftrightarrow$$

$$Buffer > 2 * \frac{TransTime * InRate * OutRate}{OutRate - InRate}$$

- The energy spent during the transfer can be calculated by the following formula:

$$Energy = 2 * \frac{Size * TransTime}{Buffer} * TransPower + \qquad (3\text{-}4)$$

$$\left(\frac{Size}{InRate}-\frac{Size-Buffer}{OutRate}-2*\frac{Size*TransTime}{Buffer}\right)*SleepPower+$$
$$\frac{Size}{OutRate}*RecvPower$$

- For the general case where Size is not a product of Buffer the energy spent can be calculated as:

$$Energy=2*n*TransTime*TransPower+$$
$$\left[n\frac{Buffer}{InRate}-(n-1)\frac{Buffer}{OutRate}-2*n*TransTime\right]*SleepPower+ \quad (3\text{-}5)$$
$$\frac{Size}{OutRate}*RecvPower$$

  - where $n=\left\lceil\frac{Size}{Buffer}\right\rceil$

3.2.3. Energy-Saving Performance

Examining the previous formulas leads to a number of observations, especially useful for assessing the experimental results included in a later chapter of this dissertation. These observations include the following:

- Minimum energy consumption is achieved when the entire data set is buffered and then flushed at once to the MR. In this case the process involves 2 state transitions, which is the smallest possible number of transitions. As transitions take up time from the sleep state the smallest possible number of transitions achieves the highest energy conservation.
- As a side-effect of the previous point, achieving minimum energy consumption introduces maximum time overhead, resulting in a total transfer duration equal to: $\frac{Size}{InRate}+\frac{Size}{OutRate}$.
- The minimum amount of buffering that creates opportunities for switching to the sleep state is lower for small values of transition time and large values of excess WLAN bandwidth (OutRate – InRate). Large transition times and small outgoing-incoming data rate difference require large buffering values in order to exploit the energy-saving potential.

From the above observations we can deduce that there is a tradeoff between energy-conservation and data delivery latency, which is related to the amount of buffered data. For larger amounts of buffered data the energy savings are higher and the delivery latency is higher as well. Furthermore, the amount of data that need to be buffered for the scheme to work is dictated by the specifications of the WNIC (i.e. state transition time) and the amount of excess rate of the outgoing with respect to the incoming data flows at the BS.

3.3. The Rendezvous Mechanism

In order to shape statistical Internet traffic into tactic data delivery and allow the MR to conserve energy during idle intervals, the energy-efficient DNT overlay implements our novel rendezvous mechanism. The rendezvous mechanism is employed at the last hop (wireless connection) of a data transfer originated from some wired node and targeted to a mobile device on an 802.11 WLAN. Incoming data are buffered at the bundle layer at the BS and flushed at certain rendezvous times, allowing the MR to switch its wireless interface to sleep mode in the meantime. For the purpose of this section we will focus on the minimum deployment of the overlay involving the 3 key DTN nodes depicted in Figure 3-5.

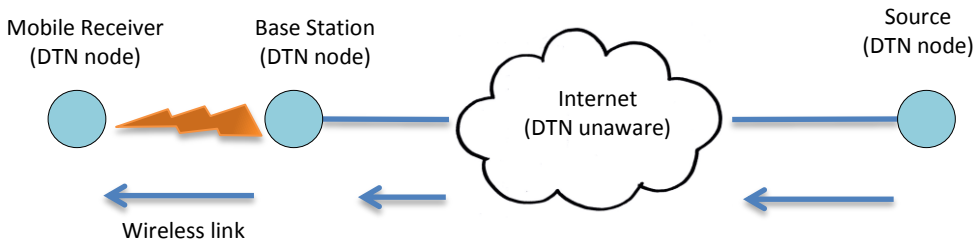

Figure 3-5: Minimum overlay deployment at Source, Base Station and Mobile Receiver.

Our original idea was to buffer an amount of data at least as large as the product of the bandwidth and the delay of the wireless link (BxD). The BxD essentially corresponds to the capacity of the link, so buffering as much data would mean that the buffer could be flushed at once, without introducing significant delays. However, the analysis presented in the previous section combined with the simulator results included in a later chapter show that, due to practical reasons arising under certain circumstances, the buffering amount may need to be significantly larger than the BxD in order to create exploitable idle periods.

3.3.1. Base Station Operation

During idle periods the base station refrains from routing incoming bundles to their final destination and simply stores them into local storage. At rendezvous time, all bundles that were partially received during the previous idle interval are dynamically fragmented to the number of the received bytes creating:

a) A fully received fragment of size equal to the number of received bytes.

b) A fragment of size equal to the remaining bytes, for which no bytes have been received.

The latter fragment may be recursively fragmented at successive rendezvous creating multiple smaller fragments, depending on the bundle size and the duration of the rendezvous intervals.

In DTN, fragmentation takes place either proactively, performed ahead of time in order to fill a contact of known duration and capacity, or reactively, performed at the event of a connection disruption in order to avoid retransmission of the already transmitted bundle fragment [1]. The dynamic fragmentation employed by our rendezvous mechanism shares characteristics from both the proactive fragmentation (it takes place

before the actual data transmission) and the reactive fragmentation (the fragmentation length is unknown until the moment the fragmentation takes place). Incoming bytes belonging to the original, unfragmented bundle can be appended to the correct fragment after consulting the segment offset field of the DTN header.

When the first bytes of a new data transfer arrive at the BS, the corresponding bundle is immediately fragmented and transmitted to the wireless node. The first rendezvous period is arbitrarily set to the initial value of 0.5 seconds, since no measurements regarding the incoming data rate are available at the time. The BS sets an alarm for the next data transfer at current time plus 0.5 seconds and communicates this value to the receiver through a field in the DTN header of the sent bundle. When the alarm is triggered the agent calculates the next rendezvous time as follows [148]:

- $BP = \frac{RB}{TBO}$: Expresses the ratio of the received bytes over the desired buffer occupancy.
    - Where BP the Buffer Portion, RB the Received Bytes and TBO the Target Buffer Occupancy.
- $SBP = BP - \frac{BP-1}{2}$: Smoothes out the BP value for faster convergence.
    - Where SBP the Smoothed Buffer Portion.
- $NextRV = \frac{RI}{SBP}$: The interval until the next rendezvous. If 0 bytes were received during the previous interval, the NextRV is set to twice the duration of the previous interval.
    - Where NextRV the next rendezvous time, RI the ReceptionInterval.

The next rendezvous interval is calculated based on the ratio of the amount of data received during the previous rendezvous interval over the Target Buffer Occupancy (TBO). The TBO corresponds to the desired amount of accumulated data that will be flushed at the next rendezvous and is set by the network administrator. Ideally, the amount of data flushed at every rendezvous would be equal to the TBO; however, since the incoming data rate is unpredictable, fluctuations do occur. If the received data exceeds the TBO, the next rendezvous interval will be shorter than the previous rendezvous interval and vice versa. The introduction of a smoothing factor limits the fluctuations so that convergence is sooner achieved. The buffer portion is smoothed to a value halfway through the buffer portion itself and 1 (i.e. 1.3 becomes 1.15, 0.7 becomes 0.85). In a future improvement the algorithm could take into account historical data, and thus, achieve convergence more promptly. A two-round example using a 30KB TBO follows:

- 1st rendezvous (0.5 sec, 45 KB of received data): $BP = \frac{45}{30} = 1.5, SBP = 1.5 - \frac{0.5}{2} = 1.25, NextRV = \frac{0.5}{1.25} = 0.4\ sec.$
- 2nd rendezvous (0.4 sec, 25 KB of received data): $BP = \frac{25}{30} = 0.83, SBP = 0.83 + \frac{0.17}{2} = 0.915, NextRV = \frac{0.4}{0.915} = 0.44\ sec.$

Once the next rendezvous is calculated the BS fragments all bundles that were partially received during the last rendezvous interval and sends the fragmented bundles, along with bundles that have been fully received, to the MR. These bundles may be originated from multiple sources and may belong to different

flows. The last bundle of the transmitted burst contains the next rendezvous time in its DTN header, while in the rest of them this field is set to 0.

3.3.2. Mobile Receiver Operation

When a bundle containing a next rendezvous value arrives at the MR, the bundle layer takes the regular action by sending a DTN custody acceptance report. The underlying TCP agent at the MR notifies through a cross-layer *bundle acknowledgment notification* callback that the custody acceptance report has been successfully received by the peer TCP layer at the BS. At this moment, the overlay checks if the time to the next rendezvous is sufficient for switching to the sleep and then back to the idle state (i.e. $\mathrm{CurrentTime} - \mathrm{NextRV} > 2 * \mathrm{TransTime}$) and, if so, it switches the wireless interface to sleep mode. In certain cases the rendezvous time may have already elapsed (i.e. $\mathrm{CurrentTime} - \mathrm{NextRV} < 0$). When the WNIC is suspended, an alarm is also set so that the wireless interface is brought back to the active state on time for the next data reception to take place. The transition time to and fro the sleep state is taken into account both when deciding whether to switch as well as when setting the alarm. In case of low transmission rate in the WLAN due to reception problems or contention, the NextRV value will be mainly in the past, essentially suspending the rendezvous mechanism, so that there is no data loss induced by the mechanism. Essentially, the mechanism is enabled only when there is excess bandwidth and implicitly disabled otherwise.

It is worth highlighting the importance of the cross-layer bundle acknowledgment notification callback. Even though the wireless interface could be suspended immediately after a rendezvous bundle is received, in such a case the custody acceptance report would not be sent and the bundle protocol at the BS would try to retransmit the rendezvous bundle. Similarly, if the WNIC is suspended before TCP receives the required acknowledgments with respect to the segments containing the custody acceptance for the rendezvous bundle, the MR TCP will keep trying to retransmit the corresponding segments, even though the interface will be suspended. Our approach allows all underlying transport operations to gracefully complete before suspending the wireless interface, so that TCP operation will not be disrupted by the energy-saving mechanism. Such seamless operation can only be achieved if a cross-layer communication scheme is in-place.

3.4. Bandwidth Estimation in 802.11 Wireless LANs

The outgoing data rate from the BS to the MR plays an important role in the proposed energy-saving scheme, as it determines the amount of time necessary to transmit the buffered data. The energy-saving potential calculations suggested that lower outgoing rate results in lower excess capacity of the wireless link, which, in turn, leads to limited possibility for exploiting idle intervals. In order to achieve energy-conservation under such circumstances, larger data amount must be buffered at the expense of additional data deliver delay. It follows that bandwidth availability information with respect to the wireless channel would allow for more educated decisions about the buffering amount and the resulting rendezvous points.

With this motivation we developed a bandwidth estimation technique on the last-hop wireless link so that higher network layers could exploit bandwidth availability information in order to make data transmission scheduling more efficient.

Even though the proposed 802.11 bandwidth estimation mechanism has not been incorporated in our rendezvous mechanism, the corresponding functionality has been implemented in the ns-2 802.11 MAC layer. Utilization of bandwidth availability information could be beneficial in a number of circumstances related to the energy-efficient overlay. For example, the DTN layer at the BS could dynamically adjust the amount of buffered data based on the available bandwidth. If bandwidth availability is lower, then more data could be buffered (allowing for better idle interval exploitation). Another possible use of the bandwidth estimation mechanism is to enable applications specify their delay requirements. For delay-sensitive applications, the amount of buffered data should be small enough so that the additional delay does not exceed the application delay-tolerance, but large enough so that some energy-conservation is achieved. Information on bandwidth availability can be helpful in order to balance the amount of buffered data so that the additional delay remains within acceptable bounds for the application.

3.4.1. Medium State Monitoring

Latest trends in estimating bandwidth availability, as described in the related work section, clearly suggest that employing a passive bandwidth estimation method, in favor of an active, probing-based one, would be the most suitable approach in our case. Since the BS will be the user of bandwidth availability information it is also the natural location for implementing bandwidth estimation. Furthermore, measuring available bandwidth requires the WNIC to constantly remain in a receive state and the BS, being connected to a power source, is the only node in the network that could support such continuous operation. If necessary, bandwidth availability figures can be communicated from the BS to the mobile nodes at rendezvous time.

The proposed mechanism is a simple, passive bandwidth estimation technique for 802.11, fitted into the wireless MAC layer of the BS. The mechanism records busy periods for the medium, as well as idle periods that are part of the collision avoidance strategy of 802.11, taking into account both the physical and the virtual Carrier Sense mechanisms of 802.11. Thus, our algorithm considers as busy periods:

- Reception of segments: Actual reception of frames from any node except the BS to any node including the BS on the WLAN. The BS continuously receives frames, even if it is not involved in the communication, so that busy medium intervals are always recorded.
- Transmission of segments: Actual transmission of frames from the BS to any other node on the WLAN.
- Back-off periods: Periods of inactivity at the BS due to the back-off, congestion avoidance mechanism.
- Inter-Frame Spaces (i.e. Short IFS, Distributed IFS, etc.): Periods of inactivity that are part of the MAC IFS mechanism for prioritized medium access.

- Transmission deferral due to Virtual Carrier Sense (VCS): Periods of inactivity that are part of the Network Allocation Vector channel reservation mechanism.

It is quite likely that the BS may be simultaneously in two of the above states, for example, the BS may be backing-off or deferring transmission due to VCS, while segments are received from the medium. The mechanism ensures that such intervals are recorded only once.

3.4.2. Bandwidth Utilization Calculations

The busy and total duration amounts are stored in the first elements of two arrays and they are updated every time there is a switch in the state of the algorithm (i.e. busy-to-idle, idle-to-busy). A timer is set off at regular intervals and shifts the measurements in both arrays by one position to the right (it resets the first element and discards the last element of both arrays). The implemented version of the mechanism involves 20 slots of 0.5 seconds for each array, so that utilization record is maintained for the latest 10 seconds. Measurements older than 10 seconds do not contribute to the bandwidth availability calculations. The number of the time slots in the two arrays, as well as the time interval for the shifting of the values can be easily changed to accommodate for various bandwidth estimation scenarios.

Time (sec)	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Timeslots	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Weights	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

Figure 3-6: Bandwidth estimation utilization timeslots and corresponding weights.

When higher network layers query for the available bandwidth estimate, recent measurements have a higher contribution to the calculated utilization than measurements that are farther away into the past. The values for the selected weights are 20, 19, 18, …, 1 for the 20 time slots starting at the most recent slot (Figure 3-6). The utilization value for each time slot is multiplied by the corresponding weight and added to the weighted utilization sum. Finally, this sum is divided by 210 (the sum of the weights) in order to calculate the weighted average. Higher layers may query the 802.11 MAC for the channel utilization via a cross-layer communication mechanism. The calculated channel utilization figure is combined with the maximum channel capacity giving the available residual bandwidth.

4. Decentralized Space-Data Dissemination with DTN

In this chapter we present our DTN-based, space-data dissemination paradigm, developed in cooperation with our partners at Telespazio Vega [25] and the European Space Agency (ESA). The work presented in this dissertation is a continuation of the study submitted to ESA, including significant improvements over the original solution and more comprehensive simulation experiments. The corresponding study has been published in [32].

The first section of this chapter gives the motivation behind our work and an overview of the proposed solution. The second section includes a classification of missions with respect to their space-data requirements, enabling us to appropriately adjust data transmission in response to each mission type. The third section describes the three space-link models that correspond to the associated types of space communication equipment regarded in this study. The space-link models are incorporated in the end-to-end, data delivery models, which are described in section four. Finally, the fifth section in this chapter focuses on the satellite data production and transmission scheduling mechanisms that participate in the proposed space-data dissemination scheme.

4.1. Overview

The work towards the decentralized space-data dissemination DTN overlay has been conducted in response to ESA's statement of work presented in [24]. The motivation for the statement of work on behalf of ESA was to explore alternative space-data dissemination paradigms inspired by the recent trend for the development of dense, low-cost LEO satellite networks. Our space-data dissemination DTN overlay was designed to support such LEO satellite missions. Contrary to traditional missions run by major space agencies, in low-cost LEO missions the contact opportunities between satellites and ground terminals may be relatively frequent, of limited capacity and often unidirectional in nature (i.e. from space to ground). Additionally, such missions may involve a large number of interested end-users, subscribed for receiving scientific data collected by mission satellites in a timely fashion.

One of the main goals of the study was to produce comparative simulation results between the current LEO satellite practices, followed by major space agencies, and the newly proposed paradigm. These comparative results combined with the cost associated with each type of solution may be utilized for a high-level, cost-benefit analysis of the proposed solution vs. the established practices. Thus, modeling of both the established space-data dissemination scheme as well as our proposed alternative paradigm was necessary.

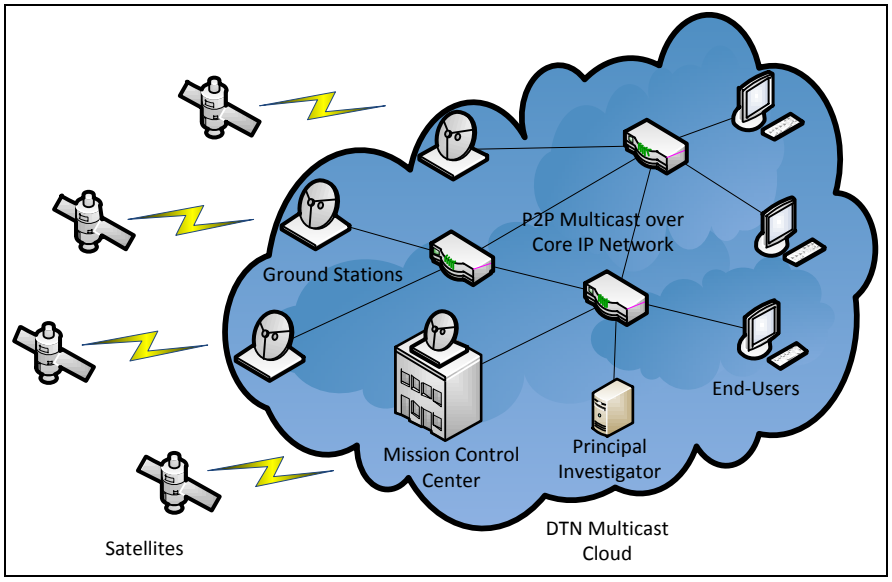

Figure 4-1: Satellite data broadcasting with P2P multicast ground distribution.

The study was based on detailed simulation modeling facilitated by a combination of two simulation tools; the Systems Tool Kit (STK) and ns-2. STK was used by our project partners at Telespazzio Vega to model the physical properties of the system, such as satellite orbits, ground station locations, radio equipment reception range etc. and, subsequently, calculate the resulting contact characteristics. The contact characteristics calculated by STK are incorporated into an end-to-end architecture modeled in ns-2, allowing the performance evaluation of the system under various circumstances. In order to simulate the end-to-end network we developed an in-house model, extending the functionality of our ns-2 based DTN model. The end-to-end simulations reflected both the current practices as well as the newly proposed dissemination overlay so that comparative results could be produced.

The space-data dissemination paradigm consists of a best-effort, broadcast transmission scheduling scheme with optional delivery feedback in space and a DTN-based, P2P multicast overlay network on the ground (a high-level diagram can be seen in Figure 4-1). The transmission scheduling mechanism can be adjusted in order to accommodate for various delivery requirements, depending on the type of space-data involved. Space-data may be consumed by a wide set of applications, ranging from water management to large-scale mapping and from space-weather monitoring to soil protection. This diverse nature of space-related applications creates the need for different types of data transmission models.

4.2. Mission Requirements

We classify space mission applications and their corresponding data with the help of a two-dimensional plane across two axes as depicted in Figure 4-2. Vertically, applications are placed on a temporal criticality axis with respect to the time-sensitivity of the collected data. The top half-plane contains applications involving data of high temporal criticality, such as real-time or near real-time services. Data in such

applications have a short expiration time and can quickly become obsolete if their delivery is delayed. Applications in this category include flooding or fire mapping, which typically require real-time data in order to produce meaningful results. Indeed, space-generated photographic data of disaster-struck areas need to be transmitted as soon as possible; otherwise they may not be useful to emergency response personnel that operate in the area. On the contrary, data for off-line services such as water management or land mapping can be useful even if their delivery is significantly delayed.

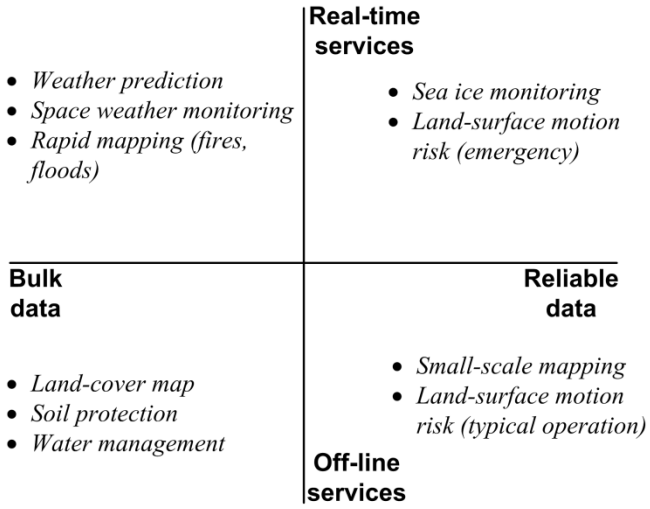

Figure 4-2: Categorization of Earth Observation applications depending on the type of space-data they consume [32].

Applications on the horizontal axis are placed according to their data reliability requirements and characterized as bulk or reliable. Bulk data applications are benefitted from a high amount of data received on the ground even at the expense of a lower data delivery ratio. Possible high data loss ratio with respect to the collected data is not crucial to the application as long as the overall amount of delivered data remains high. Applications in this category include lower resolution land-cover mapping or real-time applications where high resolution imaging is not necessary. An example of this type of applications is space weather monitoring, where data containing radiation measurements around the globe are transmitted in near-real time to allow for accurate predictions. In the other end, reliable data applications involve targeted data and, thus, require high reliability even at the expense of overall data volume. This category includes applications such as small scale mapping, since any lost data will result in missing tiles on the map.

It becomes apparent from the previous paragraphs that, considering the diversity of Earth observation missions and their corresponding data requirements, a single data production and distribution pattern fitting all mission types cannot exist. With that in mind, our proposed communication model enables mission planners to adjust the system parameters and optimize performance and efficiency according to the targeted application.

4.3. Space-Link Models

Modeling of the space-links took place in the STK simulator and enabled performance comparisons among traditional and newly proposed satellite communication patterns. The space-link models fall into the following three categories: *Direct point-to-point*, *relay point-to-point* and *low-cost broadcast*. The point-to-point models represent communication patterns that are currently in use by space agencies. These models assume bidirectional communication and support immediate feedback by the receiving ground stations for the downlinked data. On the contrary, the low-cost broadcast models correspond to the recent developments that focus on large numbers of inexpensive satellites and ground stations, where communication may often be unidirectional, as it is possible that a large number of receiving ground terminals do not have the capability and/or the authority for uploading data.

In the direct point-to-point communication pattern we consider short, direct, high-speed contacts between a single satellite and 3 ground stations (one at any given time). This is similar to a traditional setting where a satellite bearing high-end communication equipment connects to a limited number of ground stations with matching equipment, once per orbit, over bidirectional links. In the relay point-to-point case we consider a single high-end satellite that communicates with 3 GEO relay satellites (again one at any given time), each relaying data to its corresponding ground station on Earth. Due to possible multiplexing at the relay satellites, the bandwidth of the inter-satellite link is assumed to be lower than the bandwidth of the direct connection. However, the relay connection yields much higher overall downlink capacity due to the almost continuous visibility provided by the GEO relays. Links under this model are also bidirectional and use the European Data Relay System [102] as the reference communication platform.

The low-cost broadcast link model corresponds to the new communication paradigm and assumes low-cost satellites with matching ground equipment, yielding a large number of short, low-speed, and possibly overlapping contacts. These links may be unidirectional or bidirectional depending on the communicating ground station. In a typical scenario, a low-cost LEO satellite will be, for the most part of its orbit, broadcasting data to any interested receiving parties in a unidirectional manner. When coming within range of certain designated ground stations, bidirectional links will be established. Communication over these bidirectional links may involve both downlink of acknowledged data as well as uplink of satellite management commands. The uplink may also include acknowledgments and other control information, possibly related to data received from a previous unidirectional link, that was buffered at the designated uplink ground station.

The low-cost broadcast communication pattern has been applied to various simulation topologies with one or more satellites and a dense network of ground stations. Namely, the topologies contain 1 satellite with 52, 100 and 140 ground stations and 6 and 46 satellites with 52 ground stations. The topologies reflect real ground station locations retrieved from the STK database and orbits that are compatible with scientific missions involving dense satellite constellations.

4.4. Data Delivery Models

The data produced by scientific instruments on board the satellite are transported over the space links described in the previous section and, through the network, to interested end-users. Satellite data are tagged with a Type-of-Service (ToS) identifier allowing the end-users to subscribe only to a subset of the overall data. End-users subscribe to the desired ToS in order to receive only the data they are interested in. In most scientific missions satellites host multiple instruments collecting measurements for a wide variety of scientific fields. In such cases, different ToS identifiers may correspond to different scientific instruments, or groups of instruments whose measurements target the same application. The rest of this section includes an overview of the end-to-end data delivery models highlighting the relationship between the space-link type and the ground segment layout.

4.4.1. Network Elements

The data delivery models involve the following network elements:

- Satellites (Sat): The satellites collecting scientific measurements and broadcasting or exchanging data with in-range ground communication equipment.
- Ground Stations (GS): Regular ground stations capable of receiving satellite broadcast data. Ground stations are connected to the global Internet.
- Mission Control Centers (MCC): In traditional centralized space-data dissemination schemes the MCC collects and assembles all data before making them available to any interested party.
- End-users (EU): The end-users, mainly academic and research institutions, interested in receiving scientific data collected on the satellite. Normally, EUs will subscribe for receiving only a subset of the collected data.
- Principal Investigators (PI): A principal investigator is the entity responsible for organizing a decentralized mission. In such missions, PIs receive all of the data collected as part of the mission and produce appropriate acknowledgments for the satellite. PIs may also be used for archiving historical mission data.
- Uplink Ground Stations (UGS): Ground stations with uplink capabilities. In traditional high-end missions all GSs are UGSs providing instant acknowledgments over point-to-point links. In decentralized missions there is usually at most one UGS, which may be flushing uplink data originated from a PI.

Satellites, ground stations and end-users are included in all models, whereas a mission control center, a principal investigator and uplink ground stations take part in a subset of the models.

4.4.2. Ground Distribution Models

The ground segment layout may be either centralized or decentralized. In a centralized layout, satellite data received at any GS are forwarded directly to the MCC. The MCC is then responsible for assembling the data and forward them to the interested EUs in a client-server fashion. This is in accordance to the

space agency portals referred to in the related work chapter. Space-data in this centralized dissemination scheme are collected at the MCC and downloaded by the interested parties from the corresponding portal. In the decentralized layout, satellite data are forwarded to the end-users over a P2P multicast network, similar to a push version of the BitTorrent protocol. For each GS and ToS there is a separate DTN overlay multicast distribution tree, where the GS is the root of the tree (i.e. each GS is the root of as many distribution trees as the available ToS). Bundles received at a GS are forwarded down the appropriate distribution tree of the P2P multicast network. Figure 4-3 depicts an example scenario of a single ToS per GS, where colored lines represent the corresponding P2P multicast distribution trees.

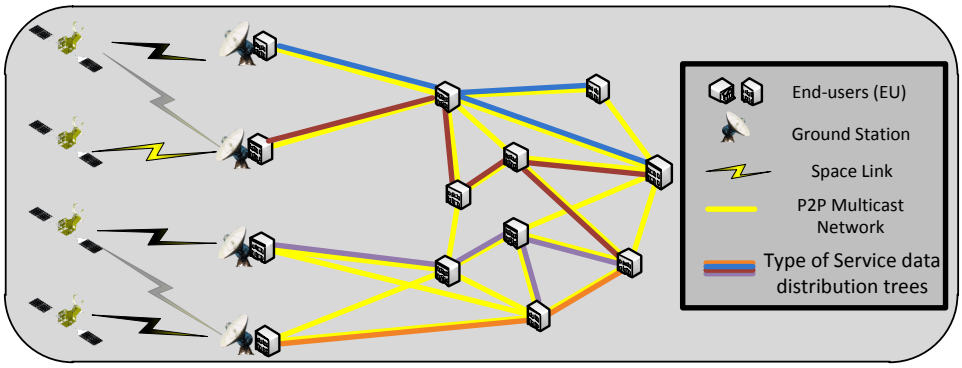

Figure 4-3: Example of a single type-of-service multicast distribution tree per ground station.

All EUs run the Bundle Protocol, through which the DTN architecture is implemented, and collaborate in the DTN multicast distribution tree. The routing within the distribution tree is dynamically updated when an EU joins or leaves the corresponding ToS of the group. The DTN distribution model exhibits the characteristics of an application layer multicast overlay. The literature includes various such solutions, (e.g. ALMA, ALMI, Narada, Yoid Overcast, etc.), most involving some central authority assuming an elevated role. A comprehensive survey on the topic can be found in [149]. Additionally to the multicast infrastructure, designated servers may hold historical data for off-line access by clients that were not part of the distribution overlay at the time of the original data transmission. This would obviously be of higher value for non-time-critical data.

Due to the broadcasting nature of the transmission, as well as the employed retransmission mechanism, it is possible that multiple copies of the same data are received by multiple GSs and, consequently, forwarded down multiple distribution trees. Therefore, as each node may belong to multiple distribution trees for the same ToS, the same bundle may be received at a node from multiple peers. In such cases, a bundle-level immediate pruning mechanism is employed. According to this mechanism, the receiver of a duplicate bundle generates a custody rejection report destined to the originating peer. The peer receiving the rejection report refrains from forwarding any more data belonging to the rejected bundle over the same multicast tree branch, thus conserving network resources. This mechanism is realized by examining the

first bytes of an incoming bundle, where the bundle identification is contained, and rejecting the bundle in case it has already been received.

The proposed space-data dissemination scheme supports optional uplink functionality, in which case acknowledgments are generated either directly at the receiving GS or at a PI (possibly subscribed to all ToS) and routed to a designated uplink GS. At the next available contact the uplink GS forwards all stored acknowledgements to the satellite.

4.4.3. End-to-End Models

The point-to-point space link models described in the previous section (direct and relay) are combined with a centralized ground segment, representing the current practice in organizing space missions. Here we assume bidirectional space links with ground stations sending acknowledgments over the uplink when they successfully receive data. The corresponding end-to-end models are referred to as *direct* and *relay* and they provide the performance benchmark for current, high-end LEO missions with and without the use of relay satellites respectively.

The emerging low-cost satellite model is represented by combining the low-cost broadcast link model (presented in the previous section) with both a centralized and a decentralized ground segment, referred to as *broadcast client-server* and *broadcast P2P* respectively. The broadcast P2P model reflects a fully distributed network setting where end-users have immediate access to satellite data received at nearby ground stations, and corresponds to the architecture proposed in the original ESA statement of work. The broadcast client-server model is used comparatively with broadcast P2P in order to quantify the benefits of a distributed vs. a centralized ground data distribution approach. In the broadcast P2P model, one of the EUs can be designated as the PI, responsible for generating acknowledgments, and one of the ground stations as the uplink (UGS), responsible for transmitting the acknowledgments to the satellite. The acknowledgments are sent from the PI to the UGS and transmitted during the next contact with the satellite.

In a real-world deployment, data produced on the satellite could be transported as DTN bundles or CFDP data units, encapsulated over the space link in TM packets. Ground stations receiving the satellite data should be aware of the higher layer protocols and forward incoming packets onto the appropriate network nodes over Internet protocols, depending on the implemented distribution scheme. The end-to-end system architecture is depicted in Figure 4-4.

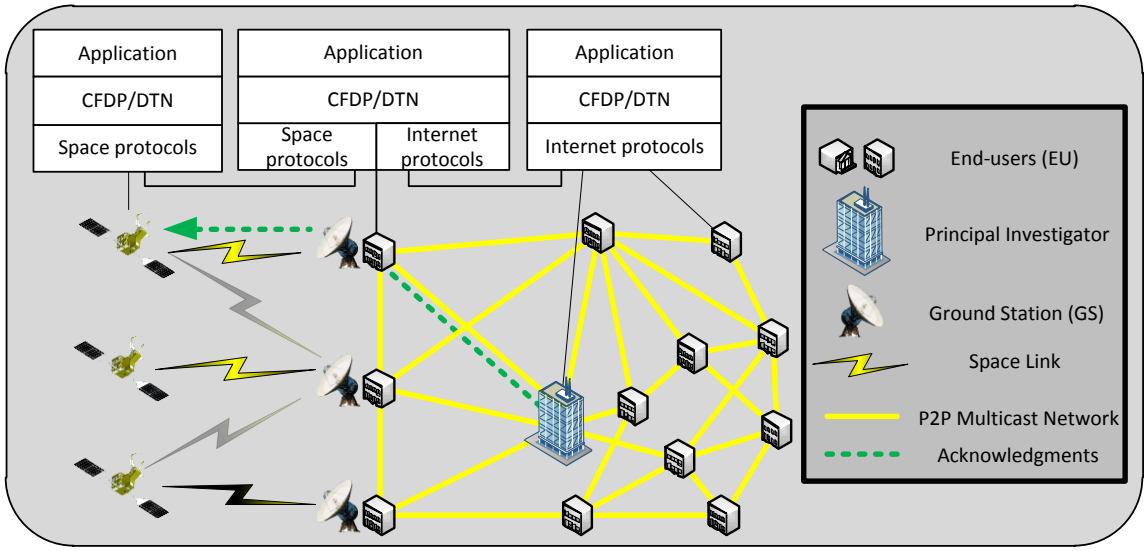

Figure 4-4: End-to-end architecture and indicative protocol stacks for space and ground.

4.5. Data Production and Transmission Scheduling

4.5.1. Application Data Production

Application data production on the satellite is assumed to follow a uniform pattern. Scientific instruments continuously acquire data at a constant rate, package it into Application Data Units (ADUs) of a specified size, and pass them on to the networking subsystem. In order for an ADU to be created, the appropriate amount of data must be first acquired; thus, the networking subsystem receives ADUs of the specified size in regular intervals. The size of the ADU must be greater than or equal to the minimum useful granule of data for a certain application (e.g. an Earth observation image and associated metadata, or a set of measurements). For performance reasons multiple data units may be combined into a single ADU, especially in case of extremely small data units (e.g. a single temperature measurement).

Each ADU is tagged with the ToS identifier of the corresponding application or, equivalently, scientific instrument. Multiple applications may be running on the same satellite, some of which possibly sharing the same ToS identifier. This way application data can be grouped into semantically similar categories, despite the fact that they may be produced by separate scientific instruments. Conversely, a single instrument may be producing data of multiple ToS based on a predefined pattern. Since EUs may subscribe for multiple ToS, a finer-grained classification of space-data could be compensated by multiple subscriptions on the ground. However, classification of data into broader categories in space does not allow for later category separation since information on the data origin is lost.

4.5.2. Transmission Scheduling

The networking subsystem fragments incoming ADUs into a number of bundles, according to the specified bundle size, and inserts these bundles into the transmission buffer. Old bundles may need to be removed in case a certain buffer threshold is exceeded, thus, imposing an implicit data time-to-live (TTL). Bundles may be transported via diverse routes and re-assembled at the destination nodes (the mission control center or some end-user).

The transport agent at the satellite continuously broadcasts bundles at the rate imposed by the wireless channel bandwidth. Newly created data are given higher priority over data that have been transmitted at least once, so new bundles are transmitted as they arrive in a first-come-first-served order. When all bundles have been transmitted at least once, an ADU is randomly selected from the buffer and transmission resumes with the first bundle belonging to the selected ADU. Transmission continues sequentially until all bundles belonging to the selected ADU have been transmitted, at which point a new ADU is randomly selected for retransmission and the process iterates. If at any point during the randomized retransmission phase new bundles arrive, the process is interrupted and the new bundles are transmitted first.

Figure 4-5 shows an example case where the transmission buffer contains 4 ADUs of 5 bundles each. While transmitting bundle 3 of ADU 2 a new ADU arrives (ADU 4) interrupting the retransmission of ADU 2 so that ADU 4 can be transmitted for the first time. As soon as all bundles belonging to ADU 4 have been transmitted, retransmission of ADU 2 resumes with the transmission of bundle 4. When ADU 2 finishes retransmitting (i.e. bundle 5 is transmitted), one of the four stored ADUs is randomly selected for retransmission and the process repeats itself. Depending on the size of the buffer, the arrival of new data may cause the deletion of the oldest data in the buffer (i.e. bundle belonging to ADU 1).

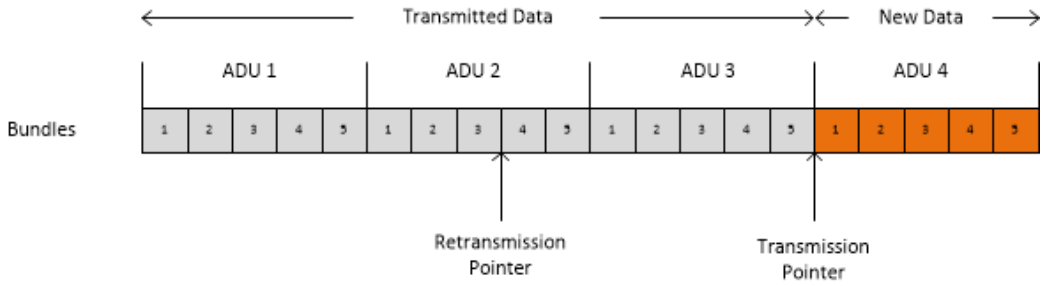

Figure 4-5: Transmission scheduling example.

For a given downlink bandwidth, the combination of the data production rate and the data TTL allows for adjusting the transmission scheduling mechanism to better suit the needs of the application involved. Generally, a low data production rate favors delivery reliability, whereas a high data production rate favors delivery volume. The data TTL also plays an important role in the transmission scheduling process as it dictates how retransmission effort will be distributed among old and recently created bundles. A high TTL value favors older bundles while a low TTL value favors newer bundles. Thus, a high TTL would generally

favor offline services, whereas a low TTL would favor time-sensitive services. A deeper discussion with respect to appropriately adjusting the system parameters for various types of space-data applications is included in the experimental results section of this dissertation.

4.5.3. Supporting Mechanisms

At the event of an acknowledgment reception, the corresponding bundle is removed from the buffer so that future retransmission is avoided. If reception feedback is not available (as in the case of broadcasting with no uplink capability), it is possible that bundles already received on the ground may be retransmitted, wasting downlink bandwidth. This is also possible during the time the relevant acknowledgments are buffered at the uplink ground station awaiting the next available contact. If the receiving ground station itself initiates the acknowledgments (as in the point-to-point models), unnecessary retransmissions are eliminated. It is evident that a delivery feedback mechanism is an optional enhancement, but not a requirement for the system.

Transmission at the satellite may be limited only when at least one ground station is within range. The benefit from limiting transmission is two-fold:

- Transmitted bundles are more likely to be received on the ground (they are transmitted only when someone is receiving)
- Valuable satellite energy is conserved.

In case of point-to-point connections (i.e. direct and relay point-to-point models) information on the presence of a receiving ground station is readily available due to the bidirectional nature of the physical link. In the broadcast models such information is not available, due to the general lack of an uplink channel, calling for some out-of-band mechanism. Transmission intervals may either be directly preloaded on the satellites or, alternatively, the ground station locations may be preloaded and transmission could suspend/resume depending on the satellite location.

5. Storage-Constrained Routing in Opportunistic DTNs

In this chapter we present our preliminary study on employing Game Theory-inspired mechanisms in opportunistic DTN routing. Section 5.1 describes the network topology and the motivation behind the selected design; section 5.2 defines the storage-constrained routing problem and discusses the game theoretic aspect of a storage-constrained routing algorithm. Finally, the last section in this chapter lists the desired bundle acceptance characteristics adopted from the related literature, and describes the specific bundle acceptance criteria, evaluated in the context of this work.

5.1. Network Topology

The nodes participating in the opportunistic DTN need to route local traffic through a certain subset of their peers, essentially creating conditions of mutual reliance. Our study considers a carefully selected network topology in which the nodes comprise two symmetric groups, top and bottom, forming a partial mesh, where all top nodes are connected to all bottom nodes (Figure 5-1). The important characteristic of the selected topology is that the top nodes need to route locally-created traffic through the bottom nodes and vice versa.

Routing takes place in a round-robin fashion such that each new bundle is forwarded to the next node in the peer group. Although this topology reflects a wired setting, the mesh between the top and bottom node groups combined with the round-robin routing pattern resembles a wireless topology, where nodes in the two groups come into occasional contact with one another (i.e. during the time of bundle forwarding). Therefore, this preliminary study could be extended to encompass an actual opportunistic topology, involving wireless connections among nodes with real mobility.

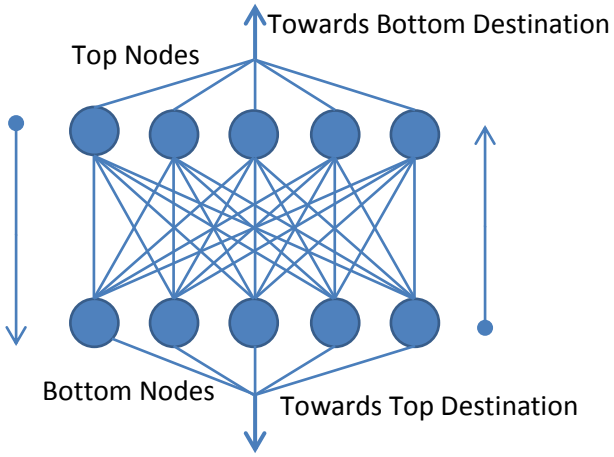

Figure 5-1: Partial mesh topology creating mutual reliance among top and bottom nodes.

5.2. Storage-Constrained Routing

The presented study considers storage constraints in the participating nodes. Bundles created locally must be first stored in persistent storage on the originating node before being forwarded. In case the originating node lacks sufficient space the newly created bundles are dropped. Once local bundles have been saved into local storage they may then be routed via a peer node towards their final destination. In order for a node to accept routing of a peer's bundle it must reserve sufficient space for storing the incoming bundle. If sufficient space is not available the node must reject the incoming bundle and notify the originating node, which, in turn, registers the bundle as "not delivered" and drops it.

Our initial plan was to enable bundle retransmission using a constant retransmission timeout for rejected bundles. However, due to the reliance of peer nodes the behavior of the system was very intolerant to modifications in the storage space size. Enabling the retransmission mechanism implies that bundles are removed from storage only when they are accepted from the downstream node, rendering the network deadlock-prone under heavy load. On the contrary, when retransmissions are disabled a bundle is immediately dropped if custody for it is rejected, allowing the network to continue servicing bundles at a lower delivery ratio under heavy load. In a more general setting where the nodes may be able to directly deliver a locally created bundle to its final destination, without the need for peer routing, bundle retransmission could be enabled.

The decision of whether to accept an incoming bundle may not always be as straightforward as checking for sufficient storage availability. Instead, acceptance may depend on other factors such as local storage space utilization level, past experience with the originating peer, intention for reserving storage space for local use, etc. Our work in these opportunistic DTN settings mainly involved developing appropriate bundle acceptance criteria in order to achieve high overall efficiency and fairness by properly allocating the network storage. As described in pervious paragraphs, each node uses local storage both to store its own, locally produced bundles, as well as to store bundles that need to be relayed on behalf of its peers. Therefore, each participating node competes with its peers for local storage, but, at the same time, relies on the peer's generosity for delivering its bundles.

Reasonably, if all nodes are impartial and accept bundles regardless of origin (i.e. local or relay) as long as there is storage availability, the overall network performance and fairness will be high. However, a selfish node entering the DTN under such circumstances could take advantage of its peers by refusing to relay foreign bundles and reserve all storage for locally-produced traffic. As a result, the selfish node will achieve higher throughput than the rest of the nodes, leading to low fairness. Additionally, the overall performance of the network, in case selfish nodes are present, may be lower as compared to the case were all nodes are impartial. Ideally, nodes must be impartial when their peers are also impartial, so that the overall network health to remain high, but they must refrain from offering local resources to peers who appear to be selfish, so that such nodes are isolated and network fairness remains at acceptable levels.

5.3. Bundle Acceptance Criteria

Our approach involves maintaining at each node acceptance history of locally produced bundles with respect to each of the node's peer. When a peer requests relaying of a bundle the receiving node retrieves the peer's history and decides on accepting or rejecting the incoming bundle based on this history. The acceptance criteria are developed with a few basic guidelines inspired from Game Theory [27][130] such as:

- Tit-For-Tat (TFT): A node starts positively by servicing a peer node and then returns the behavior it receives from this node (positive reaction to a positive action, negative reaction to a negative action).
- Initial Generosity: TFT is modified so that nodes maintain their positive behavior until they collect sufficient experience with respect to each of their peers. This essentially prolongs the positive action, enhancing "good faith" among network nodes.
- Vendetta mitigation: Nodes tend to "forget" their past experience as the time elapses, returning to the Initial Generosity phase. This way peer's behavior becomes more forgiving allowing for quickly restoring normal network operation.

Each node keeps track of the bundle acceptance history it experiences from its peers. History about each peer is maintained in the form of two arrays of size 5, where each element corresponds to a time duration of 5 seconds. One of the arrays holds the accepted and the other the total number of bundles for a peer node. These values can be easily changed to suit alternative scenarios; for the selected values, however, the peer history contains data for the previous 25-second period.

When a bundle is accepted the first elements of both arrays are incremented, whereas when a bundle is rejected the first element of only the array holding the total bundles is incremented. Every 5 seconds the arrays are shifted to the right, discarding the content of the last element of each array and setting the first elements to 0. At any point a node can query the DTN layer for the Bundle Acceptance Ratio (BAR) with respect to a peer node and DTN will calculate a weighted average of the acceptance ratio for the connected peer. The weights in a general case of n array slots are selected as: $\frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{n+1}, \frac{1}{n+1}$. In the specific case of 5 array slots the weights are: $0.5, 0.25, 0.125, 0.0675, 0.0675$. In case the samples for a certain time slot are fewer than 3 the ratio for the time slot is considered as 1 (highest possible), giving untried connections the benefit of the doubt. Additionally to the BAR, a node calculates the StorageOccupancyRatio as:

$$StorageOccupancyRatio = \frac{OccupiedSpace}{AvailableSpace}$$

In order to compare different bundle acceptance strategies a node is given the option of using bundle acceptance criteria belonging to one of the four main categories:

- Impartial: A bundle is accepted if enough space is available. In this case the node makes no distinction between local- and peer-originated bundles.
- Totally Selfish: Only accepts local bundles, provided that enough storage space is available.

- Partially Selfish: Local bundles are accepted if enough space is available, whereas foreign bundles are accepted with a probability equal to 1 – StorageOccupancyRatio. A node that follows this strategy accepts foreign bundles with a lower probability as the available storage becomes scarce, favoring locally produced bundles. In case of high storage availability the node accepts incoming bundles with high probability. Essentially, the node behaves selfishly only when necessary.
- History-based: Utilize the BAR as part of the bundle acceptance criterion.
    - Deterministic: A bundle is accepted if BAR > StorageOccupancyRatio.
    - Strategy 1 (non-deterministic): A bundle is accepted if Random(0, 1) * BAR > StorageOccupancyRatio.
    - Strategy 2 (non-deterministic): A bundle is accepted if Random(0, 1) * BAR > $StorageOccupancyRatio^2$.

Where, Random(0, 1) a number randomly selected between 0 and 1. It is noted that locally produced bundles are always accepted by the node as long as there is available storage.

The above criteria were tested in various combinations in order to evaluate their performance. The experiments included combinations such as: all nodes being impartial, some nodes being impartial and some nodes being partially selfish, some nodes following different history-based strategy than others etc. The suggested mechanisms were evaluated on a basis that is two-fold:

- When all nodes behave well and cooperate with each other the overall network performance must be as close as possible to the case where no mechanism is employed, whatsoever.
- In case selfish nodes are present, the application of the mechanism must result in punishing the selfish node and securing fair allocation of network resources for the rest of the nodes.

On the above evaluation basis the “deterministic” and the “strategy 1” mechanisms were found to be inappropriate for use in the tested network setting and, thus, rejected. On the contrary, the “strategy 2” mechanism was found to perform satisfactorily and so it was accepted as the most appropriate for the tested setup. The experiments showed that one of “strategy 2” could indeed achieve good network utilization when all nodes are well-behaved, while at the same time punish nodes that choose to be non-cooperative. In such a setting, a network node would refrain from behaving in a selfish manner, as this would ultimately lead to poor performance for the node itself. Detailed presentation and analysis of the experimental results as well as the performance of each proposed strategy are included in chapter 8.

6. Experimental Methodology

The solutions proposed in this work towards a PhD degree were developed in parallel with their detailed simulation model counterparts. The development process shares characteristics with the agile software development methodology [150], and follows the general pattern described below:

a) A simple model is initially created and employed in a number of simulation experiments. These experiments are used to conduct an initial proof-of-concept study that helps to quickly evaluate the feasibility of the original idea.
b) The proof-of-concept study may highlight important deficiencies that are subsequently addressed by applying certain core modifications to the original solution.
c) The simulation model is gradually refined with the addition of more detail to the suggested solution. As the simulation model becomes more refined, new issues surface and the proposed solution is accordingly improved to address those issues.

The simulation models were implemented in the ns-2 network simulator environment. As described in previous chapters, the main reasons for selecting this particular network simulation environment were the provision of detailed implementations for protocols throughout the entire protocol stack and the wide recognition that ns-2 receives from the computer networking research community. However, even though ns-2 provides detailed implementations of Internet protocols, which is crucial to the development of our solutions, it lacks support for a store-and-forward overlay that can be used on top of the provided protocols in order to simulate the DTN bundle layer. This lack of support was compensated for with the development of custom simulator modules.

The rest of this chapter contains specific descriptions of the experimental methodology for each of the three main solutions proposed in this PhD work. Namely, section 6.1 describes the methodology for the energy-efficient DTN overlay simulations, section 6.2 describes the methodology for the DTN-based, decentralized space-data dissemination simulations and, finally, section 6.3 describes the methodology for the storage-constrained routing in opportunistic DTNs simulations.

6.1. Energy-Efficient DTN Overlay

In order to evaluate the applicability and performance of the proposed energy-efficient DTN Overlay, we conducted a wide variety of experiments in ns-2 starting with a simple proof-of-concept model, which we gradually evolved into a detailed DTN simulation model. The initial proof-of-concept model involved a minimal deployment of the overlay and implemented the rendezvous mechanism by extending the Application module included in the simulator. This model was capable of simulating both FTP and Constant Bit Rate (CBR) streaming traffic and the energy-expenditure was calculated off-line during post-processing of the simulation outcome. At the second development stage we created the DTN agent, which enabled the deployment of a full-blown bundle overlay supporting storage-related functionality and alternative route selection. The overlay was used to test FTP transfers and the energy-expenditure was

automatically accounted for by the inherent WLAN physical layer energy model of ns-2. At the third, and final, stage of the solution development, the DTN agent was extended with functionality to support the UDP protocol, enabling accurate simulation of streaming applications. Furthermore, a new type of an HTTP application was developed allowing simulation of the browsing experience in the overlay.

Common Simulation Settings

In all development stages, the Base Station (BS) is connected with the Mobile Receiver (MR) on a 802.11 WLAN, which is simulated by the corresponding module available with ns-2. The WLAN uses a data rate of 11 Mbps and a basic rate of 1 Mbps. When TCP is used (FTP, HTTP) the packet size is 1460 (so that fragmentation at lower layer is limited) and the maximum window size 100 packets. For UDP traffic (CBR) the packet size is explicitly specified in each set of experiments. Unless otherwise noted, the rendezvous mechanism at the BS uses a TBO of 30 KB.

In order to determine the relationship between the selected TBO value and the bandwidth-delay product (BxD) of the wireless link we ran some preliminary, targeted simulation experiments that measure the propagation delay and the net throughput of the WLAN. The delay of the wireless connection between the BS and the MR was measured using a ping agent and was found to be approximately 2ms, while the net throughput for a TCP flow was measured to be 2.8 Mbps. Using these values we calculate:

- The nominal BxD based on the nominal 802.11 bandwidth (11 Mbps): 0.002 x 11 000 000 / 8 = 2.750 KB.
- The net BxD based on the net throughput of a TCP flow: 0.002 x 2 800 000 / 8 = 700 B.

The bandwidth of the wireless link is adequately higher while the delay is significantly lower than the corresponding values for the wired bottleneck link under all experiment settings. The excess capacity of the wireless link is utilized by the rendezvous mechanism in order to increase energy efficiency. According to these values, a TBO of 30 KB corresponds to 10x the nominal BxD and 40x the net TCP BxD.

Performance Metrics

The reported metrics for the simulation experiments mainly involve the delay and the energy expenditure of data transfers. In certain cases the actual amount of buffered data is also reported and compared against the TBO.

The measured delay depends on the type of the data transfer in each experiment and, generally, refers to the ADU of the application. For FTP connections the measured delay refers to the completion of the entire file transfer. In such cases, the buffering delay is usually negligible as the total file size is much larger than the buffering amount. For CBR transfers the reported delay reflects the time necessary for each UDP datagram to reach the destination. The size of a UDP datagram is usually much smaller than the buffering amount and so the delay may be quite substantial. Finally, for HTTP connections the delay refers to the

time necessary for each of the many small files normally contained in a web page to reach their destination. The size of the HTTP files is usually comparable to the buffering amount (much smaller than an FTP file, but larger than a CBR datagram) leading, again, to a substantial introduced delay.

The energy expenditure refers to the energy in joules (J) that is required by the networking subsystem (WNIC) of the MR alone for the entire duration of the experiment (i.e. one file transfer, and streaming or web browsing of certain duration). The energy expenditure calculations require special attention and so they are described separately in the following section. Subsequent sections contain detail descriptions of the three model development stages along with the corresponding experiment scenarios.

6.1.1. Energy Expenditure Calculations

The parameters necessary for the energy expenditure calculations are related to the power consumption specifications of the WNIC, which are the following: transmit power (*txPower*), receive power (*rxPower*), idle power (*idlePower*), sleep power (*sleepPower*), transition power (*transPower*), and transition time (*transTime*). The power figures for the main interface states are measured in watts (W) and are set as follows [151]: txPower = 1.400 W, rxPower = 0.950 W, idlePower = 0.805 W and sleepPower = 0.060 W. Based on measurements in [152] we can safely consider the energy expenditure in the transitive state to be the same as that of the idle state (i.e. transPower = idlePower = 0.805 W). In related literature ([153]) the transition time from idle to sleep and vice versa is found to be bounded by 20ms (in most cases it is measured as less than 2ms as in [31] and [154]), so we take the pessimistic approach by setting the transTime = 10ms (2 x transTime = 20ms). It is worth noting that, since the rendezvous mechanism operates at a higher network layer, it is not dependent on the 802.11 Power Saving Mode and, thus, it is not required to wake up and listen to the Traffic Indication Maps included in the periodic beacon frames [5]. This is a relaxation of our initial, overly conservative assumption in [148], where we considered that the mobile receiver must first listen to the next beacon frame before it is able to receive any buffered data.

The state transitions were monitored by adding code into the physical layer of the wireless interface node that traces the duration the WNIC spends in each state during simulation. The energy expenditure calculations were facilitated by a number of post-processing scripts and several interventions into the simulator. As described in the following subsections the calculations can be done both manually by special post-simulation processing scripts as well as during simulation, utilizing the ns-2 energy model.

Manual Energy Expenditure Calculations

The energy expenditure is manually calculated by post-processing scripts, analyzing the state transition records after the simulation has concluded. The energy expenditure for each of the transmit and receive intervals is calculated as the duration of the interval multiplied by the transmit and receive power respectively. Idle intervals can either remain idle intervals (no switch to sleep mode occurs) or they can be converted to sleep intervals. In order for an idle interval to be converted to a sleep interval the transition must be both feasible and meaningful. The transition is considered feasible only if the time it takes to fall

into the sleep state and then back to the active state (i.e. transTime) is less than the duration of the idle interval itself. The transition is considered meaningful if the energy required for switching back and forth to the sleep state, plus the energy spent during the sleep state is less than the energy expenditure if the WNIC had remained in an idle state for the whole interval duration. If idleDuration is the duration of the idle interval, the decision is based on the following general formula:

$$2 * transPower * transTime + (idleDuration - 2 * transTime) * sleepPower < idleDuration * idlePower$$

Based on our assumption that transPower is equal to idlePower (detailed in the beginning of the current section) and the fact that sleepPower is much lower than idlePower, all feasible transitions allowing at least an instance of sleep duration are meaningful.

For idle intervals that are not converted to sleep intervals the energy expenditure is calculated as the duration of the interval multiplied by the idle power. For sleep intervals the energy expenditure is calculated as:

$$2 * transPower * transTime + (idleDuration - 2 * transTime) * sleepPower$$

The above calculations can be used in order to find the energy-saving potential of a certain buffering scheme when the rendezvous mechanism is not employed and the WNIC always remains into the active state. The potential energy savings correspond to the energy conservation that would result if there was in place a mechanism that would switch the WNIC into the sleep state during idle intervals. These calculations provide the ideal energy expenditure for a given state transition trace as they presume some out-of-band notification of the WNIC to switch to the active state as soon it is necessary to receive a new frame.

The manual energy expenditure calculations can also be applied to a case when the rendezvous mechanism is used by adding the notion of idle intervals that are candidate for conversion. Idle intervals candidate for conversion are those that occur in the meantime between the completion of a buffer flushing and the commencement of the next buffer flushing, and are identified by marking the end of each buffer transfer in the log files. The end of each buffer transfer is marked by having the TCP agent report upon the reception of the last acknowledgment on the BS side. As in the energy-saving potential calculation, the candidate idle intervals are checked for a feasible and meaningful conversion and, if both apply, are treated accordingly.

Automatic Energy Expenditure Calculations

The functionality for on-demand switching of the wireless interface to the sleep/active state and monitoring the energy expenditure is inherently available to the wireless physical layer provided in ns-2. We observed, however, that under certain circumstances the wireless interface would ignore the sleep command and that the energy expenditure calculations did not accurately account for the transition time. Thus, we added code to the wireless physical layer implementation that amends these issues and also

records the exact state transitions of the wireless interface, providing extra visibility into the energy-saving mechanism.

The energy model in the ns-2 wireless physical layer relies on the following input parameters for monitoring energy consumption: *txPower*, *rxPower*, *idlePower*, *sleepPower*, *transPower*, *transTime*, and an initial energy value (*initEnergy*). The available energy level at the wireless physical layer is set to the initial energy value at simulation start and it is appropriately decreased, whenever the wireless interface changes states during the remaining simulation time. When the wireless interface commences transmitting or receiving data, the available energy is decreased by the amount consumed during the previous idle interval (i.e. idle time x idlePower). When the wireless interface finishes transmitting or receiving data, the available energy is decreased by the amount of the transmission or reception (i.e. transmission duration x txPower or reception duration x rxPower). When switching from sleep to idle or vice versa, the energy level is updated based on the previous interval duration (sleep or idle), the sleepPower or idlePower for a sleep or idle interval respectively, the transTime and the transPower. Care must be taken so that between a switch from active to sleep and a subsequent switch from sleep to active the wireless interface will be allowed at least 2 x transTime otherwise the calculation will not accurately reflect the energy consumption. This provision takes place at the DTN layer of the mobile node, ensuring that the next rendezvous is at least 2 x transTime into the future before switching the wireless interface to the sleep state.

6.1.2. Stage 1 (Proof-of-Concept)

In this first proof-of-concept scenario the bundle protocol behavior is emulated by introducing a proxy application at the BS. The proxy application is connected to one input agent (TCP or UDP) that receives data from the sender on the wired part of the network, and one output agent that transmits data to the receiver over the last hop wireless link. The proxy application implements the rendezvous mechanism by buffering data received on the incoming end and flushing it out to the outgoing end at the next rendezvous time. The rendezvous times are not communicated to the MR, but rather recorded so that energy consumption can be calculated during post-processing of the simulation outcome. This setup emulates the functionality of a DTN overlay on an IP network, when TCP or UDP are used as convergence layers. The proxy application was implemented as a subclass of the Application class in ns-2, sitting on top of the transport agents on the network stack.

Figure 6-1 depicts the network topology that was used as part of this proof-of-concept scenario. The BS node is part of both the wired and the wireless networks and the MR participates in the 802.11 LAN hosted by the BS. Rel-IP means that the node relays IP traffic, while CT-Src and CT-Dst are the source and destination of Cross-Traffic (CT) respectively. The lighting icon between the BS and the MR denotes that the BS – MR is a wireless link. The bandwidth of the backbone link (Rel-IP - BS) is modified across simulation runs so that the desired congestion is introduced into the network. The rest of the links are kept at the same delay and bandwidth values for all the experiments. The queuing discipline used for all links is Stochastic Fairness Queuing (SFQ) [155].

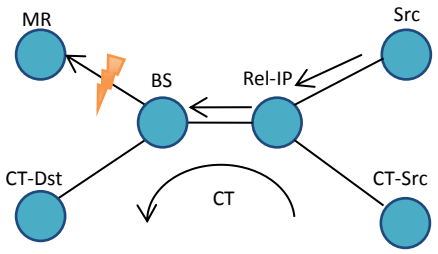

Figure 6-1: Proof-of-concept topology.

The bandwidth and delay values for the topology links are as follows:

- Src – Rel-IP: 2Mbps bandwidth, 100ms delay
- CT-Src – Rel-IP: 3Mbps bandwidth, 100ms delay
- Rel-IP – BS: varying bandwidth depending on the desired network characteristic, 300ms delay
- BS – CT-Dst: 3Mbps bandwidth, 100ms delay

The experiments involve a main data transfer, for which performance is measured, as well as a competing flow that simulates network traffic. The data transfer studied in this experiment follows the Src→Rel-IP→BS→MR route, while the competing flow follows the CT-Src→Rel-IP→BS→CT-Dst route. The main data transfer can be either end-to-end (E2E) or it can be split at the BS by the splitting application implementing the rendezvous mechanism (Split). In the E2E experiments there is an end-to-end connection between Src and MR with the intermediate nodes being mere IP forwarders of the upper layer traffic. In the Split case there is a splitting application at the BS, where the incoming connection from the Src terminates and another connection to the MR initiates; the Rel-IP node in this case is the only pure IP node, while the rest of the nodes are also transport connection endpoints. The splitting application acts as a transport layer mediator and, as previously mentioned, plays the role of the DTN Bundle protocol.

The energy expenditure in this set of experiments is calculated manually during post processing, as described in the "Manual Energy Expenditure Calculations" subsection of section 6.1.1. In certain experiment sets we also report on the potential energy expenditure and label the corresponding lines as P-E2E and P-Split. The potential energy expenditure calculations are reported in order to establish feasibility of the proposed solution. These calculations are applied both to a regular data transfer (P-E2E) with no buffering and to a data transfer with constant buffering where the rendezvous mechanism is not applied (P-Split). The P-Split compared to the P-E2E shows the additional energy-saving potential introduced by buffering at the BS, whereas the P-Split vs. the Split experiments demonstrate the ability of the rendezvous mechanism to harvest the available energy-saving potential.

Experiments are carried out both for an FTP connection transferring 10 MB of data as well as CBR traffic that flows for a specified amount of time. When present, the competing flow sends a 1000 byte packet every 3 ms. In certain experiments we are introducing a packet error rate on the WLAN in order to evaluate the performance of the proposed solution when a random reception error rather than a congestion-

related error is present on the network. The error follows a uniform distribution and is applied on the outgoing interface of both the BS as well as the WNIC of the mobile node.

6.1.3. Stage 2 (Full DTN Overlay)

In this set of experiments a full DTN overlay is formed by deploying the DTN agent in a variety of scenarios. The focus in this set of experiments was TCP connections and, more specifically, FTP transfers, so the underlying transport protocol in all experiments is TCP. When applicable, the simulation results of the experiments involving a DTN overlay are compared to those of an End-to-End (E2E) TCP connection, where the agent is not employed.

The rendezvous mechanism between the BS and the MR is implemented within the DTN agent on both ends of the wireless connection. When buffered data are flushed, the next rendezvous time is communicated to the receiver via the Bundle Protocol header and the mobile receiver switches its wireless interface to the sleep state and back to the active state appropriately. This way the energy consumption can be tracked by the improved built-in energy model in the wireless physical layer implemented in ns-2, as described in the "Automatic Energy Expenditure Calculations" subsection of section 6.1.1.

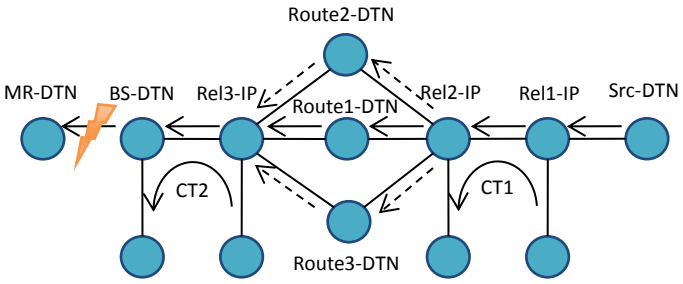

Figure 6-2: Full DTN overlay topology.

Figure 6-2 depicts the network topology that was employed to accommodate the simulation experiments. The presented topology was developed in order to: a) evaluate the energy expenditure and effective throughput achieved when the rendezvous mechanism is enabled, b) demonstrate the ability of the DTN agent to simulate complex networking scenarios including multiple routes, possibly taking storage availability into account.

Nodes whose name suffix is DTN are used as nodes in the DTN overlay in at least some of the conducted experiments. More specifically, Src-DTN, BS-DTN and MR-DTN host DTN agents under all experimental settings, whereas Route-DTN (1-3) host DTN agents only under certain experimental settings. The Src-DTN is the source node, where the main data transfer is initiated, while BS-DTN is the base station and MR-DTN the mobile receiver where the main data transfer is destined to. The lighting icon between the base station and the mobile receiver denotes that the last hop is a wireless link.

Relay nodes Rel-IP (1-3) are pure IP nodes and they were inserted into the topology in order to meet certain technical requirements. Firstly, the three IP relay nodes allow the introduction of Cross-Traffic (CT) flows CT1 and CT2. In ns-2 the router queue is implemented within the link object, and therefore, introducing cross-traffic involves channeling the contending flow through the same link as the main flow. Additionally, Rel3-IP plays the role of a wired gateway, required by the ns-2 hierarchical routing functionality that interconnects a WLAN with the wired network. Otherwise, if the BS were directly connected to Route-DTN (1-3) returning packets would be routed only through one of the three nodes creating unnecessarily long routes (e.g. packets travelling from BS to Route2-DTN would follow the Route1-DTN→Rel2-IP→Route2-DTN paths).

The bandwidth and delay values in the simulated topology were selected in order to form three network segments separated by the two bottleneck links between Rel1-IP – Rel2-IP and Rel3-IP – BS-DTN. This setting allowed us to evaluate the effect of deploying the DTN overlay so as to create an island among network segments connected through congested links. All wired links are assigned low bandwidth and long delays in order to simulate multi-hop Internet routes that may traverse several nodes along the network path. Bandwidth and delay values for the topology links are as follows:

a) The main links Src-DTN – Rel1-IP, Rel2-IP – Route1-DTN, Route1-DTN – Rel3-IP have bandwidth and delay values of 3Mbps and 100ms respectively.
b) The bottleneck links Rel1-IP – Rel2-IP and Rel3-IP – BS-DTN have a bandwidth that ranges from 1Mbps to 4Mbps depending on the simulation scenario and a delay of 100ms.
c) In the load balancing scenarios where the alternative routes are also active (dashed arrows in Figure 6-2) the 6 routing links from Rel2-IP to Rel3-IP all have a bandwidth of 1Mbps and a delay of 100ms.

All queues for the above links implement fair queuing. The experiments are carried out for an FTP connection transferring 10 MB of data.

The main data transfer follows the horizontal arrows traversing the topology through the following path Src-DTN→Rel1-IP→Rel2-IP→Route1-DTN→Rel3-IP→BS-DTN→MR-DTN. Under certain scenarios, the agent may load-balance traffic through the Route2-DTN and Route3-DTN nodes following the dashed arrows. Custody report bundles are moving in the opposite direction of the displayed arrows. Cross-traffic is introduced in the form of two TCP transfers CT1 and CT2, which flow in the direction of the elliptical arrows and create two bottleneck links before and after the intermediate DTN nodes.

The experiments at this development stage were conducted as part of four experimental scenarios, each designed to highlight certain aspects of DTN combined with our proposed energy-saving, rendezvous mechanism. The four scenarios are: Single-bundle, varied bandwidth (the bottleneck bandwidth is varied on a single-bundle ADU transfer), Single-bundle, varied Target Buffer Occupancy (the TBO is varied on a single-bundle ADU transfer), Multi-bundle, varied bandwidth (the bottleneck bandwidth is varied on a transfer consisting of multiple fragments of the original ADU), Multi-bundle, varied DTN storage space (the available DTN storage space is varied on a multi-bundle transfer). Through the single-bundle scenarios

we evaluate the energy-efficiency and throughput of the rendezvous scheme. The multi-bundle scenarios, on the other hand, demonstrate the DTN agent functionality for load balancing traffic based on storage space availability and emphasize the capability of DTN to improve network performance by uniformly distributing data storage over alternative paths.

6.1.4. Stage 3 (Simple DTN Overlay)

In this final set of experiments the DTN agent is used in order to create a simple overlay similar to that of the proof-of-concept scenario. Here, the DTN agent also utilizes UDP as the underlying transport protocol allowing for the experimentation with streaming applications as well. Additionally, a new type of an HTTP application facilitates simulating web traffic. The goal in this set of experiments is to focus on the relationship between the energy saving and the delay for each type of network traffic against a variety of competing traffic types, which also traverse the wireless network. Through these simulations we would like to assess the energy conservation vs. data delivery tradeoff from a user experience perspective across the three main types of applications: large file transfers (FTP), media streaming (CBR), and web browsing (HTTP).

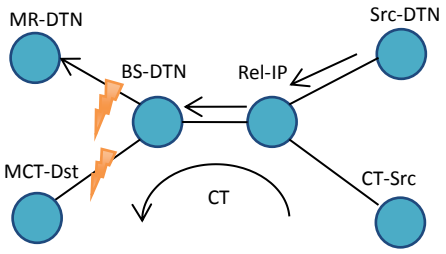

Figure 6-3: Simple DTN overlay.

The topology used in this set of experiments is depicted in Figure 6-3. Nodes with the DTN suffix host a DTN agent instance, Rel-IP relays IP traffic and CT-Src and MCT-Dst are the source and destination of the cross-traffic flow respectively. Deviating from the proof-of-concept topology, the destination for the cross-traffic flow (MCT-Dst) is located in the same WLAN as the MR and the BS introduces competing traffic in the wireless network (i.e. lighting icon on the BS-DTN – MCT-Dst link as well). The bandwidth and delay values for the topology links are kept constant for all the experiments at the following values:

- Src – Rel-IP: 1.5Mbps bandwidth, 100ms delay
- CT-Src – Rel-IP: 1Mbps bandwidth, 100ms delay
- Rel-IP – BS: 1.5Mbps bandwidth, 100ms delay

Similarly to the previous scenarios, the main data transfer follows the Src-DTN→Rel-IP→BS-DTN→MR-DTN and the competing flow the CT-Src→Rel-IP→BS-DTN→MCT-Dst route. The main transfer may be either E2E from the Src-DTN to the MR-DTN, or it may be intercepted at the DTN agent of the BS so that the rendezvous mechanism can be employed. Compared to the proof-of-concept scenario

described in section 6.1.2, a fixed, intermediate bottleneck bandwidth value was selected (1.5 Mbps), simulating conditions of a mildly congested network. The energy consumption in this set of experiments is tracked by the built-in energy model of ns-2 with the improvements described in the "Automatic Energy Expenditure Calculations" subsection of section 6.1.1.

The experiments include main data transfers of FTP, CBR and HTTP. HTTP is simulated using a simple application model that assumes no request pipelining [156] and a constant small file size. These small files are transferred one after the other; when one file is received the next file starts immediately downloading. HTTP is used to evaluate possible deterioration in the responsiveness of web browsing due to the additional delay introduced by buffering at the BS. The traffic characteristics for each of the above transfer types are:

a) FTP: The file size of FTP is 10 MB. A competing flow is either CBR sending 500 B every 5 ms, or TCP transferring a large file.
b) CBR: The flow sends 500 B every 5 ms (800 Kbps) for 1 minute. A competing flow is either CBR sending 500 B every 3.5 ms, or TCP transferring a large file.
c) HTTP: The flow sends files of size 20 KB for 2 minutes and 40 seconds. A new file is sent as soon as the previous file has finished downloading. A competing flow is either CBR sending 500 B every 3.5 ms, or TCP transferring a large file.

The simulated time for the CBR and HTTP transfer types was set such that sufficient number of units (datagrams or small files respectively) would be transferred in order for the results to be statistically significant. Due to the conversational nature of HTTP, the number of small files transferred in a certain time period is much smaller than the number of datagrams of CBR traffic, and, thus, the duration of the HTTP simulations was set to be significantly longer than that of CBR.

6.2. Decentralized Space-Data Dissemination with DTN

In order to simulate the communication patterns and data delivery models described in section 4.4, we employed a combination of STK, the physical simulator modeling the satellite orbits, and ns-2, the network simulator modeling the end-to-end network from the satellite application to the end-users. The STK simulator calculates the physical characteristics of the space contacts, including the start and end times, bandwidth, bit error-rate and delay. Each contact includes the identification numbers of the participating satellite and ground station, allowing association with the corresponding entities in ns-2. The contacts are translated into ns-2 link up and down events, which are then imported into ns-2 and incorporated in the end-to-end topology [32].

The STK simulation takes into consideration the physical aspects of the reference missions, including the orbit geometry of the satellite, geographical locations and coverage of ground stations, as well as the start epoch and duration for each scenario. Link budgets for each contact between satellites and ground stations are calculated using detailed models of the satellite transmitter and ground station receiver, already

available in STK. The STK simulation experiments are based on the direct point-to-point, relay point-to-point and low-cost broadcast templates, which correspond to the space link models described in section 4.3.

In ns-2 the entire topology is modeled as an IP network using UDP in space and TCP on the ground. The central entity of the network simulation is an extended version of the DTN agent that we have developed in ns-2 and is based on the DTN simulation model described in [30]. The DTN agent is deployed as an overlay on the relevant nodes (i.e. satellites, ground stations, mission control center, end-users, principal investigator), while the rest of the topology consists of pure IP nodes, relaying upper layer traffic. The STK output is used in order to create links with the corresponding characteristics between satellite and ground station nodes and also to set these links to the up/down state accordingly during simulation.

The simulator implements six types of network nodes as described in section 4.4.1: Satellite, Ground Stations, Mission Control Centers, End Users, Principal Investigators, and Uplink Ground Stations. All these types are simulated by multiple instances of the DTN agent, appropriately configured according to the desired simulated node type. The source DTN agent exposes an interface that accepts the number of bytes to be transmitted and, optionally, the data ToS. Through this interface the agent creates ADUs, which are then fragmented into one or more bundles, based on the maximum bundle size parameter. The bundles are inserted into a sending buffer of a user-defined capacity. Newly arrived bundles may cause the eviction of old bundles in the case of buffer space shortage.

6.2.1. Performance Metrics

The results of the simulation experiments are reported with the help of the following performance metrics:

- Delivery Ratio: The percentage of the created ADUs successfully delivered to all end-users:

$$\frac{ReceivedADUs}{CreatedADUs * EU\_Number}$$

  In case a single EU is present the delivery ratio corresponds to the ADUs delivered on the ground, since data is unlikely to be lost during ground distribution when the network is uncongested. Otherwise, ADUs may be received by a subset of the interested EUs, which is reflected in the delivery ratio value. Partially received ADUs for (i.e. some ADU bundles are missing) are completely disregarded.
- Delivery Latency: The average data latency between the production time of an ADU on the satellite and the ADU delivery to the EUs. In case of a single EU the delivery latency essentially corresponds to the time necessary for the ADU to be received on the ground (ground distribution is generally short in such a case). In case of multiple EUs the delivery latency may include ground distribution delays due to congestion.
- Data Volume: The total amount of actual data, not including the headers, that was delivered successfully to the EUs. ADUs received at multiple EUs contribute multiple times their size.

6.2.2. Physical Simulations

The experiments conducted in the STK simulator calculate the contact opportunities in the direct point-to-point, relay point-to-point and low-cost broadcast space link models. First, a selection of transmission systems based on UHF-, S- and X-Bands were compared so that the most efficient system for the low-cost broadcast model could be determined, in terms of contact opportunities duration, communication link quality and power consumption. The initial results indicated that S-band is the most appropriate solution, since it allows for longer uninterrupted contacts, increased total contact time and decreased Bit Error Rate (BER) at system level. The calculated BER corresponds to the effective error after Forward Error Correction has been applied on the physical link.

The physical parameters finally selected for each space link model are:

- Direct point-to-point: 1 Satellite, X-Band at 520 Mbps, BER 10^{-7}, 3 ground stations.
- Relay point-to-point: 1 Satellite, X-Band, inter-satellite link at 100 Mbps, BER 10^{-7}, 3 GEO satellites.
- Low-cost broadcast (S-Band 3 Mbps, BER 10^{-7}):
    - 52, 100 and 140 ground stations with 1 satellite.
    - 6 and 46 satellites with 52 ground stations (QB50-like mission models).

In all cases, satellites are orbiting in sun-synchronous orbit at 700 Km, with an inclination of 96.7 degrees. Sun synchronous orbit, the most common orbit for imaging applications, is selected here because it provides better coverage of the Earth surface, resulting in a higher number of contact opportunities. The ground stations are spread across the globe, including areas and places such as Europe, Canada, overseas territories, world-wide embassies and ESA-operated ground station locations. Finally, the duration of the simulations is set at 10 days, which is a full period for the satellite orbit (i.e. the satellite passes over the same locations). The link-up and link-down events for each of these contacts are set at the time when the calculated bit error rate exceeds a configurable threshold parameter, typically set at 10^{-7} in our simulations.

6.2.3. Satellite Operation

Data production is simulated using a custom application module that is configured to produce an ADU of a certain size every some time period. The simulation script accepts the desired daily data production rate as well as the ADU size, and calculates the ADU production period, which is then propagated to the application module. Each application tags the ADUs it creates with a ToS number. An arbitrary number of application modules can be attached to the same agent, enabling the simulation of complex data generation patterns. ADUs received at the DTN agent are fragmented into the appropriate number of bundles and entered into the transmission buffer.

In the general case, satellites continuously broadcast data to any ground station that may be receiving. Satellite-deployed DTN agents use UDP as the underlying protocol over the space link to ground stations. The packet/frame size overhead can be configured based on the total TM overhead, independently of the bundle header imposed by the higher layer protocol (i.e. CFDP, DTN, etc).

The DTN agent at the satellite implements the randomized retransmission pattern described in section 4.5.2. Once all new bundles have been transmitted for the first time a random bundle is selected from the buffer. Transmission resumes at the first bundle of the ADU that the randomly selected bundle belongs to. At the reception of an acknowledgment the corresponding bundle is removed from the buffer. Transmission at the satellite agent may freeze by some external trigger, while normally accepting new data, facilitating the simulation of the suspend/resume mechanism.

Data Unit Sizing

Ns-2 simulates network protocols at the packet level and, therefore, the overall computational complexity is determined by the total number of processed bundles during simulation. Due to the high level of simulation detail, ns-2 is normally used for small-scale experiments. In order to adapt it for large-scale simulations (i.e. the simulated duration of 10 days and GBs of data), we tried to lower complexity while maintaining fidelity of the results. In this spirit, we experimented with larger bundles, reducing the number of bundles for a certain level of data production rate. However, increasing the bundle size does not allow us to use the Bit Error Rate (BER) that is calculated by STK (i.e. for large bundle sizes all bundles would be corrupted). In order to overcome this problem we calculated the Packet Error Rate (PER) that corresponds to the given BER for a realistic nominal bundle size. The calculated PER was then applied to the larger bundles of the simulation, yielding analogous error effect.

The BER was converted to PER using the well-known conversion formula:

$$PER = 1 - (1 - BER)^{8*PacketSize} \quad (6\text{-}1)$$

PacketSize was set to the value of 65,000 bytes (approximately 64 KB) as this is the maximum size for a TM encapsulation packet.

ADU/bundle Size	Error Type	Production Period	Simulation time	Delivery Ratio
65000	BER	0.001	10	94.34
65000	PER	0.001	10	94.34
650000	PER	0.01	100	94.69
6500000	PER	0.1	1000	94.72
65000000	PER	1	10000	94.72
100000000	PER	1.53	15300	94.19

Table 6-1: Effect of bundle size and type-of-error on the delivery ratio.

The validity of the conversion approach was confirmed through a series of short comparative experiments. Using an always up, high-speed link such as the one from the direct point-to-point configuration (i.e. 520Mbps), we experimented with data production rates equal to the bandwidth of the transmission link and adjusted simulation time so that 1000 bundles would be produced. Also, we let the ADU size equal the bundle size so that the simulator would report delivery at the bundle level. Our pivot case was using a BER value of 10^{-7} for a bundle size of 65000. The same configuration was subsequently

simulated using the calculated PER for the 65000 byte (i.e. 5.07%) packet size and, reasonably, the results were identical. The calculated PER was then applied to the simulations with larger bundles (up to a value of 100MB) and, indeed, delivery ratio values were practically equal for all cases. Table 6-1 summarizes the experimental results.

The aforementioned conversion mechanism allows us to model the transmission of large data amounts and long time periods keeping the simulation execution time within reasonable bounds, while making sure that the space link BER is being correctly accounted for. Based on the previous analysis, the sizes of the ADU and the bundle are set at 10^9, 10^8 bytes (approximately 1 GB, 100 MB) for the point-to-point models and 10^7, 10^6 bytes (approximately 10 MB, 1 MB) for the low-cost models respectively. In all cases an ADU consists of 10 bundles.

6.2.4. Ground Network

Bundles received at a ground station are forwarded to the appropriate nodes of the DTN overlay based on route calculations taken place prior to simulation start. Route calculation involves creating one overlay multicast distribution tree per ground station, per ToS. Pruning is employed during simulation in order to avoid duplicate data transmission.

The DTN agent supports optional uplink functionality by generating acknowledgments either directly at the receiving GS (in the point-to-point space-link models) or at a PI subscribed to all ToS (in the low-cost broadcast model), and routing these acknowledgments to a designated UGS. In the low-cost broadcast model, acknowledgments are stored at the uplink GS until the next available contact, when they are forwarded to the satellite.

The ground segment may include a Mission Control Center (MCC) entity, in which case, routing is done in two tiers. In the first tier routes are setup between the MCC and each GS for all ToS and, in the second tier, routes are setup from the EUs to the MCC. The EUs connected to the overlay may subscribe for receiving data belonging to one or more ToS and also act as routers in the corresponding application-layer multicast trees. ToS subscription is an important feature of our proposed scheme, since it allows for categorizing space-data. However, the experiments presented in our work employ a single ToS, which all participating EUs are subscribed to.

Topology

The ground topology has been created using the gt-itm tool [157], shipped as part of the ns-2 distribution. The tool distributes all nodes onto a 2D grid and then forms communication links between nodes whose distance is less than some user-defined threshold. The length of the edge connecting two nodes determines the propagation delay for the link. Changing the user-defined parameters allows the user to vary the network density and create sparse or fully interconnected mesh networks.

Gt-itm provides a set of tools that analyze the resulting graphs and report on node degree average values and distribution, topology depth, average link size etc. We have created a script that converts the sgb output

of gt-itm (Stanford GraphBase format defined in [158]) into ns-2 topology generation instructions, which sets the link delay to a scaled value of the edge length by a user-defined ratio. The link bandwidth is globally set to a user-defined value for all ground links.

Degree	Number of Nodes
0	0
1	0
2	2
3	5
4	9
5	18
6	26
7	22
8	27
9	27
10	20
11	14
12	12
13	10
14	5
15	2
16	1

Table 6-2: Node degree distribution

The topology is created according to the original Waxman model [159] with alpha and beta parameters of 0.12 and 0.3 respectively, and created 200 nodes on a square plane of edge size 100 units. The resulting graph consists of a single biconnected component with an average degree of 8.29 (average number of links per node), a diameter of 5 hops (maximum hop count between any two nodes) and an average depth of 4.055 (average hop count between all pairs of nodes). The average length for the graph edges was 37.1 units. Table 6-2 contains the node degree distribution and Table 6-3 contains the depth distribution (maximum number of hops to the most remote node).

Depth	Number of Nodes
3	1
4	187
5	12

Table 6-3: Topology depth distribution

The created graph is converted into an ns-2 topology file where the delay of each link is set to the link length scaled by 0.1 ms, resulting in an average link delay of 3.71 ms. The link bandwidth for all ground links is set to the same value, which is adequately high so that ground links are not the bottleneck (1Gbps and 18 Mbps for the point-to-point and the low-cost broadcast cases respectively).

Depending on the configuration, only a subset of the 200 nodes is used for hosting DTN agents, while the remaining nodes function as pure IP nodes. A single topology can be used for multiple experiments as

long as the total number of ground elements is less than or equal to the number of nodes. The assignment of the DTN agents to the topology nodes follows the order of GS, MCC and then EU (UGSs and PIs are special types of GSs and EUs respectively). The simulation script places one DTN agent on each node starting from node 1 up to the total number of simulated elements. Due to their special characteristics satellite nodes are created separately resulting in a higher number of overall simulated nodes. Experiments that focus on the space segment performance employ only one EU, so that simulation time is not unnecessarily prolonged. Multiple EUs are used only in the scenario where the ground segment performance is studied.

6.3. Storage-Constrained Routing in Opportunistic DTNs

The simulation experiments were conducted in the ns-2 network simulation platform and were based on a customized rendition of our DTN simulation model. The routing problem is simulated as a game and consists of a number of interconnected instances of the DTN agent, deployed on a carefully selected network topology. In order to support the proposed routing mechanisms, the agent was extended with additional functionality necessary for keeping track of bundle acceptance history by peer nodes. The DTN agent instances correspond to the individual players and their goal is to maximize the throughput they receive from the system.

6.3.1. Simulation Setup

The simulation topology can be seen in the diagram of Figure 6-4. A DTN agent is deployed on each of the topology nodes; however the game players are only the 10 nodes in the WiredTop and the WiredBottom levels. The bundles that are created at the Bottom row are destined to the WirelessTop receiver connected through the BS Top, Base Station node. Conversely, bundles created at the WiredTop nodes are destined to the WirelessBottom wireless receiver connected through the BS Bottom Base Station. At bundle reception the agent picks a downstream peer to forward the bundle to in a round robin fashion. Nodes at the top row need to relay bundles on behalf of the nodes on the bottom row and vice versa. The nodes benefit from servicing their own applications by maximizing the number of locally created bundles that reach their final destination.

Initial testing was conducted in order to conclude on the topology parameters listed in the previous section. Our original plan of enabling bundle retransmissions was abandoned due to the observation that, under these circumstances, the system was very intolerant to buffer size modifications. More specifically, the overall delivery ratio for a 66000 storage space (11 bundles) was 0.99, whereas for a 60000 storage space (10 bundles) it was only 0.05. In the latter case, the buffers of all 10 nodes are fully occupied early on in the simulation and the network enters a deadlock where no bundles could be accepted in either direction. When retransmissions are disabled the nodes immediately drop bundles for which custody is rejected, allowing the network to continue servicing bundles instead of essentially halting service.

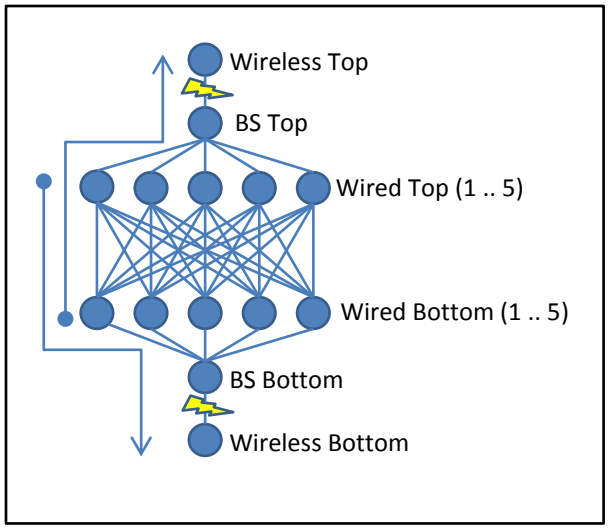

Figure 6-4: Simulation topology.

During initial testing we also observed that in cases where the wireless link between the BS and the wireless receiver was the bottleneck link, delivery ratio remained high but delivery times were constantly increasing for the total length of the simulation. Under such circumstances data kept accumulating at the 2 BS nodes (since the BSs have unlimited storage space no bundles were dropped there) and delivery times kept increasing throughout the entire simulation. Simulation parameters were selected so that the wireless connection was not the bottleneck link of the data transfers.

The experimental parameters are as follows:

- WiFi: 54 Mbps data rate, 24 Mbps basic rate.
- Wired to BS: 10 Mbps, 100 ms, Stochastic Fair Queuing.
- Wired Top to Bottom: 2 Mbps, 100 ms, Stochastic Fair Queuing.
- Bundle Size: 6000 bytes.
- Storage Space (Top/Bottom): 48000 (8 bundles).
- Storage Space (BS): unlimited.
- Wired sending rate: 1 bundle every 150 ms.
- Simulation duration: 160 seconds.
- Total of 1066 bundles produced.
- Retransmissions: Disabled.

6.3.2. Performance Metrics

In evaluating the experimental results we reported upon the following metrics:

- Per-node average delivery time: The average bundle delivery time for a single node.
- Overall average delivery time: The average bundle delivery time for all nodes.

- Per-node delivery ratio: The bundle delivery ratio for a single node throughout the entire simulation.
- Overall delivery ratio: The bundle delivery ration for all nodes throughout the entire simulation.
- Delivery fairness index = $1 - \frac{\sum_{i=1}^{n} |Ti - Avg|}{2(n-1)Avg}$ [160].

  The above delivery fairness index exhibits the following characteristics:
  - Equals 1 when all nodes experience the same delivery ratio.
  - Equals 0 when one node experiences non-zero delivery ratio and all others experience zero delivery ratio.
  - Equals $\frac{k-1}{n-1}$ when k nodes experience equal delivery ratio and the rest n-k experience zero delivery ratio.

6.3.3. Test Plan

The test plan followed in this series of experiments consists of a number of stages involving various combinations of the bundle acceptance strategies, as described in section 5.3. First we report on the findings for the impartial case, where a bundle is accepted if enough space is available without employing any particular acceptance mechanism. In this case, all nodes are well-behaved and service other nodes the same way they would service their local requests. This scenario serves as a benchmark for the maximum performance and fairness the network can provide, and assists in evaluating the rest of the simulation results.

In the second set of experiments a partially selfish, non-deterministic acceptance (favors local bundle servicing) algorithm is employed first for all nodes, then for half of the nodes and, finally, only for a single node in the topology. When some of the nodes accept bundles non-deterministically (behave selfishly), the rest follow the deterministic acceptance strategy (are impartial). This set of experiments aims at shedding light at the effects of selfish node behavior when the peers are impartial, and the degree to which selfish nodes may exploit their peers' good will.

In the third stage of the experiments we report on the results achieved by the application of the history-based deterministic and strategy 1 (non-deterministic) mechanisms that take the BAR into account. For reasons that will be expanded in section 8.3, the outcome of the experiments in this category suggested that the history-based mechanisms deterministic and strategy 1 were inappropriate for use in the current network setting and, thus, rejected.

In the final stage of our experimentation we report on experiments conducted with the strategy 2 (non-deterministic) mechanism when employed by all nodes and when one node is totally selfish (only forwards its own bundles). This mechanism was found to perform satisfactorily and so it was accepted as the most appropriate for the experimental network setup.

7. Simulating Networking Protocols

In this chapter we present our work with respect to the simulation of networking protocols. Accurate and detailed simulation models are powerful tools in the hands of networking protocols designers. However, network simulators mainly provide implementations for standardized protocols "off-the-shelf" and, usually, require extensive modifications in order to support new functionality. For the development of our models, we adopted the ns-2 network simulator. Ns-2 has a well-established record for simulating traditional Internet networks, which constitute the main target deployment environment for our proposed solutions (see sections 1.2 and 2.5.1). As the ns-2 simulator did not provide support for DTN, we decided to create our own implementation, which we made available to the community here [30]. Creating an in-house simulation model gave us full control over the design details of our proposed solutions and allowed for accurate simulations, which highlighted implementation details also present in real protocol deployments.

Our simulation models are implemented as a set of C++ classes along with some auxiliary TCL code in ns-2. Namely, we added 11 new files containing 10 new C++ classes (over 3000 lines of code) and additional higher-lever functionality in TCL (500 lines of code). During the process we also applied a number of additions and fixes to the core of the ns-2 code. The next section includes the description of the basic DTN model; subsequent sections describe the extensions implemented for the simulation of various specialized solutions, namely, the energy-saving rendezvous mechanism, 802.11 bandwidth estimation, reliable space-date broadcasting, space-data P2P multicasting, opportunistic storage-based routing and cryptographic operations.

7.1. Basic DTN Model

The simulation model was architected with respect to the encapsulation software design principle, so that classes are responsible for manipulating their own data, hiding their inner workings from outsiders. The overall architecture is presented in the following section, while subsequent sections describe the particular functionality provided by each class in the model. The final section (7.1.6) includes the main TCL procedures necessary for the DTN model operation.

7.1.1. Overall Architecture

The overall structure of the C++ classes of the model, including just a few indicative members for each class, is presented in the simplified UML diagram of Figure 7-1. The DTN agent at the top level includes a *BundleManagerDTN* and a *ConnectionManager* for storing and handling bundles and connections respectively. The bundle manager contains multiple bundles, each consisting of a DTN header (*struct hdr_dtn*) along with some additional fields for local use. A connection manager contains a number of upstream and downstream *DtnConnection* objects. Each DtnConnection contains a *TcpDtnAgent* for sending and a *TcpDtnSink* for receiving data (or, alternatively, two *UdpDtnAgent* objects in case UDP is

used). Finally, both *TcpDtnAgent* and *TcpDtnSink* include a *BundleManagerTCP*, which is a simplified version of BundleManagerDTN.

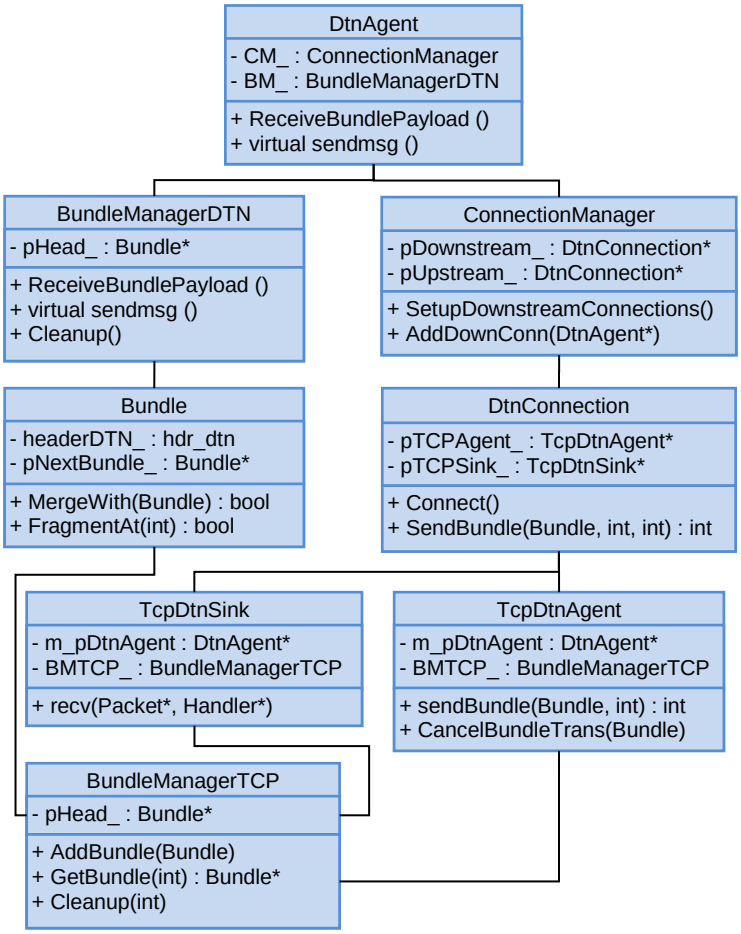

Figure 7-1: DTN Agent Class Diagram.

The DTN header contains the following major fields: *SequenceNumber*, *SourceAddress*, *DestinationAddress*, *Timestamp*, *FragmentOffset*, *BundleSize*, *ADUSize*, *IsStatusReport*, *CustodyAccepted*, *CustodyRejected*, *LeadingBytes*, *SegmentOffset*, and *Payload*. With respect to the Bundle Protocol specifications [15] a few header fields are absent from the simulation model, namely: *Report-to*, *Custodian*, *Sequence Number*, and *Lifetime*. These extra fields were not deemed necessary for the development of our proposed solutions, but they may be added in a straightforward manner in order to meet specialized needs for future simulations.

Status reporting is achieved through DTN header fields, contrary to the Bundle Protocol specifications, where status reports are standard ADUs, included in the bundle payload. It is important here to highlight that all packets in an ns-2 simulation, irrespectively of the network layer they were created on, contain the headers of all the network protocols in the stack. Consequently, a TCP packet carrying part of a bundle in

its payload will contain, among others, both the TCP as well as the DTN headers, regardless of what part of the bundle is being carried. This idiosyncrasy allows us to tag TCP packets with some DTN-related information, but also compels us to manually make provisions for bundle protocol header and payload size calculations.

The architectural description in the upcoming sections follows a bottom-up order. Classes Bundle, BundleManagerTCP and BundleManagerDTN are described in the first subsection; classes TcpDtn, TcpDtnSink and UdpDtnAgent are described in the second subsection; classes DtnConnection and ConnectionManager are described in the third subsection and, finally, class DtnAgent, is described in the fourth subsection.

7.1.2. Bundle and Bundle Managers

The Bundle class contains a hdr_dtn structure, as described in the previous section, as well as a number of extra members required for local storage and processing. Local fields include the following: *OriginalOffset*, *OriginalSize*, *TCPSequenceNo*, *ReceivedBytes*, *RetransmitTime*, *Forwarded*, *Acked*, *Suspended*, *FromConnection*, and *ToConnection*. The DTN header fields and the local bundle fields are utilized by a number of methods that provide functionality for various bundle-related operations and assist bundle management in various occasions. The most important bundle methods are:

- *IsSameAs*: Checks whether two bundles are identical. The bundle identity consists of a combination of the source address, creation timestamp, and fragment offset. Additionally, this method also checks for size equality of the two bundles.
- *IsInSameADUAs*: Checks for source address and creation timestamp equality of two bundles. These two fields determine the ADU that the bundles belong to.
- *Precedes*: When bundles are stored in a list (such as the bundle managers described in upcoming paragraphs) the order is based on the source address, timestamp, and offset of the bundle. This method checks for precedence between two bundles.
- *IsCompleteADU*: Checks if the specified bundle represents an entire ADU that can be delivered to the application layer. A bundle containing an entire ADU has a fragment offset of 0, a bundle size equal to the ADU size, and all of its bytes are received at the relevant node (i.e. the ReceivedBytes bundle field equals the bundle size).
- *LastBundleByte*: Returns the offset of the last byte of the bundle.
- *LastReceivedByte*: Returns the offset of the last received byte of the bundle.
- *TotalOffset*: Returns the total offset for a bundle portion included in a TCP segment (includes both the bundle offset and the TCP offset).
- *LastSegmentByte*: Returns the offset of the last byte of the TCP segment that includes the specified bundle portion.

- *CanBeMergedWith*: Checks if two bundles can be merged into a single bundle. The bundles must belong to the same ADU, they must be fully received at the specified node and they must overlap in terms of the respective fragment offsets.
- *MergeWith*: Combines two bundles into a single bundle.
- *CanBeUpdatedWith*: Checks if a bundle can be updated with received bytes from another bundle. The two bundles must belong to the same ADU and their bytes must overlap.
- *UpdateWith*: Updates a bundle with the received bytes from another bundle. This method is similar to the MergeWith method; however, the UpdateWith does not change the identity fields of the bundle (offset and size), contrary to MergeWith.
- *FragmentBytes*: Fragments a bundle at the specified byte with respect to the bundle size.
- *FragmentAt*: Fragments a bundle at the specified byte offset with respect to the overall ADU. The specified offset must fall within the bundle length.
- *FullyContains*: Checks if the bundle fully contains another bundle. The two bundles must belong to the same ADU and the two bundle offsets and sizes must be such that the contained bundle falls completely within the container bundle.
- *OriginalFullyContains*: The same as FullyContains only it checks for the original bundle. Original is considered as the bundle before any local fragmentation has been conducted. In case local fragmentation has not been conducted the original and the bundle itself are identical.
- *OriginalSameAs*: Checks if the original (before any local fragmentation) bundle is the same as another bundle.
- *IsFullyReceived*: Checks if a bundle has been fully received on the local node. The ReceivedBytes field must equal the bundle size.
- *RejectCustody*, *AcceptCustody*: Creates a custody rejection/acceptance report for the specified bundle. Custody bundles include the source address, timestamp, fragment offset and bundle size of the reported bundle in appropriately prefixed fields of the DTN header.
- *IsInTransit*: Checks if the bundle is in the process of being transmitted by the related outgoing TCP agent.
- *PopulateFromReport*: Populates the bundle identity fields with respect to the custody fields of the DTN header.

BundleManagerTCP stores incoming or outgoing TCP segments at the TCP agent and assists with associating bundle identity information at the transport layer. These segments contain portions of DTN bundles temporarily stored at the transport layer that have either been received from the network and have not yet been delivered to the DTN agent (incoming data), or have been received from the DTN agent and have not yet been transmitted on the network (outgoing data). Most BundleManagerTCP functionality is provided by the following methods (at this level each ns-2 packet contains both DTN and TCP headers so the terms bundle and segment are used interchangeably):

- *AddBundle*: Adds a new TCP segment to the bundle manager list with respect to the bundle precedence rules described in the bundle Precedes method.
- *GetBundle*: Gets the segment with the specified TCP sequence number.
- *FindFirstBundle*: Finds the first segment containing data of the specified bundle.
- *Cleanup*: Discards all segments up to the specified sequence number.
- *EmptyList*: Empties the entire bundle manager list.
- *BundlesStored*: Returns the number of stored bundles.
- *RemoveBundlesAndReorder*: Removes all segments containing data of the specified bundle and reorders the sequence numbers for the remaining segments.

Similarly to the BundleManagerTCP, the BundleManagerDTN class provides all the necessary functionality that allows for management of the stored bundles at the DTN layer. The BundleManagerDTN is a singly linked list and provides various functions, the most important of which are listed below:

- *AddBundle*: Adds a bundle to the manager. If the bundle already exists, the received bytes of the already stored bundle are accordingly updated. The method takes special care for the case where the incoming bundle is the same as an already received bundle that has been, in the meantime, fragmented into multiple bundles.
- *GetFirstCompleteADU*: Retrieves the first completed ADU stored in the manager.
- *GetFirstCompleteADUForDest*: Retrieves the first completed ADU for a specific destination.
- *GetFirstUnSentBundle*: Retrieves the first bundle that has not yet been transmitted or for which retransmission is pending.
- *GetFirstExpiredBundle*: Retrieves the first bundle for which retransmission is pending.
- *GetFirstBundleBelongingTo*: Retrieves the first bundle fully contained in one specified bundle and stored in the manager after another specified bundle.
- *FindBundle*: Searches for the specified bundle in the manager and, if successful, returns the retrieved bundle to the caller.
- *Cleanup*: All completely received ADUs are removed from the manager and deleted.
- *EmptyList*: All bundles are removed from the manager and deleted.
- *RemovePointer*: The specified bundle pointer is removed from the manager.
- *RemoveBundle*: The specified bundle is removed from the manager.
- *CleanIfOriginalAcked*: Removes and deletes all bundles belonging to the same original bundle as the specified bundle (sibling fragments) and have been fully received.
- *OriginalBundleAcked*: Checks if all fragments belonging to the original bundle of the specified bundle have been acknowledged.
- *OriginalBundleCompleted*: Checks if all fragments belonging to the original bundle of the specified bundle have been fully received.
- *BundlesStored*: Returns the number of stored bundles.

- *MemoryAllocated*: Calculates the total bytes occupied by all bundles in the manager.
- *MergeAllFragments*: Merges all fragments that can be merged. Eligible fragments include fragments that have been fully received, belong to the same original bundle and their bytes, with respect to the initial ADU, overlap.
- *MergeAllFragmentsForDestination*: Same as the previous method only merges fragments for a certain destination.
- *FragmentBundleToSize*: Fragments the specified bundle to the specified size.
- *FindOriginalBundle*: Finds the original of the specified bundle and returns it to the caller.
- *FindEarlierRTTime*: Finds the earliest retransmission time for all bundles in the manager.
- *RemoveForwardedBundles*: Removes all bundles that have already been forwarded.

7.1.3. Transport Agents

The TcpDtnAgent class extends the TcpAgent class, which is the default TCP version in ns-2. Different TCP flavors can be employed by changing the parent class to the desired TCP flavor class (e.g. RenoTcpAgent). The agent includes a BundleManagerTCP containing bundle segments that are either pending for transmission or have not yet been acknowledged. The class exposes methods invoked by the attached DTN agent in order to carry out a number of required operations. The most important of these methods are:

- *SendBundle*: Accepts a bundle for transmission specifying the payload and the header bytes. The TCP agent creates the appropriate number of segments based on the total bundle size (payload plus header) and the maximum segment size. Each created segment is inserted into the bundleManagerTCP.
- *output_helper* (virtual): Called by the TCP agent when a new packet is transmitted. At this point the TcpDtnAgent populates the DTN header of the TCP packet with the appropriate information according to the bundle stored in the BundleManagerTCP.
- *recv_newack_helper* (virtual): Called by the TCP agent when a new acknowledgment arrives. If the acknowledged segment is the last segment of a transmitted bundle, the attached DtnAgent is notified, implementing the cross-layer immediate acknowledgment notification mechanism. In any case all segments with lower sequence numbers are removed from the BundleManagerTCP.

The TcpDtnAgent class also contains the following two methods that facilitate cross-layer communication with the Bundle layer (as explained in section 3.1.3):

- *CancelBundleTransmission*: Removes all segments that belong to the specified bundle from the manager, reorders the segment sequence numbers and removes the corresponding data amount from the TCP buffer.
- *BundleInTransit*: Queries whether the specified bundle is in the TCP buffer, awaiting either transmission or acknowledgment by the peer TCP agent.

Class TcpDtnSink is the peer of TcpDtnAgent that receives bundle segments at the downstream DTN node through the single *recv* function. This class also contains a BundleManagerTCP, where incoming bundle segments are stored. Bundle segments are received from the lower network layers and inserted into the bundle manager. In case incoming segments complete entire bundles the corresponding bundles are delivered to the attached DtnAgent. Delivered bundle segments are removed from the bundle manager.

Besides the transport layer extensions that were implemented in the new classes and were described in the previous paragraphs the following fixes were applied to the core TCP class, the TcpAgent:

- Fix so that the TcpAgent correctly notifies the attached application (i.e. DTN agent in our simulation model) when all data have been transmitted. Previous functionality failed to detect the end of the transmission.
- Fix so that the TcpAgent resumes sending data after the sending buffer has been emptied and new application data later arrives. Original TcpAgent functionality shipped with ns-2 failed to resume transmission after the sending buffer had been emptied for the first time.

In case UDP is used at the transport layer, a UdpDtnAgent is employed both on the upstream (i.e. bundle sending) and on the downstream (i.e. bundle receiving) ends of the connection. Also, since each UDP packet corresponds to a DTN bundle, a bundle manager object is not necessary. The UdpDtnAgent provides a method (invoked by the DTN agent at the layer above) that creates a new UDP packet adds the DTN related header information and passes it on to the layer below. Another method receives packets from the layer below, creates the appropriate bundles and delivers them to the DTN agent.

7.1.4. Connection Management

Class DtnConnection contains a unique connection ID and links to the owner and the peer DtnAgent objects. In the TCP convergence layer case, the DtnConnection contains an outgoing TcpDtnAgent for sending data to the specified peer DtnAgent and an incoming TcpDtnSink for receiving data from the specified peer DtnAgent. In the UDP convergence layer case, the connection object contains two UpdDtnAgent objects one for sending and the other for receiving data to and from the peer DtnAgent. Initially, a connection has no transport agent objects, but creates them on the fly when necessary, allowing for adding dynamic DTN routing that changes during simulation. Most DtnConnection functionality is implemented through the following functions:

- *Connect*: Connects the owner DTN agent with the peer DTN agent using the specified transport protocol (TCP or UDP). The Connect function sets up the routing and multiplexing mechanisms required for exchanging data between 2 transport agents in ns-2, invoking the *connect-tcp* and *connect-udp* TCL procedures described in 7.1.6.
- *SendBundle*: Sends the specified bundle over the appropriate outgoing agent (be it a TcpDtnAgent or a UdpDtnAgent).

Class ConnectionManager includes two singly linked lists; one containing the downstream and the other containing the upstream connections. It also contains a flag to indicate whether the downstream

connections have been setup. Finally, it provides several methods implementing connection management functionality. These methods include the following:

- *EmptyList*: Removes all connections from the connection manager.
- *DownstreamConnections*/*UpstreamConnections*: Returns the number of downstream/upstream connections respectively.
- *SetupDownstreamConnections*: Sets up all downstream connections so that they are ready for sending data.
- *AddDownstreamConnection*/*AddUpstreamConnection*: Adds a new downstream/upstream connection respectively. The new connection is assigned a new ID equal to the sequence number of the connection in the manager.
- *SetTCP*/*SetUDP*: Associates the specified transport agents with the connection to the specified peer agent.
- *IsDownstreamConnection*/*IsUpstreamConnection*: Queries the manager of whether the specified connection is an upstream or a downstream connection respectively.
- *GetDownstreamConnection*/*GetUpstreamConnection*: Retrieves the downstream/upstream connection with the specified peer respectively.
- *GetConnection*: Retrieves a connection with the specified peer (downstream or upstream).

7.1.5. DTN Agent

DTN agent extends the built-in Agent class of ns-2 and implements the core DTN functionality. The class includes a ConnectionManager holding the DTN connections of the agent, and a BundleManagerDTN holding the in-storage bundles. The basic DTN agent version also contains a timer for retransmitting bundles. The class exposes commands that create connections with upstream or downstream peers as well as attach TCP or UDP agents to already existing connections.

The DtnAgent class overrides the *sendmsg* function of the Agent class. When sendmsg is called by the application layer, the DTN agent creates an ADU of the specified size, fragments the ADU to the appropriate number of bundles according to the specified maximum bundle size, and inserts the resulting bundles into the bundle manager. For each of the produced bundles the *SendDataBundle* function is called. SendDataBundle is the sole function responsible for sending data bundles whenever this is necessary (i.e. at initial transmission requested by the application, at retransmission and at bundle relaying); if necessary it adds header bytes and then sends the bundle through the appropriate connection.

In case the bundle has been received from an upstream connection or in case no upstream connections are available (i.e. this is the source node), the outgoing connection is selected among the downstream connections (mostly applies to data bundle travelling downstream). In case the bundle has been received from a downstream connection or in case no downstream connections are available (i.e. this is the destination node) the outgoing connection is selected among the upstream connections (mostly applies to

custody report bundles travelling upstream). In both cases the selection among the downstream or upstream connection groups may be done in one of the following two ways:

- Round robin: The next connection is selected in a round robin fashion. The previously selected connection is always stored for future reference.
- Flooding: The bundle is sent over all available connections in the appropriate connection groups (i.e. downstream or upstream).

Data arrives at the agent through the *ReceiveBundlePayload* function (pseudocode included in Figure 7-2). In case of an incoming custody acceptance report, the related bundle is removed from the bundle manager, while for an incoming custody rejection report the retransmission time reset. As described in section 3.1.3, possible pending transmission of a bundle related to an incoming custody report is cancelled at the TCP layer. For incoming data bundles, it is checked whether they can be accepted or not. Data bundles can be rejected under the following circumstances: 1) the first part of the bundle is missing, 2) the entire bundle is already stored in the bundle manager, 3) there is not enough space to store the bundle. In the third scenario, a custody rejection report is also sent to the peer DTN agent where the rejected bundle originated from.

```
void DtnAgent::ReceiveBundlePayload(Bundle inBundle,
   Agent *pIncomingAgent)
{
   /* If inBundle is a status report, process it and exit. */
   if (ProcessStatusReport(inBundle))
      return;

   /* If the bundle must be rejected then exit. */
   if (RejectBundle(inBundle))
      return;

   /* At this point inBundle is a data bundle.*/
   bundleManagerDTN_.AddBundle(inBundle);
   if (bundleManagerDTN_.
      OriginalBundleCompleted(inBundle))
   {
      SendCustodyReport(inBundle, true);
   }

   if (app_) /* This is the destination node. */
   {
      bundleManagerDTN_.MergeAllFragments();
      DeliverCompletedADUs();
      bundleManagerDTN_.Cleanup();
   }
   else /* This is a relay node. */
   {
      if (!pStoredBundle->Suspended())
      {
          /* Forward bundle if routing not suspended.*/
         SendDataBundle();
      }
   }
}
```

Figure 7-2: DtnAgent Class Message Receiving Function Pseudocode.

Accepted incoming data are added to the bundle manager, which either inserts a new bundle to the list or updates the received bytes counter of an already inserted bundle. If the newly arrived data happens to complete a bundle, a custody acceptance report is sent to the peer DTN agent that the bundle originated from. If an application is attached to the agent, the agent tries to merge all eligible bundle fragments and deliver the completed ADUs to the application, purging delivered bundles from the bundle manager. In the case of a regular relay node, the received data is sent to the appropriate downstream connection, unless transmission has been suspended due to the reception of a custody rejection report.

Additionally to the main control flow, the DTN agent takes certain actions responding to a number of events, which originate either from the TCP layer, through callback mechanisms, or through the expiration of the timers included in the class. These include the following:

- *BundleTransmitted*: Callback function invoked by an outgoing TcpDtnAgent object when the last part of a certain bundle has been transmitted. This is when the DtnAgent object sets the retransmission time for the specified bundle (it is actually the time when the bundle leaves the node). The retransmission time is set based on a user-defined timeout setting. For more information on this callback see section 3.1.3.
- *RTTimerHandler*: This function is called when the retransmission timer expires. The retransmission timer is always set to the earliest retransmission time of all bundles in the bundle manager of a DTN agent. When the timer expires, all bundles with a retransmission time in the past are retransmitted and their retransmission times are reset to the user-defined timeout setting. Finally, the timer is reset, again, to the earliest retransmission time in the manager.

7.1.6. Auxiliary TCL procedures

Additionally to the previously presented C++ classes implementing the main functionality of the DTN model, a number of TCL supporting procedures have also been created. These procedures are required in order to “glue” together the C++ classes and are often invoked from the C++ domain. The most important of these procedures are briefly described below:

- *attach-dtn*: Attaches a DTN agent on the specified node. The DTN agent is informed about the node it is attached to and also about the network address of that node.
- *connect-dtn*: Connects a source DTN agent to a destination DTN agent. When invoked it adds the destination agent as a downstream connection of the source agent and the source agent as an upstream connection of the destination agent. This is achieved by invoking the relevant commands of the DtnAgent class that create a new connection with the specified peer and add this connection to the correct list (upstream or downstream) of the connection manager. The connection manager functions are described in section 7.1.4. When first created, the DTN connections do not contain the necessary transport layer agents, which are setup on demand by the following two procedures.

- *connect-tcp*: Creates and configures the required TCP agents between a source DTN agent and a destination DTN agent, provided the agents are already connected at the DTN level by the connect-dtn procedure (described in the previous bullet). For a given DTN agent pair the procedure creates and connects a TcpDtnAgent to a TcpDtnSink for the downstream path (source to destination) and a TcpDtnAgent to a TcpDtnSink for the upstream path (destination to source). The references of the created TCP agents are propagated to the appropriate DTN connections via the SetTCP function of the connection manager of the DTN agent (see 7.1.4).
- *connect-udp*: Similar to the previous procedure, only it configures the required UDP agents between two already connected at the DTN layer DTN agents. For a given DTN agent pair, the procedure creates and connects two UdpDtnAgent objects for the downstream path (source to destination) and two UdpDtnAgent objects for the upstream path (destination to source). The references of the created UDP agents are propagated to the appropriate DTN connections via the SetUDP function of the connection manager of the DTN agent (see 7.1.4).

7.2. Simulation Model Energy Extensions

The energy extensions for our DTN simulation model include the rendezvous mechanism and the 802.11 bandwidth estimation as described in sections 3.3 and 3.4. These extensions are elaborated upon in the following sections.

7.2.1. Rendezvous Mechanism

The DTN agent supports the rendezvous mechanism between the Base Station (BS) and the Mobile Receiver (MR). When a bundle destined to the MR is received at the BS, the incoming data counter is increased by the received bundle payload and the state of the rendezvous timer (RVtimer) is checked. If the timer is idle, the incoming data are the first data arriving for the MR. The DTN agent fragments the bundle that has just been received to the amount of received bytes and forwards the fragment to the MR. In this case, no previous information on the incoming data rate exists and, so the timer is set to the default value of 0.5 seconds (see Figure 7-3). In case the rendezvous timer is active the agent just stores the incoming data and takes no further action.

```
/* Send data and set initial timer.*/
bytesReceived += incomingPayload;
if ( RVtimer_.status() == TIMER_IDLE )
{
   FragmentPartiallyReceivedBundles();
   SendOutstandingBundles();
   RescheduleRVTimerDefault();
}
```

Figure 7-3: Rendezvous Mechanism Initial Data Reception at the BS Pseudocode.

When the rendezvous timer expires, the DTN agent at the BS fragments all partially received bundles, calculates the next rendezvous time according to the algorithm presented in 3.3, sends the outstanding bundles and reschedules the timer to the calculated rendezvous time. The next rendezvous time is communicated to the MR via the *NextRV* field introduced in the DTN header. The NextRV field is set to the value of the next rendezvous time on the last bundle of a data burst. For the rest of the burst bundles the value of this NextRV field is set to 0.

Upon reception of a bundle at the MR DTN agent the NextRV field on the bundle header is examined. If the value of the NextRV is non-zero and the underlying transport is UDP the agent attempts to immediately suspend the WNIC of the MR; if the NextRV is non-zero and TCP is employed, then the MR sends a custody report. Suspending of the interface in the TCP case will be attempted from within the BundleAcknowledged callback (described in 7.1.5), when the transmitted custody report is acknowledged by the BS TCP. If the value of the NextRV is zero no action is taken.

When the WNIC is suspended, the agent sets a wake-up timer (WUTimer) to expire at the next rendezvous time. The WUTimer wakes up the wireless interface at the appropriate time so that the next data burst from the BS can be received. When setting the timer, the DTN agent takes into consideration the transition time from sleep to the active state.

Additionally to the two timers and the NextRV header field, the energy-saving features are supported by the following methods:

- *FragmentReceivedBytes* (Bundle class): This method has been added to the Bundle class so that partially received bundles (bundles for which some but not all bytes have been received) are dynamically fragmented to the number of received bytes. For each partially received bundle, two new bundle fragments are created: one that contains the received bytes and is fully received and one for which no bytes have been received.
- *FragmentAllBundles* (BundleManager class): Fragments all bundles in the manager that have been partially received to the number of received bytes. Invokes the FragmentReceivedBytes method on all partially received bytes in the bundle manager.
- *BundleAcknowledged* (DtnAgent class): Callback function invoked by an outgoing TcpDtnAgent object when the last part of a certain bundle has been acknowledged by the peer TcpDtnSink. This callback allows the DTN agent to receive immediate delivery notification from the underlying reliable transport protocol, instead of relying on custody messages at the Bundle layer. In case the acknowledged bundle is a custody report acknowledging a previously received bundle, this callback also notifies about the completion of a bundle/custody message exchange. For more information on this callback see section 3.3.
- *SuspendInterface* (DtnAgent class): This method is invoked on a bundle with a NextRV header field of a positive value. Firstly, it is checked whether the NextRV value refers to a future time, as it is possible that by the time the corresponding bundle arrives at the MR the next rendezvous time has already elapsed. Secondly, it is checked whether the time until the next

rendezvous is sufficient for the WNIC to switch to the sleep state and then back to the active state (i.e. it is more than 2 transition times). If both conditions apply, the physical interface of the node (WNIC) is switched to the sleep state and the wake-up timer is rescheduled at the next rendezvous time.

During experimentation with the energy consumption in the DTN overlay we observed several inaccuracies of the energy model in ns-2, which we corrected. These inaccuracies were mostly related to sleep and transition energy expenditure calculations. Additionally, we added detailed tracing functionality in order to closely follow the WNIC state transitions.

7.2.2. 802.11 Bandwidth Estimation

Bandwidth estimation for the 802.11 MAC protocol in ns-2 has been facilitated by appropriately modifying the relevant C++ classes (see section 3.4 for the details of the mechanism). The bandwidth estimation mechanism records medium utilization at the Base Station in terms of the following protocol states:

- *Send*: Frames of any type are transmitted (protocol or data frames). The Send state is switched on when the WNIC starts transmitting and switched off when transmission concludes. Transmission completion is signified by a call to the interface timer (IFTimer), which is associated with the wireless interface and is engaged anytime signals are transmitted on the wireless medium.
- *Receive*: Frames which may or may not be destined to the BS are received. The Receive state is switched on when the WNIC starts receiving and switched off when reception concludes, which is signified by a call to the handler of the reception timer (RxTimer). The RxTimer is associated with the frame reception in the 802.11 MAC code.
- *Backoff*: The BS is backing off as part of the 802.11 Congestion Avoidance mechanism. The Backoff state is switched on and off from within the Backoff timer code that takes care of the CA mechanism. The Backoff state is switched on at the start of the timer and is switched back off when the timer expires.
- *Defer*: The BS is deferring transmission observing the Short and Distributed Inter-Frame Spaces (SIFS and DIFS) specified by the 802.11 MAC protocol. The Defer state is switched on at the Defer timer and switched off with the help of a newly introduced IFS timer.

If the 802.11 MAC layer is in any of the above four states the medium is considered busy; otherwise, the medium is considered idle. The above states are not all mutually exclusive and, therefore, special care must be taken so that busy intervals are not accounted for multiple times. Medium utilization is recorded by the *UpdateBEValues* function, which is invoked every time there is a state switch. The function checks if the latest state switch brings the MAC layer from 1 to 0 active states or from 0 to 1 active states and records the previous interval as busy or idle respectively. In case the latest state switch results in more than 1 active states or the number of active states reduces from 2 to 1 no action is taken.

Bandwidth utilization is monitored with the help of two arrays; one for the busy and the other for the total durations. The array size is determined by the BE_NUMBER_OF_SLOTS macro and the time interval each slot corresponds to is determined by the BE_SLOT_DURATION macro. The updating function either adds the duration of busy intervals to the first element of both arrays or adds the duration of idle intervals to the first element of the total duration array. Using multiple slots for monitoring utilization serves the purpose of giving more weight to recent measurements and, eventually, discarding measurements that are two far in the past. Shifting the array elements and calculating the bandwidth utilization value takes place with the help of the following functions:

- *BEHandler*: This is a handler for a newly introduced BETimer. The BETimer is set to expire according to the BE_SLOT_DURATION macro. The handler records the outstanding utilization for the last interval in the first element or elements (depending on the interval type) of the utilization arrays and then shifts the elements of both arrays to the right, eliminating the last element. The first elements of both arrays are initialized to 0 and the timer is reset to the slot duration.
- *GetUtilization*: This is the API that the 802.11 MAC exposes to higher network layers that need access to the utilization values. The function calculates the average utilization applying higher weights to more recent time slots based on the application requirements. The default behavior of the function is aligned with the description of the BE algorithm included in section 3.4. The caller may use the utilization value either directly, taking action with respect to the network load, or indirectly, taking action with respect to the available bandwidth (requires multiplication of the received value by the nominal network bandwidth).

7.3. Simulation Model Space Extensions

The space extensions of our DTN simulation model contain features that refer to either the space segment (data production and transmission on the satellite) or the ground segment (data distribution from the ground stations to the end-users). The respective model enhancements are described in the following sections.

7.3.1. Satellite Reliable Broadcasting

Data production at the satellite is simulated by a simple application (SatApp) inheriting the Application ns-2 class and implemented entirely in TCL. The application is attached to the DTN agent that is, in turn, attached to the satellite node. SatApp produces an ADU of a specified size and Type-of-Service (ToS) every some time period (see section 4.5.1 for more information). The application is configured through the following parameters set at simulation time:

- *Number-of-Bytes*: The size of the produced ADUs.
- *Repeat Time*: The interval between creating successive ADUs.
- *Type-of-Service*: The ToS for the produced ADUs.

An arbitrary number of application modules can be attached to the same DTN agent, enabling the simulation of complex data generation patterns. ADUs are received by the DTN agent, fragmented into the appropriate number of bundles and entered into the transmission buffer.

Satellite-deployed DTN agents use UDP as the underlying protocol of the space link with ground stations. The packet/frame size overhead can be configured based on the total overhead of the data link protocols, independently of the bundle header imposed by the DTN. Data broadcasting is simulated through a group of point-to-point links from the satellite (Sat) to the ground stations (GS). The point-to-point approach was preferred over a pure broadcast method (e.g. using a wireless protocol such as 802.11), since it allows for precise control over the space link characteristics (bandwidth, delay, error rate).

Each Sat node has links to all GS nodes that can be set to up or down by utilizing the dynamic topology capabilities of ns-2. For each link there is a corresponding DTN connection, connecting the Sat to the GS at the DTN layer. DTN connections stored in the agent's ConnectionManager are flat, meaning there is no distinction between downstream and upstream connections. When setting a link to the up or down state, the corresponding DTN virtual connections are created or destroyed respectively via a number of auxiliary TCL functions. The DTN agent at the satellite initiates the transmission of individual bundles, one after the other, to all connections available at transmission initiation time. The agent places the bundle data in the link queue of all active links and waits until the entire bundle has been transmitted, when it initiates the transmission of the next bundle.

In order to pace the transmission even in the absence of any connected GSs, we use a hidden node that is always connected to the satellite. Transmission pacing is facilitated by a custom queue type, which notifies the satellite DTN agent whenever it empties. Built-in ns-2 queues do not support such notification mechanism, compelling us to implement the desired functionality in a new type of queue. The queue is named *FlowControlledDropTail*, due to its quality of allowing control of the link flow, and is employed on all links from the satellite to the GSs, including the link with the hidden node. When the queue empties, the DTN agent is notified via a callback function and the next bundle is placed on the links. Since all links have the same bandwidth (they are used to simulate a broadcast medium), using the callback on the hidden node link allows for pacing the transmission on all links simultaneously. Each bundle is transmitted to the GS DTN agents connected at the time of bundle transmission initiation and is received by those which remain connected for the entire duration of the transmission. Setting up of the hidden node, the UDP agent, and the flow-controlling queue is facilitated by the TCL procedure *set-broadcast-mode* of the DtnAgent class.

Transmission Scheduling

The DTN agent at the satellite implements the randomized retransmission pattern as described in section 4.5.2. Once all new bundles have been transmitted for the first time a random bundle is selected from the buffer. Transmission resumes at the first bundle of the ADU that the randomly selected bundle

belongs to. At the reception of an acknowledgment, the corresponding bundle is removed from the buffer. This process is facilitated by the two variables below:

- *LastSentBundleFirstTime*: Points at the last bundle that was transmitted for the first time.
- *LastSentBundle*: Points at the last bundle that was retransmitted.

Using the above variables the next bundle is selected with respect to the algorithm presented in Figure 7-4. Briefly, the algorithm checks if there are any unsent bundles, in which case it transmits the earliest of them. Otherwise, it resumes with retransmission of the next of the LastSentBundle, provided that it belongs to the same ADU as the LastSentBundle. If the next of the LastSentBundle does not belong to the same ADU then a new bundle is randomly chosen.

The DTN agent at the satellite exposes additional interfaces for pausing and resuming the transmission. Through these interfaces the transmission at the satellite agent may freeze by some external trigger, while normally accepting new data, facilitating the simulation of the suspend/resume mechanism. This functionality is necessary in order to simulate the feature of limiting the transmission for the duration of contacts (see section 4.5.3).

```
if LastSentBundleFirstTime not set
{
   nextBundle = GetFirstUnsentBundle();
   LastSentSnFPDUFirstTime = nextBundle;
}
else if GetNextBundleOf(LastSentBundleFirstTime) is set
{
   nextBundle = GetNextBundleOf(LastSentBundleFirstTime);
   LastSentBundleFirstTime = nextBundle;
}
if nextBundle is not set
{
   nextBundle = GetNextBundle(LastSentBundle);

   if nextBundle is not set OR nextBundle->NotInSameADUAs(LastSentBundle)
   {
      nextBundle = GetRandomBundle();
   }
   LastSentBundle = nextBundle;
}
```

Figure 7-4: Selection of the Next Bundle for Transmission Pseudocode.

7.3.2. Ground Peer-to-Peer Multicasting

Once the DTN bundles broadcast by the satellite are received at a GS, they are forwarded to the appropriate nodes of the DTN overlay based on route calculations taken place prior to simulation start. The EUs connected to the overlay may subscribe for receiving data belonging to one or more ToS and also act as routers in the corresponding application-layer multicast trees (see 4.4.2). Route calculation involves

creating one overlay multicast distribution tree per GS, per ToS, utilizing the virtual connection feature of the agent. Again, as it is also the case for the DTN agent deployed on the satellite nodes, the connections in the ConnectionManager have been flattened, losing their downstream or upstream property. Additionally, each Connection maintains a list of the ToS that have been registered with it. The main distinction of DTN connections is whether they refer to a space link (between Sat and GS) or to a ground link (between GS and EU or between two EUs) and, respectively, employ UDP or TCP.

Setting up the connections among the DTN nodes takes place according to the logic in the simplified TCL pseudocode shown in Figure 7-5. The algorithm utilizes IP routing information that has already been created by ns-2 in a prior step of the simulation preparation. For every EU, GS and ToS the algorithm traverses the IP network looking for a DTN node subscribed to the particular ToS. If one is found, a connection from the EU to the DTN node is created; otherwise, a connection between the EU and the GS is created.

```
foreach endUser $endUsersList {
   foreach groundStation $groundStationsList {
      foreach ToS $ToSList {
         set startNode $endUser
         while { 1 } {
            nextNode = FindNextIPNodeFromTo(startNode, groundStation)
            if nextNode is IP node
               set startNode $ nextNode
               continue
            if nextNode is subscribed to ToS OR equals groundStation
               create-earth-connection(endUser, nextNode, ToS)
               next endUser
         }
      }
   }
}
```

Figure 7-5: Ground Network Route Setup Pseudocode

The ground segment may include a Mission Control Center (MCC) entity, in which case, routing is done in two tiers. In the first tier connections are created between the MCC and each GS for all ToS and, in the second tier, connections are created from the EUs to the MCC. The relevant route setup process is shown in the pseudocode snippet of Figure 7-6.

The *create-earth-connection* procedure creates a connection between two nodes and inserts the specified ToS in the connection ToS list. In case a connection between the specified two nodes already exists, the ToS is simply added to the connection ToS list. Routing during simulation relies on this ToS list stored with each DTN connection. An incoming bundle is forwarded to all connections for which the bundle ToS is registered, except for the incoming connection.

```
foreach MCC $MCCsList_ {
   foreach GS $groundStationsList_ {
         foreach ToS $ToSList {
            create-earth-connection(MCC, GS, ToS)
         }
      }
   }
   foreach endUser $endUsersList {
      foreach MCC $MCCsList {
         foreach ToS $ToSList {
               create-earth-connection(endUser, MCC, ToS)
      }
   }
}
```

Figure 7-6: Two-tier Ground Network Route Setup Pseudocode

Multicast Tree Pruning

Due to the broadcast nature of the satellite transmission, it is possible that a bundle is simultaneously received by multiple GSs and, consequently, forwarded on multiple multicast trees. EUs subscribed to the bundle ToS will possibly receive the bundle from multiple adjacent nodes. In order to minimize network load, avoiding duplicate data transmission, the DTN agent employs a pruning mechanism. The pruning mechanism relies on a list of "banned" bundle IDs saved with the connection object, as well as the custody rejection mechanism.

The list of banned bundles in all connections is empty at the start of the simulation. Whenever a DTN agent at an EU receives a bundle it has already received, the agent sends a custody rejection report to the originating node. The originating node immediately interrupts transmission of the bundle in question over the corresponding connection (the rejection may arrive before the transmission of the entire bundle completes) and inserts the bundle ID in the connection banned bundles list. In the future, segments containing portions of the banned bundle that may arrive at the originating node will not be forwarded to connections for which the incoming bundle is banned. In short, the routing mechanism presented in the previous subsection does not indiscriminately forward an incoming bundle to all connections, but only to the subset of those for which the incoming bundle is not banned. This dynamic pruning of multicast trees ensures elimination of duplicate traffic caused by multiple bundle reception on the ground.

Acknowledgment Uplinking

The DTN agent supports optional uplink functionality, in which case acknowledgments are either generated directly at the receiving GS or at a Principal Investigator (PI) subscribed to all ToS. In the case where a PI is present, the acknowledgments are routed from the PI to the designated Uplink GS (UGS) and

are buffered there until the next contact, when they are uplinked to the destination satellite. The connections for the acknowledgment path are created during the initial route setup phase. One such connection is setup between the PI and the UGS and another between the UGS and the satellite.

Bundle reception acknowledgments are, essentially, custody acceptance reports. Buffering of these reports at the DTN agent of the UGS is facilitated by an additional bundle manager. Custody acceptance reports are bundles in their own right and, therefore, the functionality implemented as part of the bundle manager (see section 7.1.2) can be utilized with minor modifications. The uplink functionality supports multiple satellites, distinguishing custody acceptance reports in the same bundle manager by their destination address.

7.4. Other Simulation Model Extensions

This section includes two sets of extensions to the original DTN model that support the storage-based routing, as described in chapter 5, and the cryptographic operations, used in research work outside the scope of this dissertation. The respective extensions are described in the following sections. Especially for the cryptographic extensions, a high-level description of the motivation behind the development of the tools is also included.

7.4.1. DTN Storage-Based Routing Extension

Storage-based routing adopting Game Theory-inspired mechanisms is supported by the appropriate extensions on the basic DTN simulation model. In the Connection and ConnectionManager classes, these extensions serve the purpose of monitoring bundle acceptance/rejection history with peer DTN nodes. Keeping track of bundle exchange history with peers is implemented with the help of two arrays, holding the accepted and the total bundles forwarded to a Connection object. The Connection updates the first element of each of the two arrays. When a bundle is accepted, both elements are incremented; otherwise, only the first element of the total bundle array is incremented.

The use of an array allows for temporally delimiting the acceptance history, keeping recent measurements in the first elements of the arrays and pushing older measurements towards the arrays ends. Using this technique, bundle acceptance ratio is monitored with respect to a number of time slots. The number of time slots is dictated by the CON_NUMBER_OF_SLOTS macro and the duration is dictated by the CON_SLOT_DURATION macro. The default value for both macros is 5 (5 slots of 5 seconds each), according to the description included in section 5.3.

The array contents are shifted in regular intervals towards the array end, with the help of the newly introduced *ShiftTimer*. The new functionality is provided by the following functions in the Connection and the ConnectionManager classes:

- Connection:
    - *BundleAccepted*: When a bundle is accepted from the connection peer DTN agent this function increments the first element of both the accepted and total arrays by 1.

- *BundleRejected*: When a bundle is rejected from the connection peer DTN agent this function increments the first element of only the total array by 1.
- *ShiftArrays*: Shifts both arrays by 1 place and zeros out the first element of each array. The last element of each array is discarded.
- *GetAcceptRate*: Calculates the bundle acceptance ratio (BAR) for the connection. Recent slots have a higher contribution to the overall acceptance rate than older slots, essentially allowing the connection to "forget" older measurements.

- ConnectionManager:
  - *ShiftTimerHandler*: Invoked by the ShiftTimer and shifting the accepted and total arrays of all connections stored in the connection manager.

The above functionality is invoked by the DTN agent when custody acceptance or rejection reports are received. The agent locates the corresponding connection on which the custody report has been received and updates the acceptance history. Array shifting is done automatically through the described ShiftTimer.

Bundle Acceptance and Routing

Bundle acceptance functionality has been implemented at the main points of entrance for a bundle in a DTN agent: *sendmsg* (for locally created bundles) and *ReceiveBundlePayload* (for bundles received on the network). For locally created bundles the DTN agent considers the available storage space with respect to the configured bundle acceptance strategy. When in partially or totally selfish mode, locally created bundles are always serviced as long as sufficient storage space is available. When in impartial mode, a bundle is accepted in case the storage space occupancy (a value from 0 to 1) is higher than a randomly created number, uniformly distributed in the interval from 0 to 1. The random number is created with the help of the *RNG* random number generator utility library provided by ns-2.

Similarly, for bundles received on the network, the agent considers the configured bundle acceptance strategy and the local storage availability. In case in totally selfish mode, a bundle received on the network is rejected and a rejection report is sent to the originating node, whereas when in partially selfish mode the acceptance depends on the storage occupancy. In case a deterministic, history-based strategy is in place, the incoming connection is queried for the BAR and the bundle is accepted as long as the connection BAR is higher than the local storage utilization. In case of a non-deterministic strategy, acceptance is decided upon a combination of a randomly generated number and the storage availability, according to the currently active strategy. In all cases, the relevant report (acceptance or rejection) is sent to the originating node.

Routing in the storage-based routing extensions relies on the distinction between upstream and downstream connections. Bundles received from an upstream connection are forwarded to downstream connections and vice versa. Outgoing connection selection takes place in a round-robin fashion, as per the default behavior of the DTN agent.

7.4.2. DTN Cryptographic Operations

The purpose for the cryptographic operations enhancements to the DTN simulation model was to expand the DTN agent in order to facilitate experimentation with cryptography-related protocols and functions. These enhancements provided the infrastructure for experimentation and development of cryptographic key exchange protocols as part of an automated key exchange protocol evaluation framework for DTNs.

Figure 7-7 depicts a representative scenario, involving DTN communication among space assets, for which a key exchange protocol evaluation framework could be useful. A rover on Mars is required to transfer scientific data, gathered on the planet's surface, back to Earth. The data may reach the Earth over one of the following two possible routes:

- Source → Untrusted Node → Trusted Node → Base Station → Destination.
- Source → Trusted Node → Earth → Destination.

The first route presents the advantage of splitting the long, and possibly faulty, hop between the Source and the Trusted Node into two hops of better quality and more frequent contact opportunity. However, utilizing this route presumes the existence of a shared key between the Source and the Trusted Node, since cleartext data may not traverse the Untrusted Node. The key exchange procedure between the Source and the Trusted Node, necessary to establish a shared key, may involve a number of passes, depending on the employed protocol. Considering the long delays, the lossy/low-bandwidth links and the intermittent connectivity provided by space communications, the key exchange procedure may require significantly long time to complete.

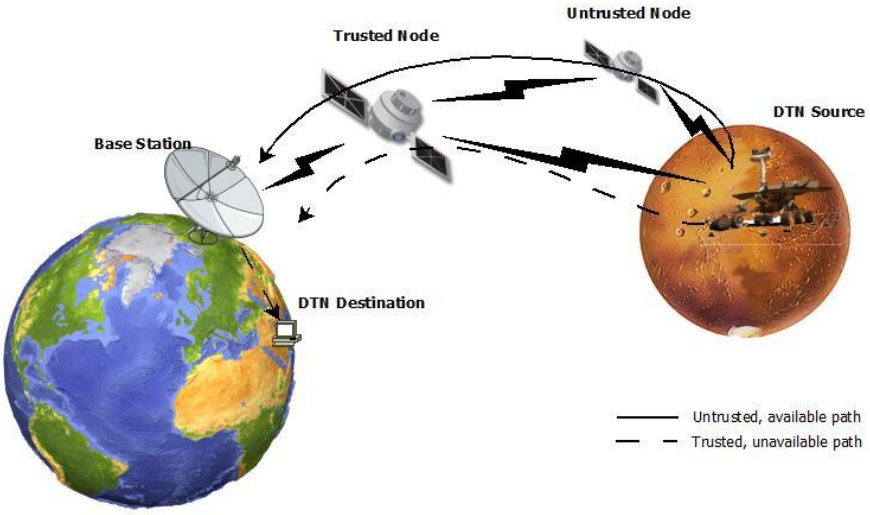

Figure 7-7: Alternative Data Transfer Route from Mart to Earth [35].

Additional delay will be introduced by the encryption/decryption procedures along the data transfer path. Once a shared key has been established between the Source and the Trusted Node the data will go

through the stages depicted in Figure 7-8, from right to left. Data created at the Source will be encrypted, introducing the corresponding encryption delay, and transmitted to the Untrusted Node. The Untrusted Node blindly relays the encrypted data, which are then decrypted at the Trusted Node, where the corresponding decryption delay is introduced. Finally, the data travels the last two hops of the route to the Destination in cleartext.

In the representative scenario described in the previous paragraphs, the key exchange protocol evaluation framework can be used to assess whether the extra time required for the key exchange, as well as the extra delay introduced from encrypting/decrypting data along the path, are compensated by the improved contact opportunities provided by the utilization of the intermediate Untrusted Node. Generally, the framework can be used in order to comparatively evaluate alternative key exchange and/or encryption/decryption algorithms and assist in selecting the most suitable communication method, given the topology, contacts and characteristics of the key exchange protocols and cryptographic algorithms.

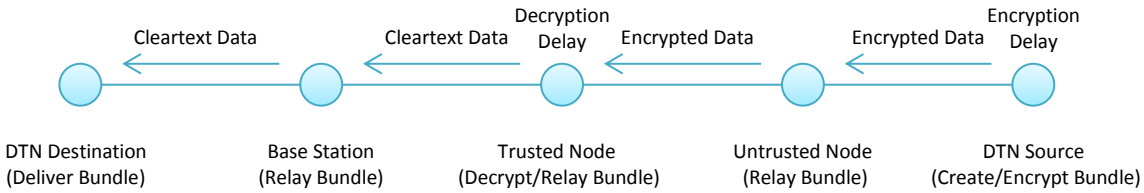

Figure 7-8: Stages of the Data Transfer from Source to Destination.

Functionality

The control flow of a DTN agent for which the cryptographic operations are enabled is shown in the flow chart of Figure 7-9. Initially the agent is waiting for incoming data. If the data are received on the network (left branch) it is checked whether a full ADU is formed. If not, the agent goes back to waiting for data. It is noted that cryptographic operations can only be applied on bundles containing entire ADUs and not on fragments. In case a full ADU is completed or the incoming data has been locally produced (implies a full ADU) then the ADU is checked whether it is cleartext or encrypted so that it can be encrypted or decrypted respectively. Depending on the required operation, the necessary delay is calculated and the corresponding crypto-timer is set.

When the crypto-timer expires, the DTN agent executes the control flow shown in Figure 7-10. The bundle manager is searched for bundles requiring cryptographic operations. Each such bundle is checked whether it is cleartext or encrypted and it is, subsequently, encrypted or decrypted respectively. The encryption or decryption may change the size of the bundle, depending on the simulated encryption algorithm. Once the bundle has been processed it is delivered to the application (in case of the destination node) or forwarded down the network path (in case of a relay node). The same process is repeated for all bundle requiring cryptographic operations.

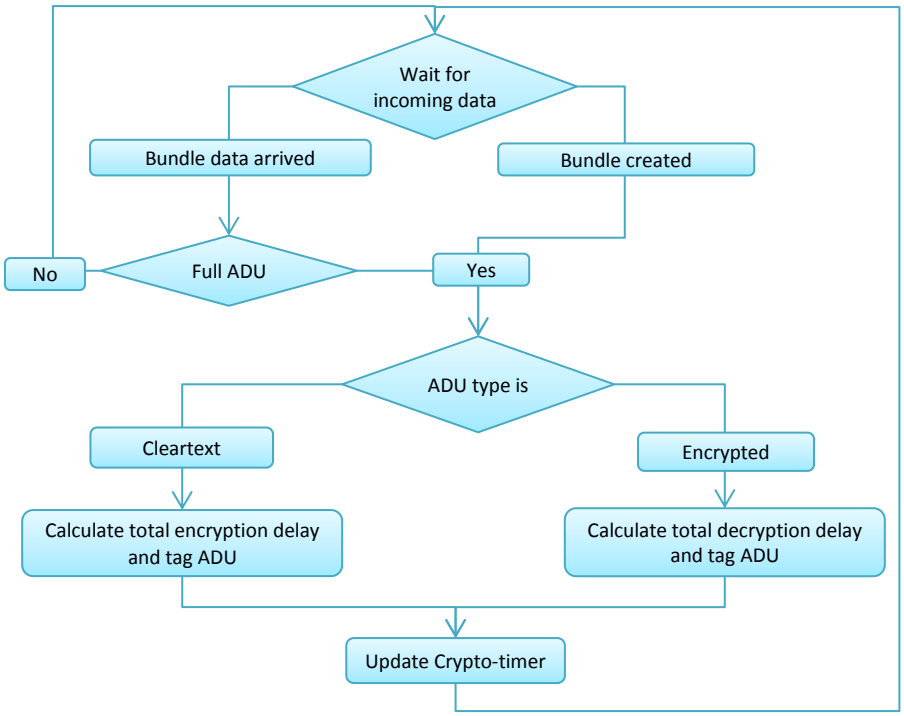

Figure 7-9: Incoming Data Control Flow at the DTN Agent.

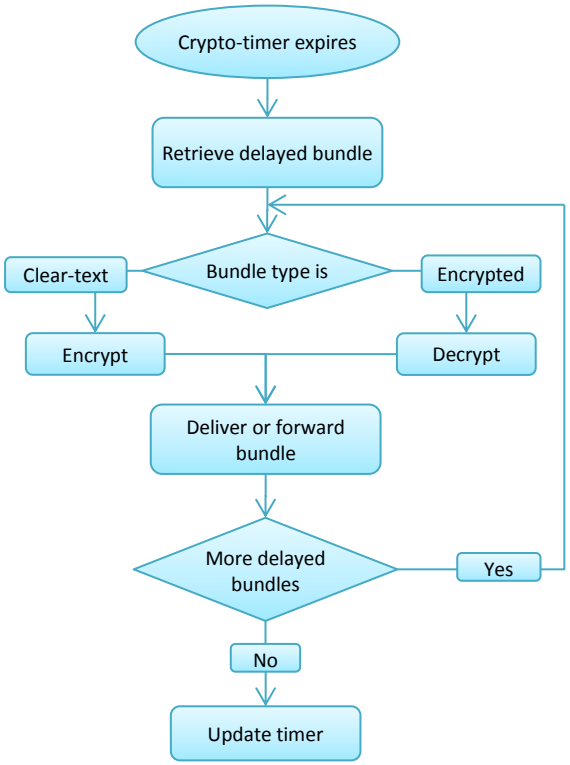

Figure 7-10: Crypto-timer Expiration Control Flow at the DTN Agent.

Implementation Details

The cryptography-related functionality is provided by a number of additions to the original DTN agent simulation model, designed to support different encryption algorithms and key exchange protocols. At the bundle level the following fields and methods have been added:

- Header fields:
  - *IsEncrypted*: Denotes whether the bundle is in cleartext or encrypted.
  - *ClearTextSize*: For encrypted bundles this field contains the cleartext size. The bundle size will be restored to the specified value at decryption time.
- Bundle functionality:
  - *CryptoDelayTime*: The time when the crypto operations on this bundle will be completed.
  - *CryptoProcessed*: Internal field denoting whether the bundle has been already processed or it requires processing with respect to some cryptographic operation.
  - *EncryptDynamic*: Encrypts the bundle setting all the required bundle fields and adds the required padding specified by the block size of the simulated encryption algorithm.
  - *DecryptDynamic*: Decrypts the bundle returning the size to the cleartext value stored in the bundle header.
  - *EncryptStatic*: Encrypts the bundle setting the resulting size to a static value.
  - *DecryptStatic*: Decrypts the bundle setting the size to a static value.

At the DTN agent the following settings have been added:

- *EnableCrypto*: Specifies whether the agent will execute cryptographic operations or it will simply relay traffic.
- *CryptoDelay*: Specifies a constant delay value for the cryptographic operations.
- *CryptoOutputSize*: The resulting size of an encrypted bundle set during static encryption.
- *CryptoBlockSize*: The block size for the encryption algorithm in case of dynamic encryption.
- *EncryptionTime*: The time required to encrypt a bundle.
- *DecryptionTime*: The time required to decrypt a bundle.
- *NoPasses*: The number of passes required by the simulated key exchange protocol. This field combined with a *state* variable, which indicates the current pass number, allows for external modification of the cryptographic parameters with respect to the simulated key exchange protocol.

It is noted that the agent is assumed to have a single processing engine so that new bundles requiring cryptographic processing are essentially queued to be processed after all pending cryptographic operations are completed. This functionality is supported by the newly introduced *CryptoTimer*.

8. Experimental Results

This chapter presents the simulation results for each of the three main solutions proposed in this work. Namely, section 8.1 presents the energy-efficient DTN overlay simulations, section 8.2 presents the decentralized space-data dissemination with DTN simulations, and section 8.3 presents the storage-constrained routing in opportunistic DTNs simulations.

8.1. Energy-Efficient DTN Overlay

The following sections report on each for the three development stages of the energy-efficient DTN overlay.

8.1.1. Stage 1 (Proof-of-Concept)

We begin the presentation of the experimental results with the proof-of-concept stage of the energy-efficient overlay, as described in section 6.1.2. The results are organized in five subsections according to the corresponding experimental setups in the following way: varying bottleneck bandwidth FTP traffic (with and without a competing flow), varying wireless error FTP traffic (with and without a competing flow), varying TBO FTP traffic (with and without a competing flow), and varying TBO CBR traffic.

Varying Bandwidth FTP Transfer

In this set of experiments the backbone link is assigned a constant delay of 300 ms, while the bandwidth varies from 0.5 Mbps to 2.5 Mbps in steps of 0.5 Mbps. The total transfer duration (Table 8-1) is virtually identical for both the E2E and the Split cases, and is greatly affected by the bandwidth of the backbone link because of network congestion. Since the upstream link (Src – Rel-IP) has a 2 Mbps bandwidth, setting the backbone bandwidth to values greater than 2 Mbps does not affect simulation results (Src – Rel-IP becomes the bottleneck of the connection and the backbone is underutilized). Because of this, the transfer duration is equal for bandwidth values of 2 Mbps and 2.5 Mbps.

L34 bandwidth (Mbps)	Duration E2E (sec)	Duration Split (sec)
0.5	216.85	216.23
1	145.63	145.19
1.5	111.45	110.98
2	90.09	90.34
2.5	89.95	90.14

Table 8-1: Varying Bandwidth, No Competing Flow Transfer Duration.

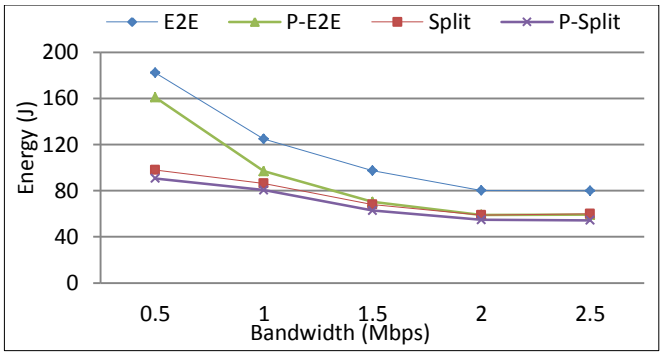

Figure 8-1: FTP, Varying Bandwidth, No Competing Flow Energy Expenditure (E2E, Split) and Potential Energy Expenditure (P-E2E, P-Split).

In terms of the energy efficiency, it is obvious from the chart of Figure 8-1 (lines E2E and Split) that smaller bandwidth values for the bottleneck link lead to greater energy gains for the Split approach. In the extreme setting of the 0.5 Mbps bandwidth, the energy conservation achieved in the Split case is approximately 46% of the energy expenditure in the E2E case.

As the network congestion eases due to the increase of the bottleneck link capacity, the energy conservation becomes less dramatic, remaining however at a significant level. In the 2.0 Mbps and the 2.5 Mbps bandwidth cases it reaches the 20 J value, representing around 25% of the energy expenditure when no energy saving mechanism is employed. When severe network congestion is present, the overall transmission duration in both cases is particularly high. In the E2E case, the mobile node at the receiving end is involved in this extra effort and has to remain in active mode for a considerable amount of the transmission duration, waiting for incoming packets to arrive. Conversely, in the Split case the BS absorbs the effects of the network congestion (lost packets that lead to out-of-order data reception) and only transmits to the mobile host when consequent, buffer-sized chunks of data are available.

In the intervals during which the BS accumulates incoming data, the mobile node has the opportunity to switch its WNIC to sleep mode through the rendezvous mechanism, achieving great energy savings. The BS is receiving the incoming data without engaging the receiving node in the process. Upon successful bundle reception, forwarding of the data to the mobile node starts. In the meantime, the WNIC of the mobile host can switch to sleep mode and save significant amounts of energy. The higher the congestion level the greater the energy gains achieved by the Split approach.

Figure 8-2 contains two additional lines representing the potential energy expenditure of the E2E and the Split cases (labeled as P-E2E and P-Split), based on experimental data from our original work on the subject [16]. For a heavily congested network (a bottleneck bandwidth of 0.5 Mbps) the E2E and P-E2E cases have comparable values. Reasonably, the E2E case exhibits the highest energy expenditure since in this case, no energy conserving mechanism is employed. The P-E2E value for the same setting is 21 J lower. This amount of energy could be potentially saved during long intervals of inactivity for the mobile host, when packet drops due to congestion and timeouts occur. When no congestion is present on the network, the P-E2E case produces almost identical results as the Split case (>= 1.5 Mbps), while the E2E case requires additionally 33% energy for the data transfer as compared to the other setups. For all bottleneck bandwidth values the Split case has slightly higher energy expenditure than the P-Split case, showing that the rendezvous mechanism is capable of taking advantage of almost all of the energy saving opportunities.

Bottleneck bandwidth (Mbps)	Duration E2E (sec)	Duration Split (sec)
0.5	364.61	358.45
1	225.87	194.86
1.5	159.26	158.84
2	125.39	124.69
2.5	111.68	111.19
3	89.79	89.44
3.5	90.74	90.32

Table 8-2: FTP, Varying Bandwidth, Competing Flow Transfer Duration.

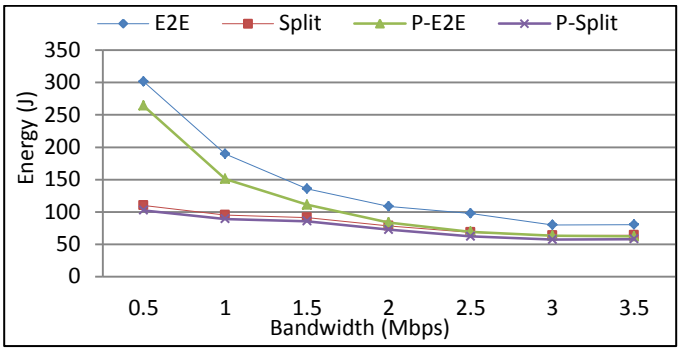

Figure 8-2: FTP, Varying Bandwidth, Competing Flow Energy Expenditure (E2E, Split) and Potential Energy Expenditure (P-E2E, P-Split).

In the previous scenario we added cross-traffic and rerun the experiments. In this new set of experiments the maximum bandwidth value for the backbone link was increased to 3.5, since beyond that capacity the measurements remain unchanged. As expected, the transfer duration, shown in Table 8-2, is significantly longer for both the E2E and the Split approaches when a competing flow is present. Nevertheless, unlike the previous scenario, small bandwidth values (i.e. 0.5 and 1 Mbps) result to a slightly shorter transfer duration for the Split case.

Regarding the energy consumption, it can be observed in Figure 8-2 (lines E2E and Split) that despite the significantly longer time it takes for the transfer to complete for the 0.5 Mbps setting (over 65% more in both cases), approach the additional energy requirement in the E2E is 65%, while in the Split approach the additional energy requirement is only around 13%. As the bandwidth increases, and the network is relieved from congestion, the energy conserved by the Split approach is limited to approximately 17 J (a value that was observed in almost all congestion-free experiments).

The conclusions that can be drawn with respect to the potential energy expenditure are roughly equivalent to those of the scenario when no competing flow is present. The only difference is that, because of the additional network congestion, at low bandwidth values it takes longer for the P-E2E energy expenditure to reach the energy expenditure of the Split and P-Split cases. Again, the main conclusion from this chart is that the rendezvous mechanism takes advantage of almost all energy conserving potential.

Figure 8-3 depicts the state transitions for the 10 indicative seconds of the file transfer in both the E2E and the Split cases when the backbone bandwidth is 2 Mbps and a competing flow is present. The active state includes sending, receiving and idle intervals (not adequately long for switching to sleep state), while the potential sleep state includes idle intervals long enough to accommodate transition to the sleep state. This chart emphasizes the increased potential for energy saving that the Split application creates irrespectively of whether a mechanism to take advantage this potential (such as the rendezvous mechanism) is in place.

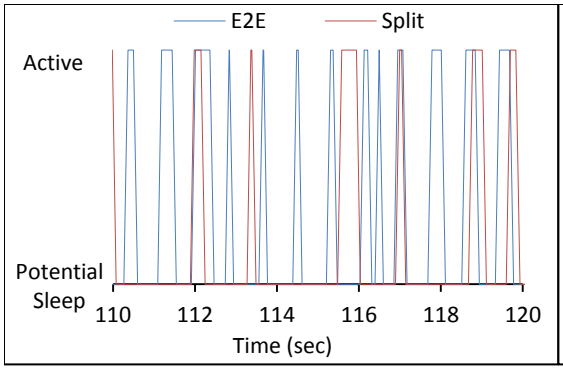

Figure 8-3: FTP, 2Mbps Bandwidth, Competing Flow State Transitions.

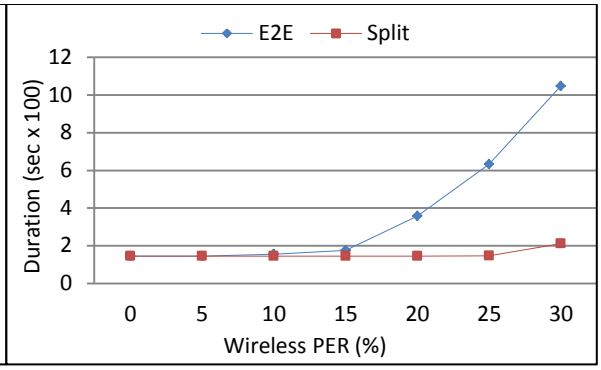

Figure 8-4: FTP, Varying Wireless PER, No Competing Flow Duration.

It becomes clear from the chart that, in the E2E case, the WNIC needs to switch to active more frequently and that usually it needs to remain active for longer times than in the Split case. For example, between 110 and 114 seconds, in the E2E case the WNIC needs to switch to active 5 times, while in the Split case only 3 switches of significantly shorter duration are required.

Varying Wireless Error FTP Transfer

The purpose of this scenario is to reveal the effect of packet error on the wireless part of the network both on the transfer duration as well as the energy efficiency of the E2E and the Split approaches. The backbone link was assigned a 300 ms delay and a 1 Mbps bandwidth. The delay was selected so that it is considerably longer than the delays of the rest of the links in the topology and the bandwidth was selected so that the backbone is also the bottleneck link of the transfer route (i.e. network congestion is present).

At first glance, it may be striking that as the wireless packet error rate (PER) increases the duration of the E2E transfer increases as well (Figure 8-4), reaching in the 30% case more than eight times the duration of the 0% case, whereas the duration of the Split case transfer remains almost the same across all runs. Part of the explanation to this, seemingly bizarre, result can be found in the Indirect-TCP solution proposed in [83]. In the E2E scenario, it takes very long for TCP to recover from packets lost due to the wireless error, because of the long RTT of the connection (around 800 ms). On the contrary, the RTT of the wireless link is measured to be around 6-7 ms, and, therefore, a TCP connection between the BS and the mobile host can be very swift in recovering the lost packets. Additionally, since the RTT of the wired part of the route is much longer than the RTT of the wireless part, the time it takes for the BS to buffer the required data is enough for the TCP of the last hop to recover from any packet losses. This observation is not true only for the 30% PER case, in which it is evident that the last hop TCP cannot serve the incoming data, delaying the overall transfer time. In such a case, data could be dropped at the BS if the buffer capacity is depleted. The enhanced performance achieved by a solution along the lines of Indirect-TCP (i.e. isolation of the wireless error in the short-delay part of the transfer route) comes naturally as an additional benefit when using DTN.

The energy consumption chart in Figure 8-5 (lines E2E and Split) shows that up to an error rate of 10% the energy savings achieved by the Split approach are approximately 30%. As the error rate increases so does the conserved energy, reaching as much as 78% for an error rate of 30%. At this error rate the energy required for the completion of the data transfer in the E2E case is 861 J, while in the Split case the required energy is 186 J.

Figure 8-5 contains two additional lines representing the potential energy expenditure of the E2E and the Split cases (labeled as P-E2E and P-Split), based on experimental data from our original work on the subject [16]. As expected, the P-E2E value (i.e. the value for the E2E case less the potential for switching to sleep mode in long idle intervals) is considerably lower than the E2E case, where no energy saving mechanism is employed, and higher than both the Split and P-Split values, where buffering at the BS takes place. Finally, the difference between the Split and the P-Split values falls between 7-10% for all error rates, showing that the rendezvous mechanism exploits 90-93% of the maximum theoretical energy saving amount. The observed difference consists of long idle intervals that cannot be used by the rendezvous mechanism for switching to sleep mode, as they occur during data transmissions (i.e. as opposed to occurring between the finish of a transfer and the next rendezvous). Nevertheless, the rendezvous mechanism takes advantage of the vast majority of the exploitable idle intervals, rendering it a very effective energy saving approach.

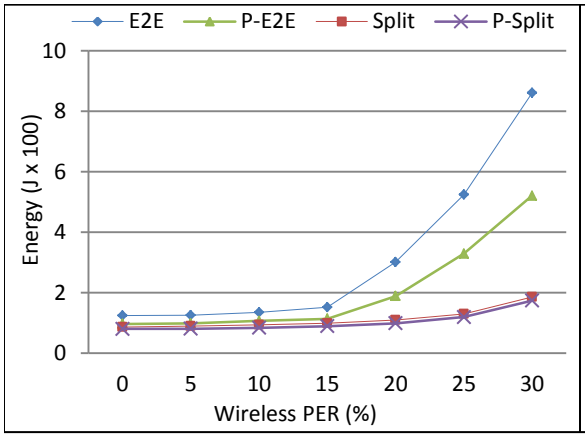

Figure 8-5: FTP, Varying Wireless PER, No Competing Flow Energy Consumption.

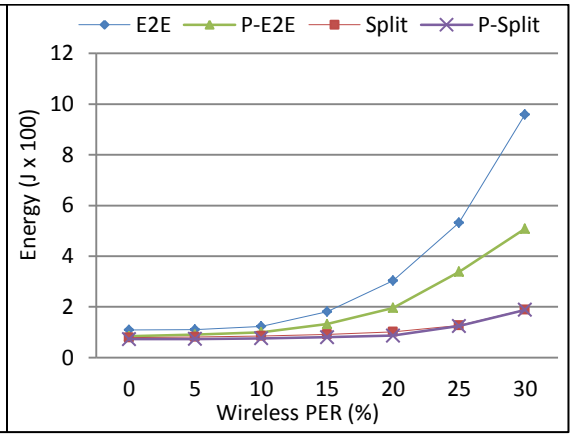

Figure 8-6: FTP, Varying Wireless PER, Competing Flow, Energy Consumption.

The experiments in the previous section were repeated with a competing flow present on the network. Because of the huge transfer times that would otherwise be necessary, the bandwidth of the backbone link was set to 2 Mbps (as opposed to the 1 Mbps value that was used in the previous section). Even though we cannot directly compare the resulting figures for the cases with or without a competing flow, we can still draw meaningful conclusions for the relative performance of the E2E vs. the Split case. The most significant observation in this scenario as depicted in Figure 8-6 (lines E2E and Split) is that when a competing flow is present, the energy expenditure of the E2E approach exhibits higher increase rates for smaller error rates. For higher error rates both the competing and the non-competing flow cases exhibit

similar behavior. Comparing the Split and P-Split lines we observe that the difference is even smaller than in the previous set of experiments (when no competing flow is present), and therefore we can conclude that under heavily strained network conditions, the rendezvous mechanism exploits virtually all of the available energy saving potential.

Varying Target Buffer Occupancy FTP Transfer

In this set of experimental runs the delay is set to 300ms, the bandwidth of the backbone link is set to 1Mbps (rendering it as the bottleneck link of the transfer route) and the TBO varies from 0 up to 180 KB. For the 0 buffering case we used the E2E approach, whereas for the rest of the runs we used the Split approach with the corresponding TBO. The transfer duration is, reasonably, identical for all buffer sizes, and thus, not reported. The energy expenditure, however, drops significantly as the TBO increases (Figure 8-7, No-CF line). Since the wireless link of the transfer route is consistently underutilized, buffering more data allows the WNIC of the receiver to switch to the sleep state for longer periods, while at the same time not getting penalized for waiting. Increasing the TBO beyond a certain value could cause congestion problems on the WLAN and/or prolong the transfer duration. Such problems could become more severe in case multiple mobile clients are present, simultaneously downloading data using the Split approach. With the increase in the buffering capacity, the traffic on the WLAN becomes burstier and, consequently, more prone to congestion. The size of the buffer, in case DTN is employed, can be thought of as the size of the bundle size that the BS would forward to the last hop of the transfer connection.

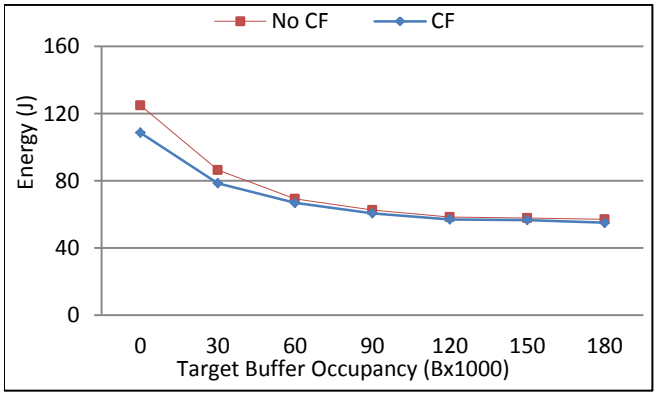

Figure 8-7: FTP, Varying Target Buffer, FTP, With and Without a Competing Flow, Energy Consumption.

Target	Mean	Max
30	30.70	125.96
60	57.98	131.40
90	84.30	147.46
120	109.15	186.88
150	132.67	219.00
180	153.99	271.56

Table 8-3: No Competing Flow, Buffer Occupancy Values (KB)

The experiments of the previous scenario were repeated with the introduction of cross traffic caused by a competing flow. In order to compensate for the higher network traffic, the bandwidth of the backbone link was increased to 2Mbps. In both sets of experiments the transfer duration remains constant for all runs. With respect to the energy expenditure, shown in Figure 8-7 (CF line), it is observed that while for small TBO values the amount of energy required by the no competing flow scenario is significantly higher, as the TBO increases the two values converge to a common joules value in the mid 50s. This shows that

increasing the buffer size can alleviate differences in the energy expenditure, caused by varying congestion conditions in the wired part of the network.

Table 8-3 reports on the actual mean and max buffer occupancy values for each target occupancy value that was used in the experiments. Reporting on the actual buffer occupancy is of great significance in evaluating our approach because it reveals the effectiveness of the rendezvous mechanism in predicting the incoming data flow, and also allows the administrator to more accurately plan buffer allocation. The mean occupancy is, almost always (with the exception of the 30 KB case), smaller than the target occupancy, ranging from 3% - 14%. As the TBO increases, so does the lag of the mean occupancy, showing that the mechanism exhibits slightly conservative behavior. The max occupancy starts at almost 10 times the mean occupancy for the 30 KB case and drops to less than double for values of 90 KB and higher. Even though the max occupancy value can be quite high, when multiple mobile nodes are present the collective buffer occupancy at the BS will be highly predictable, due to the statistical characteristics of the buffering process.

In order to better grasp the workings of the rendezvous mechanism, we present in Figure 8-8 the fluctuations of the buffer occupancy throughout the data transfer for the 60 KB target buffer occupancy case. The chart can be roughly split into two areas: 0-60 and 60-120 seconds. In the beginning of the transfer TCP enters the slow start phase in order to probe for available bandwidth. The slow start phase is also entered at around 60 sec due to packet losses and timeouts. It becomes apparent from the chart that the greatest buffer occupancy values occur during the slow start phase, in the beginning of each of the two areas. The reason for the intense fluctuations at around 0 and 60 seconds is that during the slow start phase, the connection data flow increases rapidly and the rendezvous mechanism takes a few seconds to adjust. However, for the rest of both areas the actual buffer occupancy lies well with the 40 – 80 KB area, only 33% around the TBO value. Again, this chart gives more evidence that in case of a multitude of mobile devices being serviced by the BS at the same time the buffer occupancy would be highly predictable, allowing for adequate buffer size allocation while avoiding underutilization. It also shows that there is room for making the rendezvous mechanism more intelligent so that it responds better to rapid data flow fluctuations, possibly by taking into account historical data.

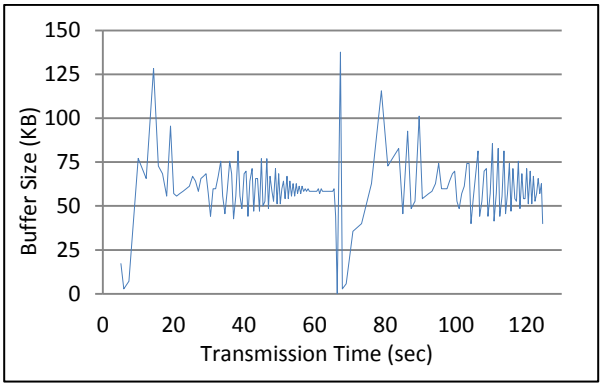

Figure 8-8: Actual Buffer Occupancy at 60 KB Target Buffer Occupancy.

Varying Target Buffer Occupancy CBR Transfer

In this set of experiments a CBR traffic flow of 10 MB is considered in a network with no contending traffic. As in most of the previous scenarios, the delay of the backbone link was set to 300 ms and the bandwidth to 2 Mbps. The transfer duration does not convey any important information, as it merely refers to the time the last packet is received by the mobile node and it depends solely on the end-to-end network delay.

This scenario can be mostly considered as a preliminary testing with respect to streaming data traffic, and is further studied in upcoming sections. When measuring the service quality of a file transfer the time for completion can be an adequate metric. The sooner the file transfer finishes the better it is for the end user. In all the experiments, so far, the transfer duration was not affected by the buffering mechanism, rendering the mechanism acceptable from a user standpoint. However, CBR traffic is, in most cases, related to multimedia streaming. Extra delays introduced by the buffering mechanism can lead to unacceptable service quality, a factor which is not captured by the experimental setup used in this section (see section 8.1.3). Nevertheless, the experiments in this scenario reveal the potential a DTN-based Split approach can have for reducing the energy consumption for CBR traffic.

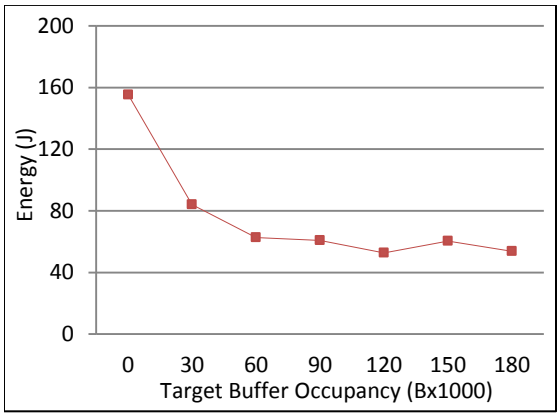

Figure 8-9: CBR, Varying Buffering Threshold Energy Consumption.

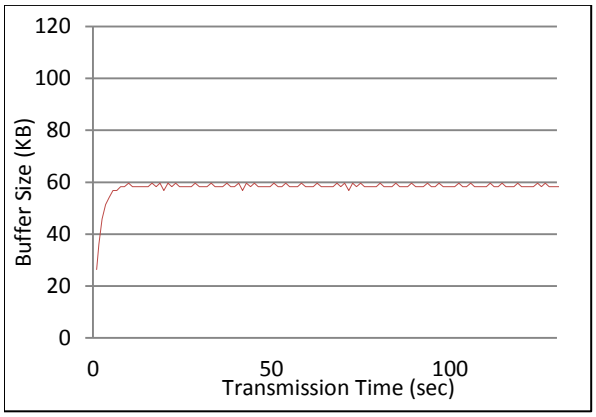

Figure 8-10: Actual Buffer Occupancy at 60 KB Target Buffer Occupancy.

It becomes apparent from the chart in Figure 8-9 that even for a TBO of 30 KB, the energy saving achieved by the rendezvous mechanism is 46% compared to the case where no power saving is used (0 TBO). As the TBO increases, the conserved energy increases as well, reaching almost 66% (for a TBO of 120 KB) of the energy spent in the no-buffering case. For the current experimental settings it seems that setting the TBO to more than 60 KB does not significantly improve the energy efficiency, even though there is a general such trend along the horizontal axis of the chart.

Figure 8-10 depicts the actual buffer occupancy achieved by the rendezvous mechanism in the 60 KB TBO case. As opposed to the corresponding chart for the TCP connection of the previous section, the actual buffer occupancy in this case is almost constant, since in a CBR connection the incoming data flow is

constant as well. It should be noted that large buffering in multimedia streams can cause long delays and, thus, unacceptable user experience; however, if used with caution, the rendezvous mechanism can deliver significant energy conservation with little or no quality degradation. A more comprehensive discussion about this issue is included in section 8.1.3.

8.1.2. Stage 2 (Full DTN Overlay)

In this section we present the results of simulation experiments involving the deployment of the DTN agent on the topology described in section 6.1.3. The experiments were conducted as part of four scenarios, each designed to highlight certain aspects of DTN combined with our proposed energy-saving, rendezvous mechanism. The four scenarios are: Single-bundle, varied bandwidth (the bottleneck bandwidth is varied on a single-bundle ADU transfer), Single-bundle, varied Target Buffer Occupancy (the TBO is varied on a single-bundle ADU transfer), Multi-bundle, varied bandwidth (the bottleneck bandwidth is varied on a transfer consisting of multiple fragments of the original ADU), and Multi-bundle, varied DTN storage space (the available DTN storage space is varied on a multi-bundle transfer). Through the single-bundle scenarios we intended to evaluate the energy-efficiency and throughput of our proposed rendezvous scheme. The multi-bundle scenarios, on the other hand, allowed us to demonstrate the DTN agent functionality for load balancing traffic based on storage space availability and, emphasize the capability of DTN to improve network performance by uniformly distributing data storage over alternative paths.

Single-bundle Transfer with Varied Bandwidth

The first set of experiments presented in this subsection demonstrate the capability of the rendezvous mechanism to achieve major energy conservation without sacrificing network performance, and the inherent ability of a DTN overlay to improve network performance, when deployed on top of a congested IP network. The experiments in this section consist of an end-to-end TCP transfer from Src to MR (TCP-E2E), a DTN transfer where the DTN agent is deployed on Src, BS and MR (DTN-2Hop), and a DTN transfer where the DTN agent is deployed on Src, Route1-DTN, BS and MR (DTN-3Hop). In both DTN-2Hop and DTN-3Hop the data transfer is conducted as a single bundle (no bundle fragmentation takes place at the source), the DTN storage is unlimited and the retransmission mechanism is disabled. Bundle fragmentation of a dynamic nature takes place only as part of the rendezvous mechanism at the BS (see section 3.3). In the DTN-3Hop scenario Route1-DTN routes the DTN bundle in a cut-through fashion by maintaining one incoming TCP connection with Src and one outgoing TCP connection with BS. All data transfers follow the main path (i.e. Route2-DTN and Route3-DTN remain inactive). Results in this set of experiments are reported for a bandwidth of the bottleneck links that ranges from 1 Mbps -4 Mbps.

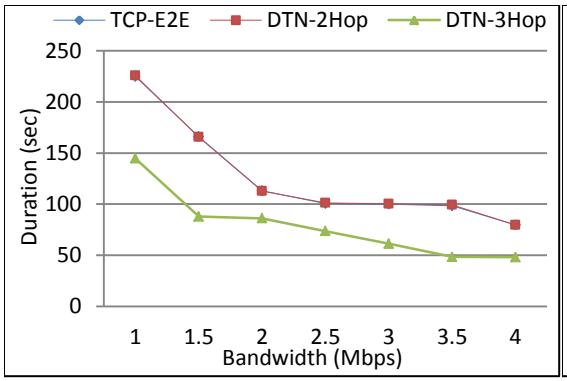

Figure 8-11: Duration of a Single-bundle Transfer with Varied Bandwidth.

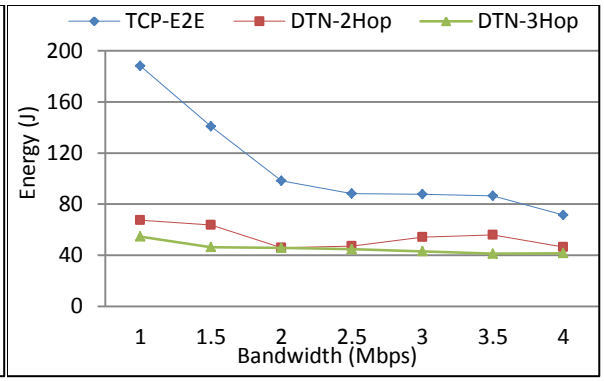

Figure 8-12: Energy Expenditure of a Single-bundle Transfer with Varied Bandwidth.

The chart in Figure 8-11 depicts the transfer duration for the three experimental settings in the single-bundle, varied bandwidth scenario. The results for TCP-E2E and DTN-2Hop are identical for all values of the bottleneck bandwidth, showing that the rendezvous mechanism does not hamper the throughput of the data transfer. In the DTN-3Hop case, however, the transfer duration is significantly shorter than the duration of the other two cases by a percentage ranging from 24% to 50%. The extra DTN node that is present along the network path in the DTN-3Hop case essentially splits the TCP connection between the Src and the BS into two TCP connections of half the Round Trip Time, achieving better bandwidth utilization and claiming a larger share of the bottleneck links capacity from the cross-traffic flows. Low RTT TCP connections have a quicker response to changes in the network conditions and are more effective in competing with contending traffic.

The chart in Figure 8-12 shows the energy consumption for the three experimental settings as a function of the bottleneck bandwidth values. It is evident that the energy efficiency is greatly enhanced when the rendezvous mechanism switches the wireless interface of the MR to the sleep mode (DTN-2Hop and DTN-3Hop cases), as opposed to not when the rendezvous mechanism is not employed (TCP-E2E). Generally, for bottleneck links of smaller capacity the gains from employing the rendezvous mechanism are higher, reaching the value of approximately 70% for the 1 Mbps bandwidth case. For higher bandwidth values the rendezvous mechanism increases energy efficiency by the smaller, albeit still very significant, value of 40%. In the TCP-E2E case, long transfers lead to considerably higher energy expenditure, since all the extra time is spent by the wireless interface in the idle state (i.e. a high-power state). On the contrary, when the rendezvous mechanism is used, long transfers do not significantly increase the energy expenditure, since most of the extra time is spent by the wireless interface in the sleep state (i.e. a low-power state). Finally, the DTN-3Hop is slightly more energy-efficient than its DTN-2Hop counterpart under all bottleneck capacities due to the lower transfer time and the better convergence achieved the rendezvous algorithm, caused by the smaller RTT value of the TCP connection to the BS (i.e. increased ability to exploit idle intervals).

Single-bundle, Varied Target Buffer Occupancy Transfer

The purpose of the second set of experiments with a full DTN overlay is to study the transfer duration, the energy consumption at the mobile receiver and the actual buffer occupancy at the BS with respect to the TBO of the rendezvous mechanism. The DTN agent is deployed on the Src, BS and MR as in the DTN-2Hop scenario of the single-bundle experiments. In this set of experiments the TBO ranges from 0 to 70 KB and the bandwidth of the bottleneck links is set to 2.5 Mbps (the median of the bandwidth values tested in previous experiments, which leads to mild network congestion).

The overall transfer duration was, expectedly, identical for all tested TBO values, and, thus, the corresponding chart was omitted. In the chart of Figure 8-13 the energy consumed for completing the data transfer is plotted against the TBO. In the 0 TBO case, the rendezvous mechanism is not employed and, therefore, no energy conservation takes place (i.e. the wireless interface of the MR remains in the active state for the entire duration of the transfer). Enabling the rendezvous mechanism even with a small TBO of 10 KB immediately drops the consumed energy by approximately 20%. The consumed energy continues to decrease until the 40 KB value for the TBO, where it is reduced by more than 50% with respect to the 0 KB case (88 vs 42 J). Beyond the 40 KB TBO value the energy consumption stabilizes within a range of 40-43 J. Small TBO values lead to the frequent appearance of short rendezvous intervals that cannot be exploited by the MR. By the time the trailing bundle of a data burst with a short rendezvous time is received at the MR, the rendezvous time may already be in the past, thus, not allowing the MR to switch to sleep mode. Additionally, every switch to the sleep state and back to active involves 20 ms of transition time requiring power equal to that of the idle state and, therefore, it is generally more energy efficient to have fewer rendezvous intervals of longer duration.

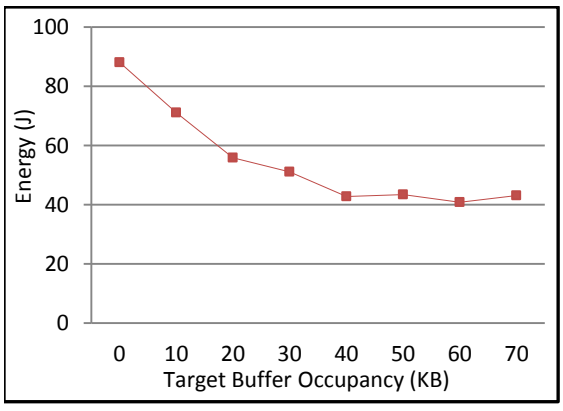

Figure 8-13: Single-bundle Transfer with Varied Target Buffer Occupancy, Energy Expenditure.

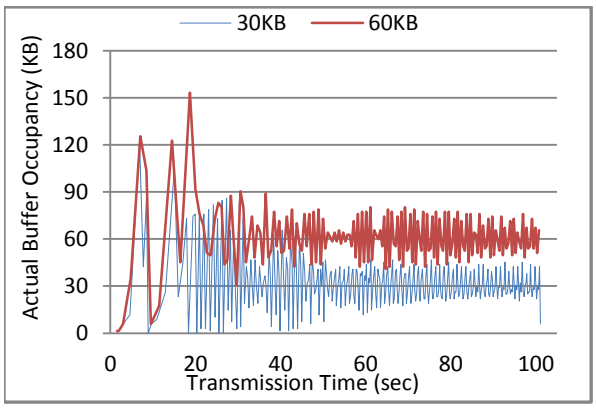

Figure 8-14: Single-bundle Transfer with TBO of 30 and 60 KB, Actual Buffer Occupancy.

Figure 8-14 depicts the amount of data flushed at each rendezvous for the entire data transfer in the 30 KB and the 60 KB TBO cases. The amount of flushed data for both TBOs deviates greatly from the desired TBO in the beginning of the transfer due to the sharp increase in the flow rate of TCP during the slow start

phase and the lack of initial measurements (i.e. the initial rendezvous time is arbitrarily set). When the incoming flow rate at the BS stabilizes the actual buffer occupancy oscillates within an area of approximately 15 KB and 20 KB plus or minus the TBO for the 30 KB and the 60 KB cases respectively. Despite the oscillations, the average actual buffer occupancy over time converges to the desired TBO (the relative chart was omitted for brevity), so due to the statistical nature of the incoming traffic, the buffer space that must be allocated at the BS will be equal to the number of flows times the TBO plus some safety margin. The rendezvous algorithm could be improved in order to eliminate the initial high surges as well as narrow the range within which the actual buffer occupancy oscillates.

Figure 8-15 shows the state transitions at the wireless interface of the mobile node for the 30 KB and the 60 KB TBO cases during 7 arbitrary seconds in the steady state of the data transfers. The chart demonstrates how setting a larger TBO value produces a smaller number of longer rendezvous intervals and vice versa. The tradeoff of a large TBO is the additional delay introduced by the data buffering, which may cause problems to real-time applications. Studying the application of the rendezvous mechanism on real-time traffic applications is the focus of the next section (8.1.3).

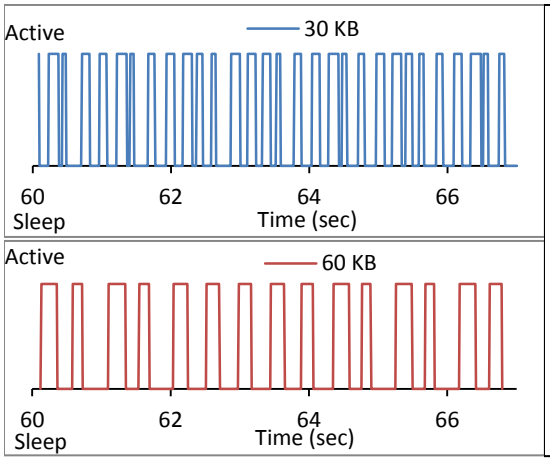

Figure 8-15: Single-bundle Transfer with TBO of 30 and 60 KB, Sleep/Active state transitions.

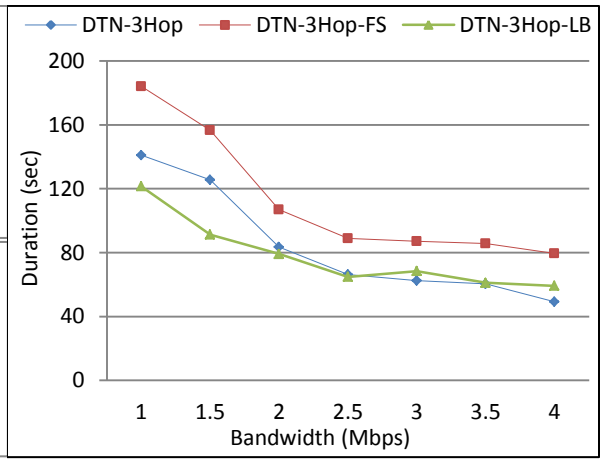

Figure 8-16: Multi-bundle Transfer with Varied Bandwidth, Duration.

Multi-bundle, Varied Bandwidth Transfer

The third set of experiments consists of three variations of a DTN overlay deployment, designed to evaluate the effects of load balancing and bundle storage limitations at the DTN layer in a multi-bundle setting. All instances of the DTN agent deployed in these experiments use a maximum bundle size of 30,000 B at the source node and, therefore, the 10 MB file is fragmented into 350 bundles. The experimental settings include a deployment of the DTN agent on Src, Route1-DTN, BS-DTN and MR-DTN with infinite storage space (DTN-3Hop), a deployment of the DTN agent identical to the previous setting, but with a finite storage of 90,000 B (i.e. 3 bundles) at the Route1-DTN node (DTN-3Hop-FS), and a scenario in which the DTN agent is additionally deployed on the Route2-DTN and Route3-DTN nodes,

receiving along with Route1-DTN bundles from Src in a round-robin fashion and load balancing network traffic (DTN-3Hop-LB). In the latter scenario, the storage space of each of the Route1-DTN, Route2-DTN and Route3-DTN is 90,000 B and, thus, the overall storage space is thrice that of the DTN-3Hop-FS setting, whereas the total bandwidth remains at 3 Mbps.

DTN-3Hop is identical to the corresponding setting of section 6.1 and is used as a benchmark for the rest of the simulations. The DTN-3Hop-FS setting serves the purpose of demonstrating how limited storage space at an intermediate DTN node affects throughput and, finally, the DTN-3Hop-LB setting shows how load-balancing over multiple intermediate DTN nodes, exploiting available network storage, ameliorates the effects of storage limitations. The retransmission timeout for the DTN-3Hop-FS and the DTN-3Hop-LB scenarios is set to a constant value of 2 seconds. When a bundle cannot be forwarded from Src to the next DTN node due to storage limitations, the Src makes a new attempt every 2 seconds. The retransmission timeout is not relevant in the infinite storage scenario (DTN-3Hop) since bundles are not rejected. As in the single-bundle case, the results in this set of experiments are reported for a bandwidth of the bottleneck links that ranges from 1 Mbps -4 Mbps.

The chart in Figure 8-16 presents the transfer duration for the three experimental setups. In the DTN-3Hop-FS transfer, limited storage space at the DTN layer causes the rejection of incoming bundles at the Route1-DTN node, which, in turn, results in a periodical starvation of the storage space, prolonging the transfer duration for all bottleneck capacities tested. The limited storage space in the DTN-3Hop-FS case introduces a time overhead ranging from 25% to 61% with respect to the DTN-3Hop infinite storage space case. The general trend is that the time overhead percentage is larger for higher bandwidth values, since a faster data transfer is more likely to starve the storage of the intermediate cushion node. For the highly congested cases where the bottleneck bandwidth is 1 and 1.5 Mbps, the DTN-3Hop-LB transfer (where routing is employed) outperforms the single-route, infinite-storage, DTN-3Hop transfer, due to receiving an increased share of the bottleneck links. Since three TCP flows compete with the TCP cross-traffic, they claim a larger share of the available bottleneck link. For higher bottleneck bandwidth values the performance difference between DTN-3Hop and DTN-3Hop-LB is insignificant. The lower transfer durations of the DTN-3Hop-LB case versus the DTN-3Hop-FS case show how load balancing among DTN nodes with the same amount of storage space on different network paths can improve performance, even when the collective bandwidth remains the same. Finally, we can deduce that the overall 3 x 90,000 B (i.e. 9 bundles) storage space of the DTN-3Hop-LB case is large enough to ensure that there will be a continuous flow from the relay nodes to the BS and that the intermediate cushion node storage will not starve, since the high bandwidth cases are not penalized for the limited storage.

The energy expenditure results (the corresponding chart is omitted for brevity) show that even though there is a general trend for lower energy consumption when the network congestion is milder, no clear correlation between the available bandwidth and the energy expenditure can be deduced. Further experimentation would be necessary in order to draw safe conclusions about the behavior of the rendezvous mechanism when the incoming flow rate at the BS is highly variable.

Multi-bundle, Varied DTN Storage Space

The final set of experiments in the full DTN overlay was designed for the purpose of evaluating the effects of varying the available storage space at the relay DTN nodes on the performance of a 3-hop, DTN file transfer. The DTN agent is deployed in the same overlay as in the DTN-3Hop-FS and the DTN-3Hop-LB experimental settings of the previous section (i.e. the single route and the load balancing cases) and the available storage space is varied from 30,000 – 150,000 B (1-5 bundles). The bandwidth of the bottleneck links is set to 2.5 Mbps.

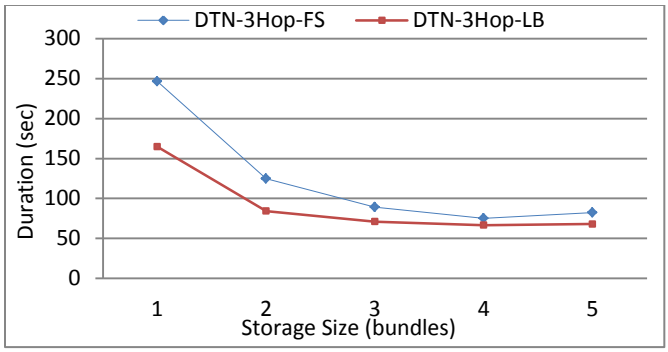

Figure 8-17: Multi-bundle Transfer, Varied DTN Storage Space, Duration.

The chart in Figure 8-17 depicts the transfer duration for the corresponding storage space size in both settings. As mentioned in the multi-bundle, varied bandwidth section the overall DTN storage space in the 3Hop-LB case is three times that of the corresponding 3Hop-FS case. For the smallest possible storage size of 1 bundle the duration in the load-balancing setting is approximately 33% less than that of the single-route setting (164 vs. 246 sec). When the storage size is increased to 2 bundles the transfer duration in both cases is halved, while their ratio remains intact (84 vs. 124 sec). Having only a single available slot for accepting new bundles severely affects the protocol performance; the relaying nodes cannot accept a new bundle unless the BS (next hop node) has accepted custody for the previous bundle, essentially interrupting the data transfer between consecutive bundles. As long as at least 2 bundle slots are available at the relay DTN node(s) the transfer duration is reduced to reasonable levels, reaching a minimum for the value of 4 slots in both cases. For values beyond the 3 slots threshold, the difference in the performance of the two settings can be attributed to the slightly increased share of the bottleneck links that is claimed by the three TCP connections of the 3Hop-LB case as compared to the single TCP connection of the 3Hop-FS case.

8.1.3. Stage 3 (Simple DTN Overlay)

In this section we present the simulation results of the simple DTN overlay stage described in section 6.1.4. The results are organized in three subsections according to the type of the main data transfer, such as: FTP, CBR and HTTP. All types of traffic are first simulated in the absence of competing traffic and, subsequently, with a CBR and an FTP competing flow. In each subsection we first report on the energy and

delay tradeoff and the corresponding sleep and transition interval information for various TBO values, when no competing flow is present. Next, we report on the delay and energy expenditure of the main data transfer under all three competing traffic possibilities (competition free, FTP competing flow, CBR competing flow). All experiments include the End-to-End (E2E) case, where no buffering takes place, as the reference scenario.

FTP Data Transfer

The chart in Figure 8-18 depicts the energy consumption and the delay for the completion of a 10 MB FTP data transfer. The first significant observation is that, as shown in the simulation results of the proof-of-concept (section 8.1.1) and the full DTN overlay deployments (section 8.1.2), the additional delay imposed by the data buffering at the BS is negligible. In the E2E case 59.7 seconds are required in order for the entire file to be delivered to its destination, whereas 60.4 seconds are required in the maximum tested TBO case of 150 KB. That is, the additional delay for buffering 150 KB is only 0.7 seconds or 1.2% of the overall transfer duration. The energy line on the same chart shows that the consumption drops from 55.5 J in the E2E case to a minimum of 35.6 J for the 100 KB TBO case, yielding a reduction of approximately 36%. These results confirm the results of the previous experiments and show that, for relatively large file transfers, substantial energy conservation can be achieved at a negligible performance cost.

TBO	Sleep Count	Sleep Time	Trans Time
E2E	0	0	0
10	232	2.39	4.62
20	521	14.93	10.36
30	348	19.93	6.9
40	262	22.47	5.18
50	212	23.04	4.16
60	176	22.74	3.48
80	134	26.42	2.62
100	108	27.19	2.1
150	69	26.48	1.32

Table 8-4: FTP with no competing traffic, sleep and transition information.

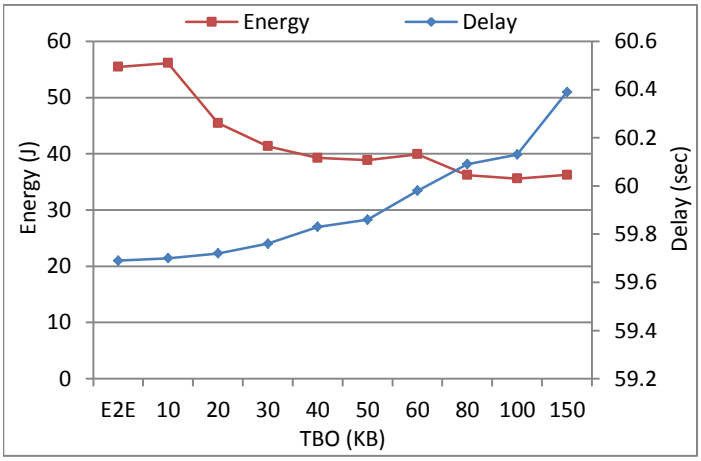

Figure 8-18: FTP with no competing traffic energy and delay vs. the TBO.

Table 8-4 includes information on the number of sleep intervals and the total sleep time and transition time for all TBO values. For the smallest TBO of 10 KB the overall idle time (sum of the sleep and transition time durations) is only 7 seconds. Due to the small TBO, the interval until the next rendezvous time is, in most cases, too short and does not allow enough time for transitioning to and fro the sleep state (the process is described in detail in section 3.2). For TBO values of 20 KB and larger, the overall idle time is consistently higher than 25 seconds. However, the actual sleep time achieved is lower for small TBO

values due to the high number of sleep intervals and, consequently, the increased time spent in transitioning between the active and sleep states. Maximum sleep time is achieved for the 100 KB case, which also yields the lowest energy consumption as shown in the energy chart.

The duration of the FTP transfer under all types of competing traffic is shown in Table 8-5. The transfer duration for the case where no competing flow is present is virtually identical with the case containing a CBR competing flow for almost all TBO values (with the exception of the 10 KB TBO). On the contrary, when the competing traffic consists of a TCP flow, the duration of the main transfer is approximately double that of the other two cases. The above observation signifies that the CBR competing flow is such that the bottleneck link remains the same as in the case where no competing flow is present (i.e. link Src-DTN – Rel-IP as described in section 6.1.4).

TBO	Delay No Flow	Delay CBR Flow	Delay TCP Flow
E2E	59.69	59.69	123.36
10	59.7	63.33	122.95
20	59.72	59.72	120.12
30	59.76	59.76	120.15
40	59.83	59.84	120.17
50	59.86	59.85	123.11
60	59.98	59.98	120.17
80	60.09	60.09	117.42
100	60.13	60.13	121.04
150	60.39	60.38	116.58

Table 8-5: FTP delay with all types of competing traffic.

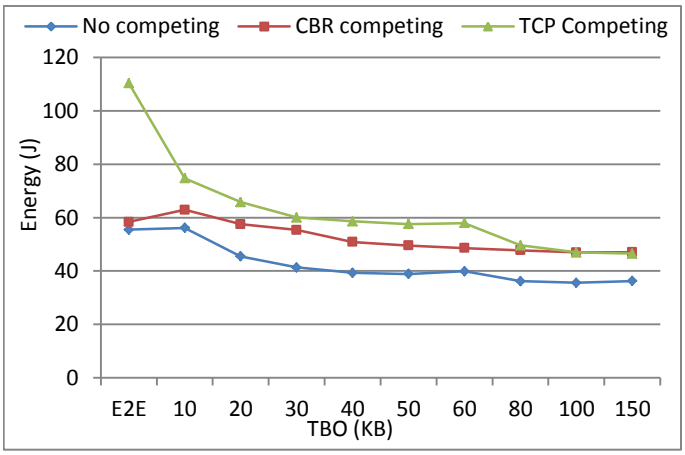

Figure 8-19: FTP energy expenditure with all types of competing traffic.

Figure 8-19 depicts the energy consumption under all three competing traffic possibilities. Despite that the transfer delay is the same for the no competing and the CBR competing traffic cases, the energy requirements of the CBR competing traffic case are higher. This is due to the limited portion of the WLAN bandwidth that is available to the main transfer that leads to longer buffer flush durations and, consequently, shorter overall sleep time. The same applies when the competing traffic consists of a TCP flow. In this case, however, the possible merits of the rendezvous mechanism are much higher. Contrary to the CBR case, the TCP competing flow limits the bottleneck link bandwidth and creates additional potential for energy conservation. The energy consumption chart shows that the energy expenditure starts at 110.44 J in the E2E case and drops to 47 J for a 100 KB TBO, a conservation of 57%. The above result is in-line with equation 3-3, which dictates that the sleep time for the transfer, and, therefore the achieved energy conservation, increases with the increase of the ratio $\frac{OutRate-InRate}{InRate*OutRate}$. Even though the reduction of OutRate and InRate, in case competing flows are present, is the same maintaining their difference at the same value, the denominator of the $\frac{OutRate-InRate}{InRate*OutRate}$ ratio decreases, and so the sleep time increases as well.

A final observation, with respect to the TCP competing flow case, is that larger values of the TBO reduce the main transfer energy consumption, which converges to the values of the CBR-competing traffic case. This is due to the fact that larger flush intervals allow the main flow to claim more bandwidth and force the competing TCP to slow-down, increasing the sleep duration.

In the presence of competing traffic on the wireless network, the energy consumption of the main transfer increases for an additional reason, besides the limited bandwidth availability described in the previous paragraphs. During idle intervals that cannot be exploited by the rendezvous mechanism for switching to the sleep state, the MR needs to listen to frames destined to the other mobile host (MCT), which is receiving the competing traffic. Since the *receive* state is more power-consuming than the *idle* state this leads to increased energy expenditure. This holds true both for the CBR and the TCP competing flow cases. In all cases, the overall bottleneck remains on the wired part of the network (except during buffer flushes).

In terms of the user experience, file transfers are only marginally affected by the deployed DTN overlay and the buffering at the BS introduced by the rendezvous mechanism. The TBO value is usually much smaller than the application ADU (i.e. the transferred file) and so the additional delay is negligible. Therefore, the proposed energy-efficient solution can be unconditionally applied to file transfer applications without deteriorating the experience of the end-user.

CBR Data Transfer

Figure 8-20 includes the energy consumption vs. the average delay of delivered datagrams for a CBR flow producing a 500 Byte-datagram every 5 ms (800 Kbps) for 1 minute, in the absence of competing traffic. It is apparent that even for the smallest tested TBO value of 10 KB the energy saving is substantial. Specifically, the energy consumption drops from 54.14 J in the E2E case to 36.15 in the 10 KB TBO case, achieving conservation of 33%. The corresponding average delay values for these experiments are 0.2 sec for the E2E and 0.27 sec for the 10 KB cases respectively; an increase in the order of 26%. That is, 33% energy conservation is achieved at the expense of an average of 26% delay increase of the delivered datagrams. In the 30 KB case the energy expenditure drops to 30.32 J (44% reduction), while the average delay is doubled with respect to the E2E case (0.41 sec). The energy conservation reaches a maximum of 48% for the 100 and 150 TBO cases, when the delay is 4.4 and 6 times that of the E2E case respectively.

The main observation, as compared to the FTP transfer simulations presented in the preceding subsection, is that significant energy savings are achieved even for a minimal TBO of 10 KB (in the FTP transfer a TBO of at least 20-30 KB was necessary). At 800 Kbps the simulated CBR flow rate is lower than the 1.5 Mbps bandwidth of the bottleneck link, achieved by the TCP connection and, therefore, the energy conservation potential is higher. Additionally, the CBR flow is paced in a more tactical manner, allowing the rendezvous mechanism to flush almost uniform data amounts (see Figure 8-10), achieving more efficient utilization of the wireless network. On the contrary, TCP's transmission control may

abruptly adjust transmission rate forcing the rendezvous mechanism into temporary fluctuations (see Figure 8-8). During these fluctuations the amount of flushed data may either be too small or too large compared to the TBO, producing intervals of extremely low or extremely high load in the wireless network, limiting the effectiveness of the rendezvous mechanism.

TBO	Sleep Count	Sleep Time	Trans Time
E2E	0	0	0
10	587	24.15	11.64
20	297	30	5.84
30	199	31.98	3.9
40	153	32.92	2.94
50	123	33.26	2.34
60	100	33.93	1.96
80	77	34.44	1.48
100	61	34.94	1.2
150	45	35.44	0.82

Table 8-6: CBR with no competing traffic, sleep and transition information.

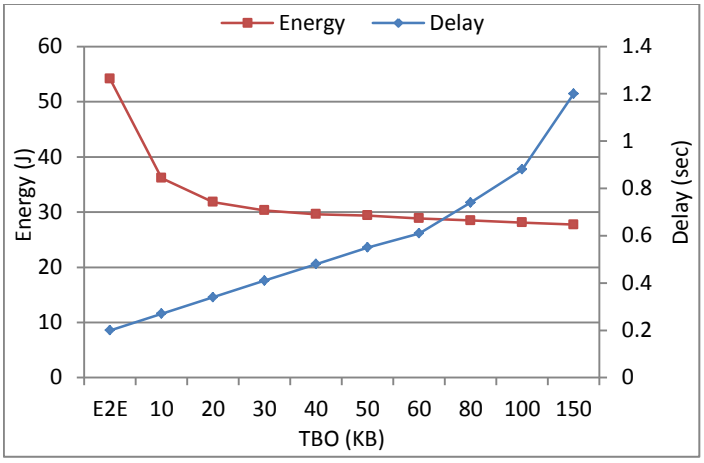

Figure 8-20: CBR with no competing traffic energy and delay vs. the TBO.

The information displayed in Table 8-6 shows how the overall idle time (sum of sleep and transition times) remains almost identical (35-36 sec) for all TBO values. The additional energy conservation achieved for higher values of the TBO results from the decrease in the overall number of sleep intervals and the consequent reduction of the time spend in the energy-consuming transition state.

TBO	Delay No Flow	Delay CBR Flow	Delay TCP Flow
E2E	0.2	0.21	0.43
10	0.27	0.28	0.48
20	0.34	0.34	0.52
30	0.41	0.41	0.56
40	0.48	0.48	0.58
50	0.55	0.55	0.6
60	0.61	0.62	0.65
80	0.74	0.75	0.76
100	0.88	0.89	0.89
150	1.2	1.21	1.22

Table 8-7: CBR delay with all types of competing traffic.

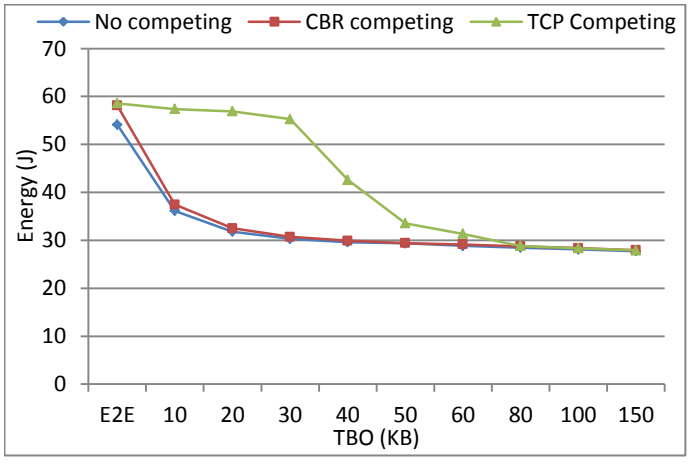

Figure 8-21: CBR energy expenditure with all types of competing traffic.

Table 8-7 contains information on the average delivery latency for the stream datagrams measured under all types of competing traffic. The additional delay introduced by the presence of a CBR competing flow appears to be negligible, both in the E2E reference case and under all TBO settings. The energy

consumption for these two cases is also virtually identical (Figure 8-21). However, in the presence of a TCP competing flow, the additional delay for the E2E case as well as for relatively small TBO values is substantial. The additional delay introduced by the TCP flow starts at almost double that of the E2E case (0.2 vs. 0.43 sec) and converges to the other two competing traffic possibilities for TBOs above 60 KB. As the TBO increases, larger buffer flushes suppress the competing TCP on the wireless hop and achieve shorter delivery time.

Similar behavior is observed in Figure 8-21 with respect to the energy consumption. The CBR competing flow does not substantially affect the energy consumption, which is only marginally higher for small TBO values. However, in the presence of TCP competing traffic, the buffering amount (expressed by the TBO value) needs to be much higher so that significant energy savings can be achieved. For small TBO values the competing TCP flow takes up most of the available bandwidth, essentially suffocating the streaming flow. Additionally, the MR during active intervals (buffer flushes) needs to receive a high amount of data destined to the competing mobile node, wasting energy. For larger TBO values, larger buffer flushes contain the competing TCP claiming a larger share of the available wireless bandwidth and achieving higher energy efficiency.

It must be noted that certain UDP datagrams belonging to the main CBR transfer may be discarded by the network due to buffer overflow. These datagrams are not accounted for in any way during the simulations. However, during preliminary experimentation it the simulation parameters was selected such that generally the network was not overloaded and a good portion of the datagrams of the streaming flow was successfully delivered to the destination. In the context of this work towards a PhD degree simulations reporting on the quality of a received streaming flow remains an open issue.

The simulation results have shown that the energy-efficient DTN overlay creates significant energy conservation potential for CBR streaming flows. However, this potential comes at the expense of significant delay increase at the datagram level (i.e. the ADU of the streaming application). Since the TBO is, generally, larger than the stream datagram, buffering introduces substantial delays. Nonetheless, streaming applications involve client buffering so the effect of incoming data rate fluctuations on the end-user experience is limited. Depending on the timing requirements of a certain streaming application, the additional buffering at the BS could be compensated in a straightforward way by a small increase in the client buffering amount. Furthermore, the last-hop wireless link can be almost always guaranteed to be congestion-free, so the extra delay will be mostly translated to a small delay in the reception of the entire stream. For the 30 KB TBO, where the consumed energy is almost halved, the additional delay was measured at 200 ms, an interval that is hardly noticeable by the end-user. Therefore, our proposed solution can be applied for most streaming application without any adverse effect on the end-user experience.

HTTP Data Transfer

The chart contained in Figure 8-22 depicts the energy consumption and the ADU delay for a 160-second web browsing session. The ADUs in these simulations are small, 20 KB files that constitute the simulated web page. Our simplistic web browsing simulation approach assumes no HTTP pipelining, so that another file starts transmitting when the previous file has been received, as described in section 6.1.4. What becomes immediately obvious from the energy-delay chart is that the potential for energy saving is tremendous as the energy consumption drops from 134.1 J in the E2E case to 36.18 J in the 20 KB TBO case, a reduction of 73%. The average ADU delivery latency is, respectively, increased from 0.27 sec in the E2E case to 0.47 sec in the 20 KB TBO case. The large energy saving potential is attributed to the long idle periods between transmission of successive ADUs that may be readily turned into sleep periods when the rendezvous mechanism is employed. For TBO values larger than 20 KB, the energy consumption only slightly improves with a significant deterioration of the introduced delay. The key value of 20 KB for the TBO coincides with the file size of 20 KB selected for this set of experiments. Further simulations with different ADU sizes would be necessary in order to verify the two figures are, indeed, correlated. However, for a TBO half the transfer size and given that each transfer starts when the previous transfer completes, it is reasonable to deduce that the rendezvous mechanism remains highly fluctuating for the entire browsing session, resulting in suboptimal energy savings.

TBO	Sleep Count	Sleep Time	Trans Time
E2E	0	0	0
10	391	62.04	7.64
20	357	131.02	6.86
30	239	133.31	4.48
40	186	134.12	3.44
50	151	135.51	2.76
60	131	135.55	2.32
80	101	136.1	1.76
100	85	136.1	1.42
150	64	137.24	0.98

Table 8-8: HTTP with no competing traffic, sleep and transition information.

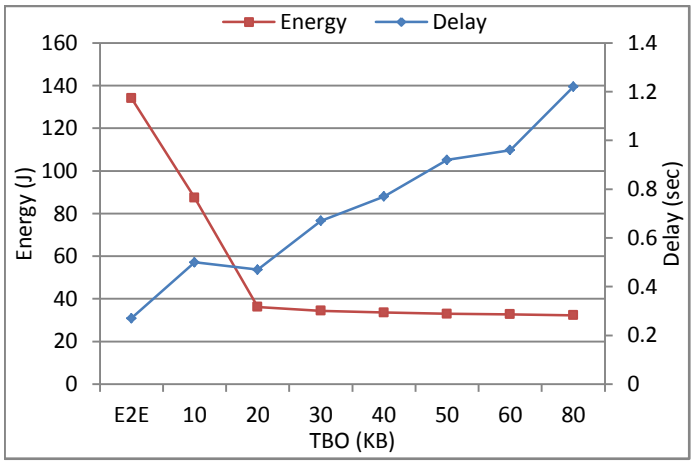

Figure 8-22: HTTP with no competing traffic energy and delay vs. the TBO.

Table 8-8 contains the sleep and transition information for the E2E and the various TBO cases. Except for the 10 TBO case, the idle time (sum of sleep and transition times) is 138 seconds for all cases, so the marginally improved energy expenditure for larger TBOs is attributed to the reduced number of sleep intervals, causing a reduction of the time spent in the energy-consuming transition state. The information regarding the 10 KB TBO case corroborates our assumption from the previous paragraph that a TBO

smaller than the ADU, combined with the particular transfer initiation pattern used in this web browsing simulation, results in suboptimal harnessing of the energy saving potential.

TBO	Delay No Flow	Delay CBR Flow	Delay TCP Flow
E2E	0.27	0.3	0.32
10	0.5	0.52	0.57
20	0.47	0.49	0.6
30	0.67	0.7	0.81
40	0.77	0.81	0.97
50	0.92	0.96	1.11
60	0.96	1.02	1.22
80	1.22	1.29	1.56
100	1.47	1.6	1.93
150	2.17	2.37	2.62

Table 8-9: HTTP delay with all types of competing traffic.

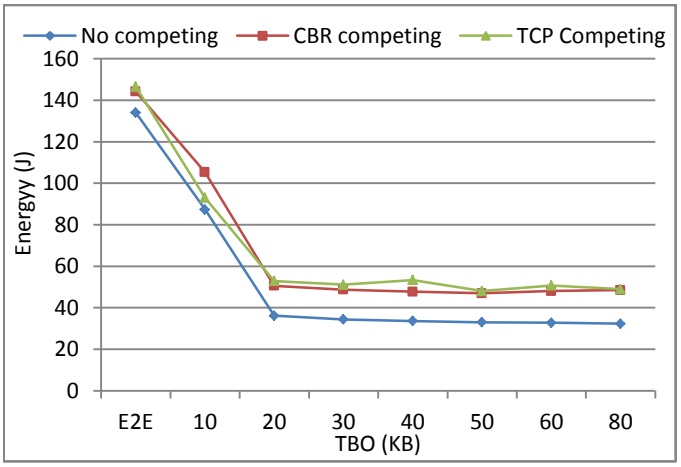

Figure 8-23: HTTP energy expenditure with all types of competing traffic.

Table 8-9 contains the delay values for the average ADU delivery during the simulated web browsing session. Expectedly, the delay when no competing flow is present is the lowest, followed by the delay in the CBR case and then by the delay in the TCP case. Despite using small files, the TCP connection of the main HTTP transfer attempts to utilize all the available bandwidth during the busy intervals, so even a low-rate competing flow affects the data delivery time. This contrasts the results of the previous subsection, where, due to the constant data rate of the CBR traffic, the delivery latency may be unaffected by other CBR traffic as long as the total load remains within the bounds of the network capacity. The chart of Figure 8-23 shows that the energy consumption follows a similar pattern for all types of competing traffic. However, the achieved energy savings are considerably higher when no competing flow is present.

Contrary to file transfers and streaming applications studied in the previous subsections, the web browsing user experience is sensitive to the application responsiveness. Furthermore, the additional delay of 200-300 ms introduced by the buffering at the overlay will be actually multiplied by the number of elements included in a web page, resulting in a, potentially unacceptable overall delay. However, modern browsers simultaneously open multiple TCP connections so that the web page elements can be more quickly downloaded reducing the overall delay. All these connections in the DTN overlay are handled collectively so that the TBO refers to the total data amount belonging to all active flows, destined to the same node, mitigating the total delay. In any case, this type of proposed buffering in web browsing applications may deteriorate user experience and, thus, it should be employed with prudence. A possibility would be to only enable this feature with the user's consent, in cases the battery level of the device is running low. This way the device operation could be extended at the expense of lower web browsing user experience.

8.2. Decentralized Space-Data Dissemination with DTN

The main simulation scenarios constructed as part of our proposed space-date dissemination scheme evaluation plan are fairly complex, as they involve a large number of nodes, irregular contact patterns and large amounts of data over long time periods. The main simulations allow us to approach the overall problem at a macroscopic level, but provide us with no insight into the microscopic, contact-level system operation. Thus, before describing the main simulations, we present simulation results involving a single, average-sized contact taken from the low-cost broadcast space-link model (see section 6.2.2) that help us better understand the small-scale behavior of the system. Then we investigate the overall system dynamics for all three types of space-links and, finally, we focus on the broadcast simulations and our proposed ground data distribution schemes.

8.2.1. Small-Scale Simulations

In these small-scale simulations we examine the effects of varying parameters such as the ADU/bundle size and data production rate in the system performance during a single contact. By focusing on a single contact and carefully setting the data production rate we isolated the effect of the contacts pattern, mitigating the effect of the transmission scheduling strategy on the system performance. The experiments involve a single contact of 446 seconds duration and 3 Mbps bandwidth (the average duration and the bandwidth of the contacts in the low-cost broadcast space-link model). In this series of experiments, real-sized bundles were used (10 – 70 KB) so the actual BER was applied (i.e. 10^{-7}), rather than the approximation of section 6.2.3.

Varying Bundle Size

Firstly, the bundle size was varied from 10 to 70 KB setting the ADU and the buffer size to 1.2 MB and the ADU production period to 4 seconds. The production of a 1.2 MB ADU every 4 seconds along with the required protocol headers almost saturates the link. It can be seen in Table 8-10 that varying the bundle size does not have a significant impact on the delivery latency and that the delivery ratio exhibits small fluctuations around an average value of 40%. ADUs are lost due to corruption of the corresponding bundles caused by the BER. The number of produced ADUs for this series experiments is relatively small (i.e. 112) and so the effect of the random error may cause fluctuations especially for larger bundles.

Bundle Size	Avg Latency (s)	Delivery Ratio (%)
10000	3.42372	45.54
20000	3.345104	41.96
30000	3.369672	43.75
40000	3.455364	44.64
50000	3.520799	41.96
60000	3.518108	33.04
70000	3.52377	41.96

Table 8-10: Effect of the bundle size

Varying ADU Size

In this set of experiments the bundle size was set to 60 KB and the ADU size was varied from 300 KB to 2.1 MB. The ADU production interval was varied accordingly so that the data production rate remained equal for all experiments (1-7 seconds). The delivery ratio, displayed in Figure 8-24, starts at around 82% for the 300 KB case and drops to 25% in the 2100 KB case. The decrease in the delivery ratio is attributed to the effect of the BER. Larger ADUs are more susceptible to bit corruption and, thus, the amount of delivered data decreases. In Figure 8-24 it can be seen that the average latency increases linearly with the ADU size, so time-sensitive data would rather be packaged in small ADUs.

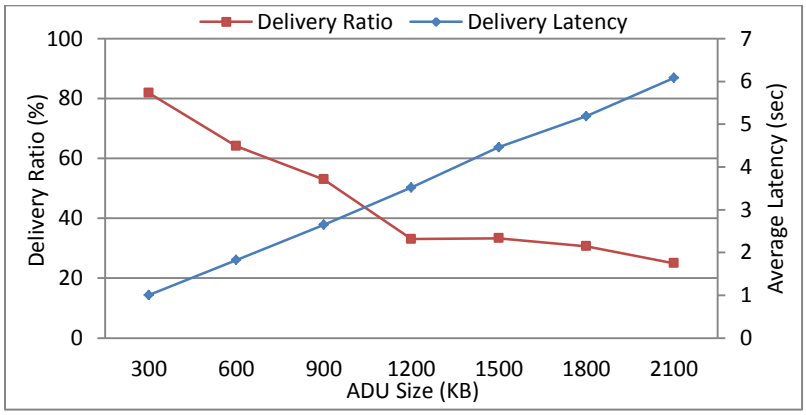

Figure 8-24: Delivery ratio and average delivery latency vs. ADU size.

Varying Production Rate

Assessing the effect of varying the data production involved fixing the ADU and the buffer sizes to the same value of 1.2 MB and varying the ADU production interval from 4 to 10 seconds, amounting to approximately 1x to 0.4x of the link bandwidth respectively. Since the buffer size is equal to the ADU size only one ADU can be stored in the buffer at any given time and, therefore, a new ADU always displaces the previous ADU. It follows that each ADU is transmitted (and possibly retransmitted) for duration equal

to the transmission interval. If the ADU is not received successfully during a transmission interval it is terminally lost. The bundle size was set at 60KB (each ADU consists of 20 bundles).

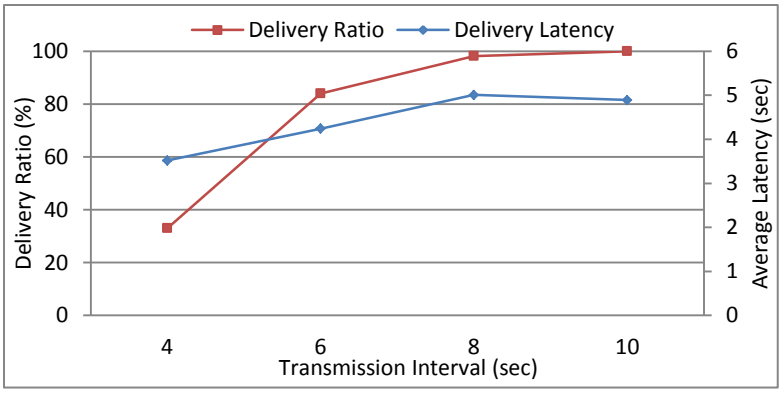

Figure 8-25: Delivery ratio and average delivery latency vs. transmission interval.

Figure 8-25 depicts the delivery ratio and latency achieved for the corresponding ADU creation periods. For a period of 4 seconds (equal to the link bandwidth) only 33% of the produced ADUs are delivered on the ground. This value reaches 98.2% and 100% at periods of 8 and 10 seconds respectively, which correspond to data production rates equal to or less than half the available bandwidth and result in virtually reliable communication. At the 4 second case, where the data production rate equals the link bandwidth, the delivery ratio approximates the probability of an entire ADU to be transmitted error free if we apply the conversion formula 6-1, as described in section 6.2.3, for a packet size equal to the ADU size of 1200000 bytes. Specifically, the probability for the successful reception of an ADU, calculated by the conversion formula for 10^{-7} BER and 1200000 bytes packet size, equals 39%, close to the simulated 33% value.

The delivery latency, also depicted in Figure 8-25, starts at 3.5 sec for the 4 sec production interval case and advances to 4.2, 5 and 4.9 sec for the 6, 8 and 10 sec production interval values respectively. Reasonably, the delivery latency in all cases is smaller than the production interval value, since each ADU is discarded after a production interval. For low values of the transmission interval the delivery latency is closer to the transmission interval value because the produced ADUs have less chance for retransmission before the next ADU arrives. For high transmission interval values the ADUs are successfully received almost half way through the transmission interval, suggesting that the link is underutilized.

8.2.2. Investigating System Dynamics

In this section, we present the simulation results of various experiments targeting the selection of representative model/system configuration values with respect to the link budget analysis produced by the STK simulations. The motivation for this initial set of experiment runs was to gauge the capacity of the network for each set of physical link characteristics. The input of the simulated system corresponds to the amount of data produced by the application layer on-board the satellite and, therefore, we aimed our effort at determining a data production rate and pattern that would lead to an acceptable ADU delivery ratio on

the ground. The rationale behind our work plan was to view the space link capacity as the bottleneck of the system and try to accordingly adjust data production in order to match the available capacity.

As our proposed solution does not presume any type of feedback with respect to in-range GS or ADU delivery status (reception or delivery feedback is optional), the system cannot guarantee reliability and, therefore, an acceptable ADU delivery ratio threshold needed to be specified. Preliminary results confirmed the intuitive hypothesis that increasing the data production rate leads to a higher overall amount of delivered data, while at the same time the ADU delivery ratio drops. The desired trade-off between the volume and the ratio of delivered data can be determined according to the application requirements. For the purpose of the simulation experiments presented in this section, we used a target delivery ratio value of approximately 80%. At this point our intent was to concentrate on the space network segment, so we included only a single end-user, significantly expediting simulation execution.

In the upcoming subsection we report on the link budget that resulted from the physical simulations with STK. Subsequently, we describe the end-to-end simulation results for the point-to-point models (direct and relay) and for the broadcast models.

Physical Links Capacity

The physical simulations were conducted in the STK simulator according to the experimental methodology described in section 6.2.2. These simulations include a total of 7 scenarios based on the space-link models of section 4.3, producing the corresponding contact file for each scenario. Each resulting contact file contains several thousand records and, therefore, information on the contact characteristics could not be extracted by simple inspection. Thus, the resulting contact files were analyzed using post-simulation scripts. Table 8-11 presents contact opportunities statistics for each one of the examined scenarios. Avg. Duration is the average contact duration, and Total Duration and Total Capacity are the contact duration and capacity for the entire 10-day simulation period after any overlapping contacts have been merged.

Link Model	Avg. Duration	Total Duration	Total Capacity
Direct (520 Mbps)	8 minutes	23.6 hours	5.5 TB
Relay (100 Mbps)	1.3 hours	10 days	10.8 TB
Broadcast (3 Mbps) (52 GSs, 1 Sat)	7.4 minutes	1.3 days	43.4 GB
Broadcast (3 Mbps) (100 GSs, 1 Sat)	7.4 minutes	3.1 days	100 GB
Broadcast (3 Mbps) (140 GSs, 1 Sat)	7.4 minutes	3.7 days	119 GB
Broadcast (3 Mbps) (52 GSs, 6 Sats)	6.6 minutes	8.1 days	260 GB
Broadcast (3 Mbps) (52 GSs , 46 Sats)	5.8 minutes	40.7 days	1.3 TB

Table 8-11: Contact Opportunities Statistics

The reported statistics provide useful information about the contact characteristics and set the upper bound on the data amount that can be transferred from space to Earth. In the direct point-to-point (Direct) case the total capacity of the contacts is 5.5 TB for 520 Mbps links of average duration of 8 minutes. In the relay point-to-point (Relay) case, the total capacity is almost double that of the Direct case (10.8 TB), despite the lower bandwidth (100 vs. 520 Mbps). This is due to the continuous contact of the satellite to the GEO relay, reaching 10 days for the 10-day simulation duration vs. less than 1 day for the direct point-to-point case.

The baseline low-cost broadcast scenario with 52 GSs and 1 satellite produces contacts of 43.4 GB total capacity and a total duration of 1.3 days. When the number of GSs is increased, the average contact duration remains at 7.4 minutes, but the total capacity increases to 100 and 119 GB for the 100 GS and 140 GS cases respectively. The additional 40 GSs of the 140 vs. the 100 GS cases do not produce analogous increase in the total capacity due overlapping contacts with other GSs. For 52 GSs and 6 satellites the average contact duration drops to 6.6 minutes, because each GS may only be receiving the transmission of a single satellite at any given moment. However, the total system capacity is 260 GB, which amounts to approximately 6x the capacity of the 52 GSs and 1 satellite case. For 46 satellites the total capacity reaches 1.3 TB and the average duration decreases even further to 5.8 minutes, suggesting that the 52 GSs become the bottleneck with respect to the resulting contacts.

Point-to-Point System Dynamics

Our aim in the point-to-point case was to study the system behavior and gauge its capacity under the conditions commonly used nowadays. The satellites only transmit data during contacts and ground stations immediately acknowledge received bundles over bidirectional links. First, we experimented with the direct model (i.e. no relay satellite) using a daily data production of 400 GB and varying the Time-to-Live (TTL) for the satellite data (by appropriately adjusting the transmission buffer), in an attempt to specify an appropriate TTL value. It can be seen in the chart of Figure 8-26 that the delivery ratio increases with the TTL up to the 12 hour value, where it slightly exceeds 97%. For a TTL of 24 hours the results are identical, so we select 12 hours as an appropriate TTL for the transmission mechanism. The delivery latency increases with the delivery ratio since higher TTL values give older data better chances of being received, contributing to the average delivery latency reported on the chart.

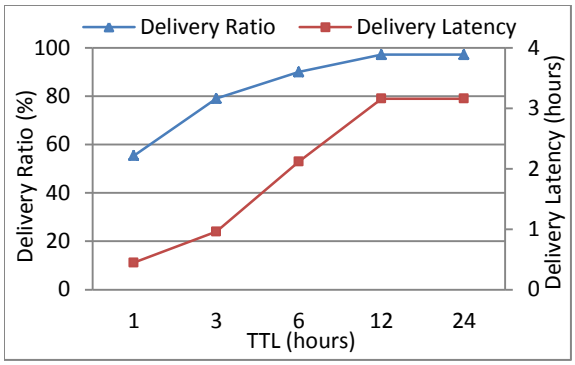

Figure 8-26: Direct point-to-point delivery ratio and latency varying data TTL.

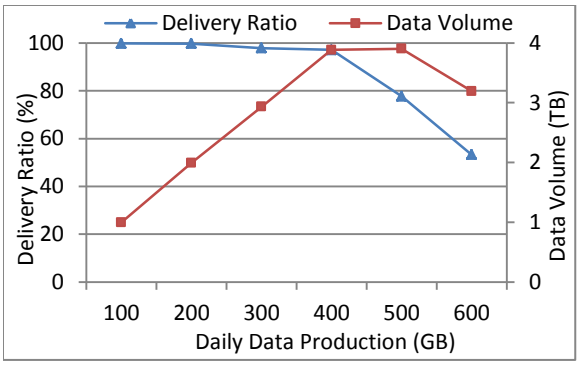

Figure 8-27: Direct point-to-point delivery ratio and data volume varying the daily data production.

For the selected TTL of 12 hours, we experimented with various daily data production values ranging from 100 to 600 GB. The results are shown in the chart of Figure 8-27. For low daily data production values (100 and 200 GB), the delivery ratio is over 99%, while the total system throughput is quite poor at 1 and 2 TB, respectively. The system throughput increases up to a daily data production value of 400 GB, where the delivery ratio is slightly over 97% and the data volume is 4 TB. For higher data production values the delivery ratio drops and the system throughput remains the same or drops as well. The small percentage of lost data for a daily data production up to 400 GB amounts mostly to data produced towards the end of the simulation, which are allowed only limited time in order to be delivered (data production continues until the simulation end). From the above discussion we deduce that the maximum data that the system can reliably accommodate is 400 GB per day. Following the same methodology, we found that for the relay point-to-point model the corresponding value was 1 TB per day.

Broadcast System Dynamics

Similarly to the previous section, this section includes simulation experiments aiming at selecting an appropriate data TTL and probing the system capacity for the broadcast models. Since a single EU is used, the ground distribution method (P2P or client-server) does not significantly affect the simulation outcome, so the results apply to both cases. Using a moderate daily data production of 2 GB, we experimented with data TTL values ranging from 1 to 24 hours, assuming the collected data are time-insensitive.

Our findings, which are depicted in Figure 8-28, show that the data delivery ratio exhibits a steady increase up to a 12-hour TTL and remains almost the same (slightly over 80%) for the 24-hour TTL case. However, the delivery latency in the 24-hour case is almost double that of the 12-hour case, so we select the 12 hours TTL value as in the point-to-point models.

Using the 12-hour TTL we experimented with daily data production values ranging from 0.5 to 30 GB. The chart of Figure 8-29 clearly highlights the tradeoff between the achieved delivery ratio and the total volume of delivered data. For very small data production values, the delivery ratio is almost 100%, while the data volume is a mere 5 GB (0.5 GB per day for 10 days). For a data production of 2 GB the data

volume is 16.5 GB at a delivery ratio of 82.7%. Increasing the data production increases the data volume to almost 37 GB, which is close to the maximum theoretical capacity of 43.4 GB reported in Table 8-11. The corresponding value for the delivery ratio is as low as 12%. For time insensitive applications requiring data reliability, such as small scale mapping and typical land surface motion, a TTL of 12 hours and a data production below 2 GB would be appropriate system parameters. Applications requiring data bulk such as soil protection and water management could push data production to around 10 GB, achieving 31 GB of data at a 30% delivery ratio.

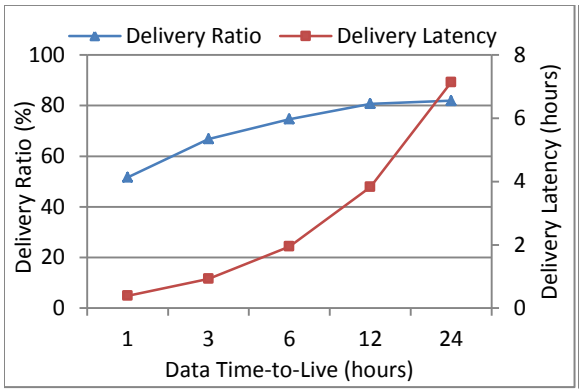

Figure 8-28: Low-cost broadcast delivery ratio and latency varying data TTL.

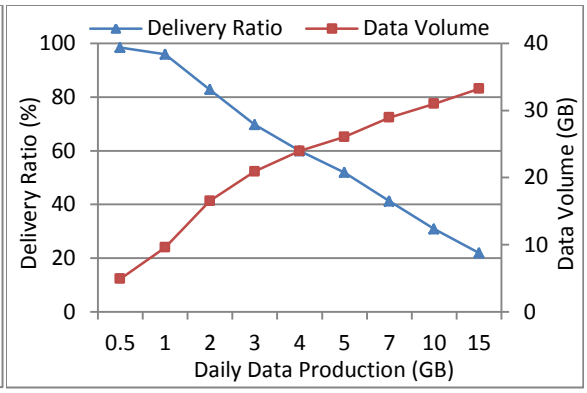

Figure 8-29: Low-cost broadcast delivery ratio and data volume varying the daily data production.

Similar simulations have been carried out assuming time sensitive data, discarded when the expiration time elapses. In this case, the data TTL on the satellite is set to the data expiration date, as there would be no use in transmitting already expired data. Figure 8-30 shows the delivery ratio and the data volume for a data time sensitivity of 0.5 and 2 hours. In the 0.5-hour sensitivity case, the delivery ratio starts at 36% and drops gradually as the daily data production increases, while the data volume exhibits a steady increase. In the 2-hour sensitivity case the delivery ratio starts at 72% for 0.5 GB of daily data production and drops to around 20% for the maximum value of 15 GB. The results here show that while the previously observed tradeoff between data reliability and data bulk is apparent, for highly sensitive data a low production rate does not significantly increase reliability and, therefore, data bulk may not be worth sacrificing. Real-time services requiring high reliability, such as sea ice monitoring and emergency land surface motion, may opt for moderate daily data production (approximately 3-4 GB) at the expense of only a small delivery ratio reduction. At high daily data production values, results for both cases converge to virtually identical values.

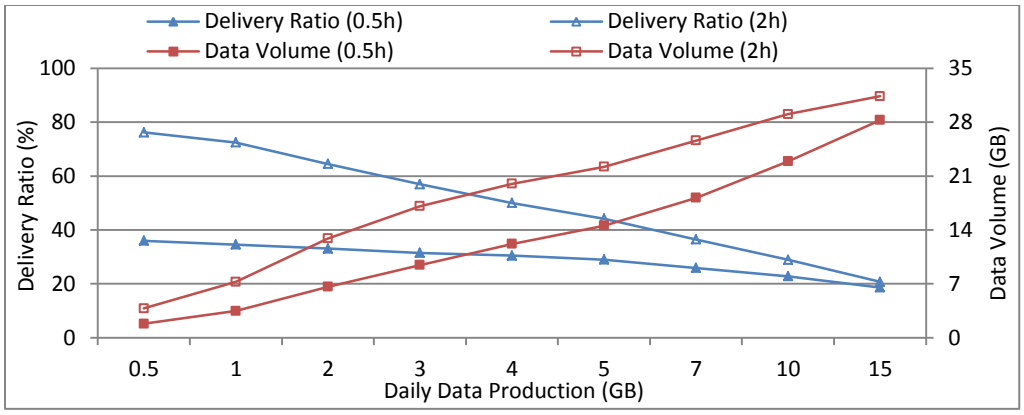

Figure 8-30: Low-cost broadcast delivery ratio and data volume varying the daily data production for time-sensitive data.

8.2.3. Broadcast Simulations

In this section we focus on various scenarios involving the low-cost broadcast space-link model. In the previous section the system capacity for the baseline configuration of 52 GSs and 1 satellite was investigated. Based on this investigation, and unless otherwise noted, the TTL used in the simulations presented in the rest of this section is 12 hours and the daily data production 2 GB.

Space Segment

This subsection includes simulations varying the number of ground stations and the number of satellites in the low-cost broadcast space link model. In the first set of simulations, the number of ground stations providing downlink services to a single LEO satellite varies among 52, 100 and 140 units, so that the system performance can be evaluated for different ground station geographical distribution. In the second set of simulations, a fixed number of 52 ground stations is used, while the number of satellites varies among 1, 6 and 46 units. Comparing the broadcast P2P model, supported by a single satellite solely, against the direct and the relay point-to-point models would be largely unfair due to the major difference in the associated deployment cost and equipment specifications. Therefore, the purpose of the second case is to estimate the total data return achieved by the broadcast P2P model in configurations that employ several low-cost satellites. This estimate will give us a general idea of the comparative performance of dense, low-cost satellite networks vs. the traditional high-end approach. The focus of these simulations has been on the space segment so a single EU was employed. The data TTL on the satellite was set to the previously specified value of 12 hours.

Figure 8-31 presents the results of the first test case, which clearly show that employing more ground stations has a positive impact on the overall performance of the system. Delivery latency is significantly decreased while all other metrics, including delivery ratio, are improved. Furthermore, a trade-off between the number of the employed ground stations and the improvement of system's performance is also made apparent. The addition of the extra 40 GSs between the 100 and the 140 GS cases, which provide downlink

services to the LEO satellite, does not seem to have a significant impact on the overall system performance. This is attributed to the geographical location of the extra 40 GSs, which create overlapping contacts with already existing GSs, thus, providing limited additional contact opportunity.

Results regarding the second test case are presented in Figure 8-32. We can observe that the data return increases linearly with the number of employed satellites for the 1 and 6 satellite cases (when 6 satellites are employed the data return is 6 times higher than the single satellite configuration). This linear increase does not hold true for the 46 satellites configuration, where data return of the system improves only by a factor of 3.8 vs. a 7.7 increase in the number of satellites compared to the 6 satellites configuration. This fact indicates that the ground station network is transformed into a bottleneck; a conclusion that it is also supported by the low values of the delivery ratio metric in comparison to the previous cases (i.e. single and 6 satellites cases). Since the creation of the bottleneck cannot be related to the ground station link bandwidth, the total number and duration of contact opportunities between the satellites and ground stations seems to be the only remaining factor causing this issue.

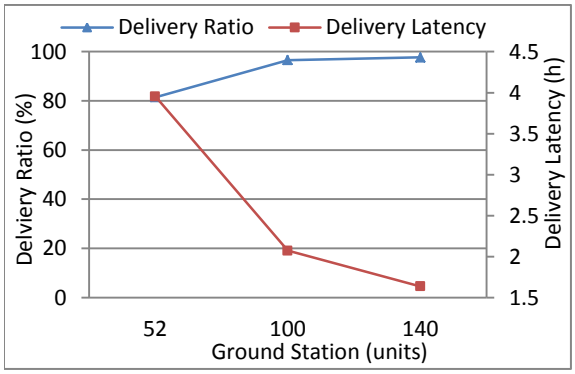

Figure 8-31: Low-cost broadcast delivery ratio and delivery latency varying the number of ground stations.

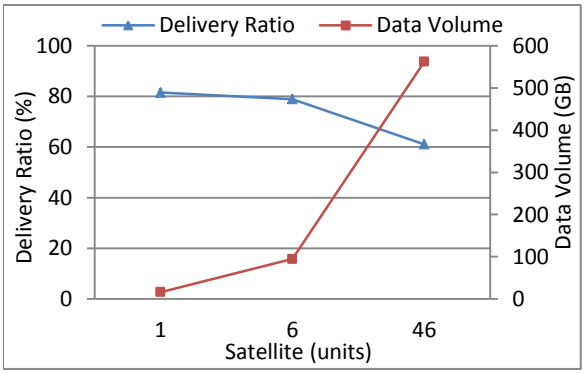

Figure 8-32: Low-cost broadcast delivery ratio and data volume varying the number of satellites.

In the 46-satellite case the total volume of data delivered on the ground reaches 562 GB. This value is approximately 1/7 of the maximum 3.9 TB amount of data delivered by the high-end scenario in the direct point-to-point model (section 8.2.2). Apparently, a direct comparison of the performance of the low-cost vs. the high-end satellite mission types would clearly favor the latter. However, the results are, indeed, promising, considering the vast gap between the specifications of the two systems (i.e. 3 Mbps vs. 520 Mbps). Furthermore, due to practical purposes, the 46 satellite simulation included a limited number of 52 GSs, restricting the overall contact time (a larger number of GSs would increase the achieved data volume). The comparison could lead to more useful insights in case economic data were utilized for the calculation of the cost-per-byte in each type of mission. Such analysis is not part of this work, but constitutes an interesting direction for possible future work.

Reliability Mechanisms

In this set of simulations we present comparative results of the basic broadcasting mode of operation, which was used in the previous sections and includes no delivery feedback (*Plain*), with the acknowledged mode of operation (*Ack*) and the acknowledged/transmission limited mode of operation (*Ack & Limit*). A single EU is used so that the focus remains on the space segment. As detailed in sections 4.5.2 and 4.5.3, data in the *Ack* mode are acknowledged via a designated uplink ground station and, in the *Ack & Limit* mode, additionally to the data acknowledgment, transmission is limited during contacts. In this set of simulation runs, a daily data production of up to 4 GB was employed as this is the maximum theoretical capacity of the contacts, guaranteeing relatively high delivery ratio values.

The chart of Figure 8-33 depicts the delivery ratio achieved in the three modes of operation. For a daily data production of 1 GB all three cases achieve almost 100% reliability. For higher data production values, the Ack & Limit mechanism achieves consistently higher delivery ratio than the other two and the Ack mode of operation achieves higher delivery ratio than the Plain mode. At the maximum data rate of 4 GB per day, the delivery ratio for the Ack & Limit, Ack and Plain modes of operation is 87%, 72% and 60% respectively.

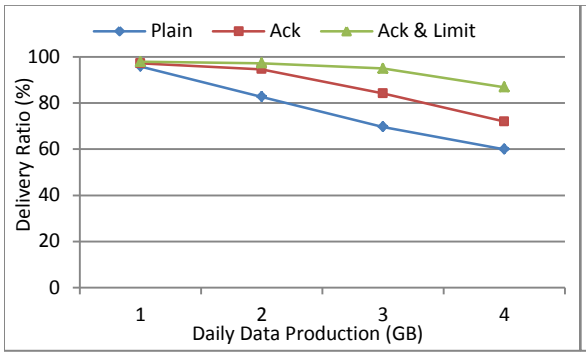

Figure 8-33: Delivery ratio for broadcast plain, acknowledged and limited transmission varying the daily data production.

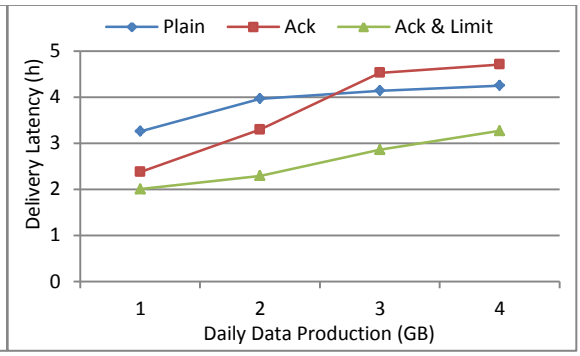

Figure 8-34: Delivery latency for broadcast plain, acknowledged and limited transmission delivery latency varying the daily data production.

Regarding the delivery latency, it can be observed in Figure 8-34 that the *Ack & Limit* mode has the highest performance ranging from 2 to 3.3 hours. For low data production rates the *Ack* mode performs considerably better than the *Plain* mode. For higher data production rates the *Ack* mode exhibits slightly higher latency, due to the significantly higher delivery ratio it achieves (84% vs. 70% and 72% vs. 60%). The results in this section show that employing "smart" transmission mechanisms, including delivery feedback and/or transmission suspension, can substantially improve system performance and should be considered as an option during mission design.

Ground Segment

In this final set of experiments the ground network includes 100 EUs, so that the performance gains of a peer-to-peer vs. a centralized ground data distribution scheme could be quantified. The ground network uses links of lower bandwidth than the rest of the simulations (10 Mbps vs. 18 Mbps) and no acknowledgments or transmission limitation mechanisms are employed (*Plain* mode).

Figure 8-35 depicts the delivery ratio and delivery latency for both ground distribution schemes. At low data production values the delivery ratio is identical in both cases, while the delivery latency is slightly higher in the centralized case. As the data production rate increases, the delivery ratio deteriorates more quickly in the centralized case reaching 31% vs. 60% of the P2P case at a daily data production of 4 GB. In the centralized distribution scheme the links around the MCC become congested and end-users located far from it are less likely to receive data within the simulation duration. In the P2P case, however, data received at a certain ground station become directly available to nearby EUs and dissipate over the multicast tree, evenly loading the network links. This becomes more apparent when examining the delivery latency line. In the P2P case delivery latency stabilizes at approximately 4 hours for higher data production values, whereas it exceeds 8 hours in the centralized case. The results suggest that a push, peer-to-peer ground data distribution would be highly beneficial especially for applications involving high volume, time-sensitive data, such as rapid mapping for fires and floods and weather prediction.

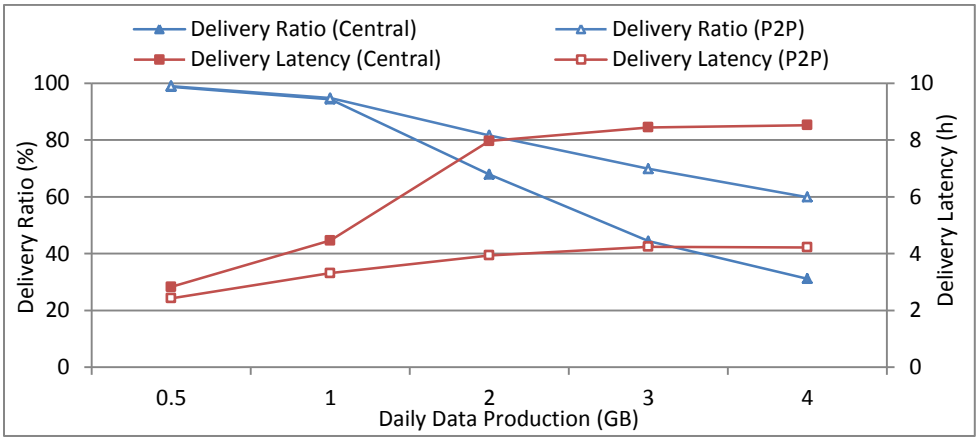

Figure 8-35: Low-cost broadcast centralized vs. P2P ground data distribution delivery ratio and delivery latency, varying the daily data production.

8.3. Storage-Constrained Routing in Opportunistic DTNs

The following sections present the simulation results for the storage-constrained routing in opportunistic DTNs. Firstly, the baseline simulations are described, where each node accepts incoming bundles with respect to only locally available storage. Secondly, the history-based simulations are presented, where bundle acceptance depends on the peers' acceptance history.

8.3.1. Baseline Simulations

This section presents simulation results for scenarios where nodes exercise impartial and partially selfish bundle acceptance criteria as described in section 5.3. The impartial bundle acceptance case serves as a benchmark for the ideal network operation, in terms of both efficiency and fairness. The partially selfish bundle acceptance case investigates the network performance in the presence of both partially selfish and impartial nodes. For a detailed description of the test plan please refer to section 6.3.3.

Impartial Bundle Acceptance

Table 8-12 lists the 3 metrics showing the overall network picture. The achieved delivery ratio of 0.71 represents the maximum network capacity and shows that we inject approximately 40% more traffic than what can be serviced. The fairness index under these conditions is very close to the ideal value of 1.

Avg Time (sec)	0.607
Delivery Ratio	0.710
Fairness	0.987

Table 8-12: Performance metrics in the deterministic bundle acceptance simulations.

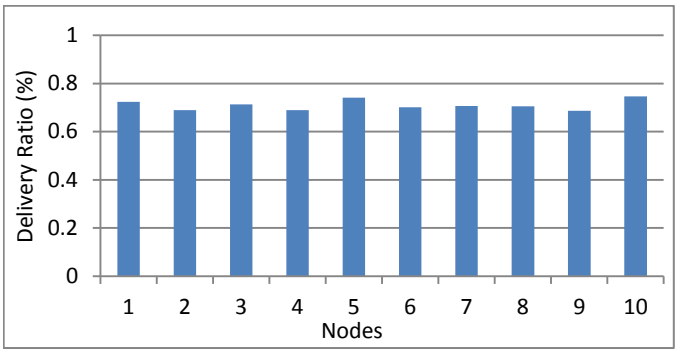

Figure 8-36: Delivery ratio for all nodes in the deterministic bundle acceptance simulations.

The chart in Figure 8-36 includes the delivery ratio achieved by the 10 individual nodes and corroborates the conclusion drawn by the value of the fairness index that the bundle delivery is fairly distributed among the 10 nodes. Delivery times are also on the same level for all participating nodes and, thus, reporting on them has been omitted for brevity.

Partially Selfish Bundle Acceptance

Table 8-13 shows the overall network metrics for the cases where all nodes, a single node and half of the nodes respectively apply partially selfish criteria in accepting bundles for relaying. As we described in previous sections, partially selfish (or non-deterministic) bundle acceptance means that the nodes favor their own, locally created bundles, especially as local storage space becomes scarce. In the all-selfish case, the fairness index is very high, but the overall delivery ratio value of 0.516 is considerably smaller than the 0.71 value achieved in the benchmark case. When just a single or half of the nodes behave partially selfishly the overall delivery ratio is higher, while the fairness index drops.

The chart of Figure 8-37 presents the per-node delivery ratio. In the all-selfish case the blue bars certify that bundle delivery is highly fair, but has a rather low value, showing that the network is underutilized. In the single case, node 1 is the partially selfish node and enjoys higher delivery ratio (0.92) than all other nodes. The nodes 6-10, which are on the opposite side of node 1, experience slightly lower delivery ratio than the nodes 2-5 since they are serviced by only 4 nodes. In the half-selfish case nodes 1-5 (the partially selfish nodes) experience delivery of almost 1 whereas the rest of the nodes experience delivery ratios as low as 0.4, a situation that results in the very low fairness index seen in the previous table.

	All	Single	Half
Avg Time (sec)	0.469	0.577	0.489
Delivery Ratio	0.516	0.703	0.678
Fairness	0.978	0.954	0.778

Table 8-13: Performance metrics in the non-deterministic bundle acceptance simulations.

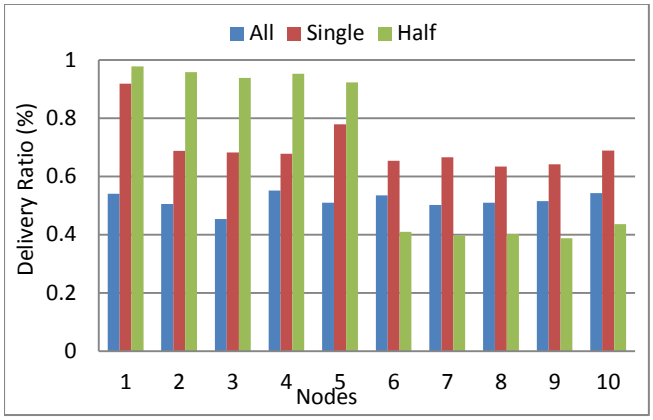

Figure 8-37: Delivery ratio for all nodes in the non-deterministic bundle acceptance simulations.

8.3.2. History-Based Simulations

This section contains simulations involving history-based bundle acceptance criteria inspired from Game Theory as described in section 5.3. Three types of criteria are tested as part of this set of simulations, one of which is deterministic and the other two non-deterministic. The rest of this section includes a subsection containing results of the deterministic and the 1st non-deterministic strategy (these strategies were rejected as not appropriate), and another subsection containing the 2nd non-deterministic strategy (this strategy was finally accepted as the most appropriate under the simulated conditions).

Rejected Mechanisms

Table 8-14 includes the overall metrics for the rejected mechanisms. In the deterministic case the nodes accept bundles for relaying by comparing the BAR of the requesting node with their occupied storage ratio as described in section 3.2. The delivery ratio achieved by this mechanism is 0.573 which is a rather acceptable value. However, the fairness achieved in this case is very low and thus the mechanism cannot be accepted. A complete picture of the individual delivery ratios can be seen in the corresponding chart of Figure 8-38. It is obvious that some nodes experience very high while others experience very low delivery ratio. During simulation it was observed that pairs of nodes spiraled into a hostile relationship, ending up not servicing each other at all. Other nodes seem to maintain better relations with their peers, enjoying

better service. This irregularity of the network operation called for rejection of the deterministic mechanism.

	Det	Non-det
Avg Time (sec)	0.453	0.689
Delivery Ratio	0.573	0.199
Fairness	0.821	0.86

Table 8-14: Performance metrics for the rejected bundle acceptance mechanisms.

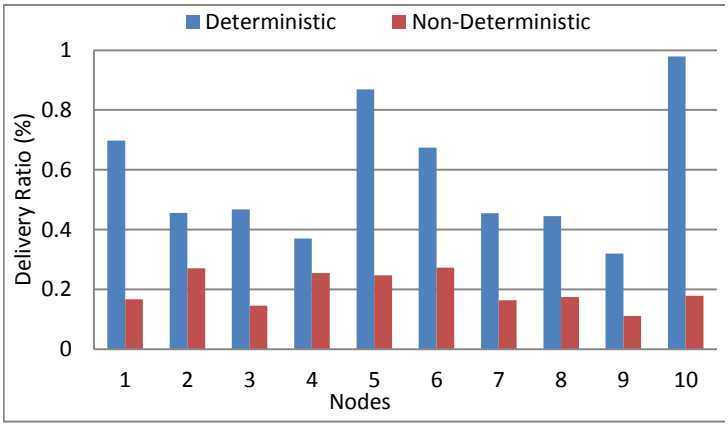

Figure 8-38: Delivery ratio for the rejected bundle acceptance mechanisms.

The non-deterministic mechanism (1st strategy) adds a probability for rejecting a bundle even if the BAR compared to the storage occupancy ratio suggested otherwise. The result is very low delivery ratio for all nodes additionally to the relatively low fairness index. Employing this mechanism has the effect of the nodes being very suspicious with each other, not servicing relay traffic and so the overall network performance is extremely poor.

Accepted Mechanism

This section reports on the experiments when the accepted mechanism is used. The mechanism is based on the non-deterministic 2 bundle acceptance criterion, as described in section 3.2. This mechanism eases the rejection probability of the non-deterministic 1 mechanism by squaring the storage occupancy and, thus, lowering the rejection threshold, especially when the storage occupancy is low. Therefore, the nodes are more generous and friendly with each other and overall network performance improves. The first column of Table 8-15 shows that when all nodes use this mechanism, both the delivery ratio and the fairness index are acceptable. When one node is totally selfish (does not relay any bundles for other nodes) both values drop.

	All	Single
Avg Time (sec)	0.453	0.468
Delivery Ratio	0.58	0.49
Fairness	0.959	0.905

Table 8-15: Performance metrics for the accepted bundle acceptance mechanism.

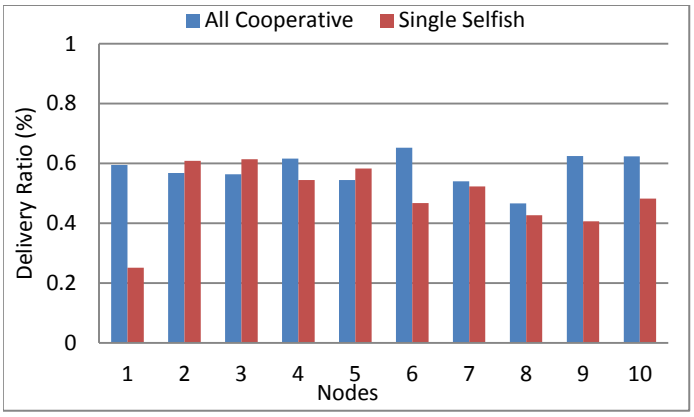

Figure 8-39: Delivery ratio for the accepted bundle acceptance mechanism.

The detailed chart in Figure 8-39 shows that, in the single selfish case, node 1 (the selfish node) experiences very low bundle delivery ratio compared to the rest of the nodes. Thus, the mechanism proves successful in giving incentives to the nodes for cooperating with each other and punishing those that behave selfishly. Under such a scenario one can be sure that node 1 has no gain from being selfish and would rather cooperate, in order to improve the experienced delivery ratio.

9. Afterword

The focus of this dissertation has been on employing concepts of the Delay-Disruption Tolerant Networking (DTN) in order to address computer networking problems, not directly related to the intended objectives behind the DTN architecture design. Contrary to the original purpose of DTN (i.e. the support of infrastructure-less, space and terrestrial wireless communications), our effort was mainly invested in developing solutions for infrastructure networks that involve traditional Internet protocols and possibly include a last-hop, wireless connection.

From its inception, DTN was eagerly embraced by the computer networking research community and has been proposed as a possible solution for a wide range of applications. To the day of writing these lines, over twelve years have passed since the first draft of the relevant RFC 4838 [1] and researchers are still striving to devise a DTN "killer application". Adoption of the architecture in space communications has also been relatively slow, mostly due to the high risk involved in deploying new technology on the always sensitive and expensive space missions. Nevertheless, there have recently been positive signs towards a wider acceptance of DTN, including a standardization initiative taken by Boeing, a major player on the world market, and the successful deployment of the Bundle Protocol on the International Space Station ([161][162]).

We argue that, unlike other networking technologies, DTN may not need a vastly successful application in order to be firmly established as a networking architecture. Instead, DTN can become the de facto standard for supporting various types of solutions on the edge of the global Internet, where connectivity is not to be taken for granted. Currently, any such solution must rely on custom mechanisms (e.g. proxy services, protocol encapsulation, etc.), which are difficult to implement and may break the network semantics. Our work reflects this view by showing how expanding DTN facilitates such solutions on the edge of the well-connected Internet.

Specifically, the vast majority of our work has been conducted in the following directions:

- Enhancing the energy-efficiency of mobile networking devices by deploying a DTN overlay on a traditional IP network. The overlay buffers incoming data and allows the mobile receiver to switch its wireless interface to the sleep state, thus, conserving energy.
- Improving dissemination of space-data from the satellites to the end-users, employing a decentralized delivery scheme based on DTN. The delivery scheme is accompanied by a transmission scheduling mechanism, which allows for adjusting the data reliability vs. data volume tradeoff.
- Developing routing policies for storage-constrained opportunistic DTNs, inspired from Game Theory. These routing policies must achieve efficient and fair network resource allocation without the need for a central coordination authority.

- Implementing a detailed DTN simulation model in the well-known network simulation platform ns-2. This simulation model has been used as the basis for the development of the proposed solutions.

The rest of this chapter expands on the detailed conclusions we have drawn for each of the previously described directions followed in our work.

9.1. Energy-Efficient Networking with DTN

In our first proposed solution of an energy-efficient DTN overlay, we evaluated the potential of DTN to shape Internet traffic in a manner that allows mobile devices to utilize energy-conserving modes of operation, mitigating the risk to lose packets and degrade throughput. In order to take advantage of the traffic shaping capabilities of DTN we deployed a novel rendezvous mechanism between the BS and the mobile host. The evaluation of the DTN overlay has taken place in three stages; the conclusions for each stage are summarized in the following paragraphs.

In the first stage, the "proof-of-concept" scenario, a basic overlay was simulated with the help of a simple buffering application. The energy consumption was calculated in a post-simulation analysis of the recorded state transitions of the mobile device, as they resulted from the communication between the end-device and the edge network node. Our analysis reported both on the energy expenditure potential (the ideal energy conservation for a given state transition log) and on the actual energy expenditure when the rendezvous mechanism was employed. The main focus of the simulations was on FTP transfers and the results highlight the enormous energy conservation that can be achieved by exploiting the DTN qualities.

In cases of high network load, the energy conservation achieved by the rendezvous mechanism is well over 50% as compared to a regular end-to-end connection, while the introduced delay for the completion of the file transfer is negligible. Additionally, the results indicate quite clearly that the rendezvous mechanism exploits most of the energy saving potential created by DTN (the difference between the actual and the potential energy expenditure is marginal). Similar results have been observed in case a random error is present on the wireless channel, achieving energy conservation of more than 70% for high wireless error rates. The general trend in all types of simulation scenarios was that, as the network conditions become more challenged, the energy conservation achieved by the rendezvous mechanism becomes higher.

Experiments in this "proof-of-concept" scenario were also conducted in order to assess the effect of the Target Buffer Occupancy (TBO) on the achieved energy conservation. The corresponding results indicated that, generally, the energy conservation increased with increasing the amount of buffered data, reaching a steady value for TBOs larger than 60 KB. However, using excessively high TBO values increases traffic burstiness and intensifies network contention, thus limiting the reasonable TBO values in the range of 30-90 KB. The mean value for the actual buffer occupancy during the entire data transfer was found to coincide with the TBO value, showing that the rendezvous mechanism successfully converges to the desired data buffering amount. For the most part of the transfer, the instant actual buffer occupancy was

confined within an area of approximately 30% the TBO, exhibiting, however, a small number of large spikes during TCP slow start.

In the second stage of experimentation, the “full DTN overlay” scenario, the DTN overlay was simulated deploying multiple instances of our DTN agent along the network path, allowing for balancing network traffic over alternative routes. In cases of high network congestion and/or low storage availability on intermediate nodes, application data may be pushed closer to their destination instead of being withheld at the source. Hence, load balancing strictly associates with less retransmissions and shortened communication time, two tactics with energy-conserving consequences. Central role in this load balancing quality of the DTN overlay is the distribution of storage space occupancy over alternative network paths.

The focus of the “full DTN overlay” experiments was, similarly to the “proof-of-concept” scenario, on FTP transfers. However, the energy expenditure results were produced during simulation, utilizing the inherent ns-2 energy model, instead of relying on post-processing. The experimental results corroborated previous findings that our design is capable of enabling energy-efficient communication, without sacrificing data transfer performance. The results also showed that the additional DTN nodes further improve the energy efficiency achieved by the overlay. In case the connection between the source and the base station is split by intermediate nodes, the energy consumption drops by an additional average of 20% over the simple deployment scenario. In case there are storage space limitations on the intermediate nodes, load balancing on the bundle layer significantly expedites the data transfer, even though the additional energy conservation achieved is marginal.

In the third stage of evaluating our energy-efficient solution, the “simple DTN overlay”, we employed the DTN agent and utilized the inherent energy model of ns-2 on a basic 3-node DTN overlay, with a second receiver on the WLAN receiving competing traffic. The motivation behind this set of experiments was to evaluate the energy-efficiency gains of our solution with respect to the effect on the user experience for the three main types of traffic: file transfer, streaming and web browsing.

Experiments in the third stage reinforced previous results that file transfers are only marginally affected by the DTN overlay operation. The amount of buffered data is usually much smaller than the transferred file (i.e. the ADU of the application) and so the additional delay is negligible. Therefore, the proposed energy-efficient solution does not affect the end-user experience and can be unconditionally applied to file transfer application. With respect to streaming flows, the “simple DTN overlay” results have also shown that significant energy conservation can be achieved, introducing, however, significant delay at the datagram level (i.e. the ADU of the streaming application). Since the buffering amount is, generally, larger than the stream datagram, substantial delay is introduced due to the data buffering at the base station. Nevertheless, most streaming applications employ buffering on the client side in order to cope with data rate fluctuations, so the effect of buffering on the end-user experience should be fairly limited, and could be compensated by a slight increase in the client buffering amount. In contrast to file transfers and streaming applications, the user experience during web browsing depends highly on the application responsiveness. Our proposed buffering scheme in the DTN overlay may deteriorate user experience in web browsing

applications and, thus, it should be employed with prudence. A solution to this problem could be to selectively enable the energy-saving features with the user's consent, when the battery level of the device is running low. Thus, the device operation could be extended sacrificing the web browsing experience of the user.

During our work with the energy-efficient DTN overlay, we have identified a number of possible directions for future exploration. One such direction is to improve the response of the rendezvous mechanism to fluctuations in the incoming traffic by utilizing historical data and employing more sophisticated prediction algorithms. Currently, the next rendezvous calculation relies on the immediately preceding buffering interval only. Another direction is to extend the scheduling algorithm in order to support traffic originating from the mobile device to the Internet, as well as traffic destined to multiple mobile hosts. This enhancement could be benefitted from utilizing the bandwidth availability information, which is provided by our proposed passive bandwidth estimation mechanism for 802.11 networks. Finally, a natural continuation of our work would be to incorporate a simple version of the rendezvous mechanism within a real-world implementation of the Bundle Protocol and to evaluate the performance and energy efficiency on a real testbed.

9.2. Decentralized Space-Data Dissemination with DTN

In our second proposed solution we suggested a DTN-based, space-data dissemination paradigm that extends traditional satellite communications in order to accommodate low-cost, dense satellite networks with limited uplink capabilities. Our paradigm consists of a broadcast-based, P2P multicast ground distribution scheme and a best-effort transmission scheduling for space-data. Data dissemination from the satellite to the end-users is facilitated through a DTN overlay that utilizes Internet protocols to interconnect the space segment, which connects satellites and ground stations, and the ground segment, which connects ground stations and end-users.

Simulation experiments showed that the P2P ground distribution improves both the data reliability and timeliness over a centralized distribution, and is especially beneficial to applications involving high volumes of time-sensitive data (i.e. rapid mapping, weather prediction). More specifically, the P2P ground distribution achieved twice the reliability of the centralized distribution for high values of daily data production, while limiting the delivery latency to half that of the centralized case under the same conditions. Our proposed ground distribution scheme improves data delivery by creating shorter routes from the ground stations to the end-users and by reducing the burden on central nodes, eliminating the long delays introduced in a traditional centralized approach.

Our versatile transmission scheduling mechanism, employed in space, is able to adjust data reliability even in the absence of delivery or reception feedback, while it can utilize such feedback for improving performance when available. Through the simulation experiments, we were able to quantify the tradeoff between data reliability and data volume, and to determine appropriate adjustments for a variety of space application types. For applications involving time-insensitive, reliable data, such as small scale mapping

and land surface motion, a single satellite network transfers 16 GB at 83% delivery ratio, whereas for applications involving time-insensitive, bulk data, such as soil protection and water management, 31 GB can be transferred at 30% delivery ratio. For data of higher sensitivity, the data reliability/volume tradeoff becomes less apparent and the delivery ratio decreases significantly with the data expiration time. Real-time services requiring high reliability, such as sea ice monitoring and emergency land surface motion, could be benefitted from a moderate daily data production of 3-4 GB, sacrificing only a small portion of the achieved delivery ratio.

Space-data transmission efficiency can be significantly improved by incorporating two supplementary mechanisms: delivery feedback and transmission limitation during contacts. Delivery feedback may be provided by a designated end-user, generating acknowledgments that are routed through the DTN overlay to a ground station with uplink capabilities. The uplink ground station stores the acknowledgments and uploads them to the satellite during the next contact opportunity. Acknowledged bundles are removed from the satellite transmission buffer, eliminating unnecessary retransmission of data that have already been received. The second supplementary mechanism limits transmission during contacts, improving the chances for successful bundle delivery, while conserving valuable energy on the satellite. Contacts may be either preloaded on the satellite or dynamically calculated based on the satellite position. When both mechanisms are in place, the system performance is substantially improved, up to 40% in terms of data volume and 50% in terms of data latency.

Expectedly, the performance of currently used, high-end networking equipment was found to be considerably higher than that of the low-cost equipment, especially in a single satellite system. Firstly, this performance gap is attributed to the vast difference, of more than 2 orders of magnitude, in the space-link bandwidth offered by the two equipment types (3 vs. 520 Mbps). The performance gap widens even further, due to the immediate feedback provided by the bidirectional links of the high-end networking equipment, as opposed to the mainly unidirectional links in the basic low-cost configuration.

The gap in the performance achieved by low vs. high cost satellites narrows when multiple, low-cost satellites are employed, rendering the deployment of a low-cost broadcasting approach a promising solution. Our simulation experiments showed that a swarm of 46 low-cost satellites achieved 0.56 GB of delivered data, which, even though considerably lower than the 3.8 TB achieved by the high-end system, appears as a viable alternative in economic terms, taking into account the cost of a low-cost vs. a high-end satellite (i.e. few hundreds vs. several million Euro). In a future study, simulation results could be combined with economic data in order to calculate actual cost vs. performance indices (i.e. cost-per-byte) for each mission type and support related mission design decisions.

In the technical front, our future plans include the improvement of the transmission scheduling by adding support for various data prioritization schemes through weighted probabilities. Prioritization can be based on criteria such as data age, retransmission counter, or Type-of-Service. Future simulations could also report on the energy expenditure of the proposed communication patterns, since prudent energy management is vital for the satellite operation. Furthermore, future simulations could explore alternative

data acquisition schemes, involving targeted data collection over specific areas of interest, as opposed to the continuous data acquisition scheme used in the presented work. Finally, the necessary functionality for managing the DTN multicast distribution tree could also be incorporated into our proposed paradigm.

9.3. Storage-Constrained Routing in Opportunistic DTNs

The third part of our work involved a preliminary study of the storage-constrained routing problem in opportunistic DTNs. We developed bundle acceptance criteria that were based on the locally available storage space and incorporated mechanisms inspired from Game Theory. The goal was to devise acceptance criteria that would achieve high network performance and guarantee resource allocation fairness, especially in the presence of malicious nodes. The bundle acceptance criteria were implemented within our DTN agent simulation model and were tested in appropriate simulation experiments, involving a convenient network topology. The resulting network was first studied with all nodes adopting an impartial bundle acceptance in order to assess the maximum service capacity. Then, we experimented with a selfish bundle acceptance criterion where a node favored its own locally produced bundles over relay bundles and, finally, tested three mechanisms that take into account bundle acceptance history between nodes.

When all nodes behaved impartially, the network throughput achieved maximum throughput and fairness. However, this setup was susceptible to selfish nodes taking advantage of the network's openness. When planting a selfish node among an impartial group of nodes, the selfish node received approximately 30% better service than the impartial nodes. Additionally, we experimented with three, history-based bundle acceptance strategies. The first two strategies were deemed inappropriate due to poor performance in terms of the fairness and efficiency respectively. The third mechanism was found to achieve satisfactory network utilization when all nodes behaved well, while at the same time it punishes nodes that choose to be non-cooperative. Specifically, the delivery ratio reaches approximately 86% and the fairness 97% of the baseline scenario when all nodes are cooperative. The presence of a non-cooperative node does not significantly affect the service enjoyed by cooperative nodes, whereas the non-cooperative node itself receives very low service. In such a setting, a network node would refrain from behaving in a selfish manner, as this would ultimately lead to poor performance for itself.

In order to solidify our work on the subject, we would like to further test and refine the accepted mechanism so that it works well under different network load conditions. We would also like to explore scenarios where malicious nodes follow more sophisticated strategies (e.g. pretending to forward received bundles, but ultimately discarding them). Finally, the presented routing problem could be formulated as a game allowing the theoretical evaluation of the proposed strategies, possibly determining their respective equilibrium solutions.

9.4. DTN Simulation Model Development

Significant effort of our PhD work was put in the development of a DTN simulation model in the ns-2 network simulation environment. In the early stages of our work we made the decision to create a DTN

simulation model in ns-2, since the existing DTN simulation tools did not support the deployment of DTN over Internet protocols. Ns-2 provides detailed simulation models of protocols throughout the networking stack, specializing on Internet protocols, but includes no inherent support for DTN; hence, the DTN model had to be built from scratch. The resulting DTN simulation model has been made available to the research community on this link [30]. To this date, the corresponding code package is being frequently downloaded by interested parties and we also receive sporadic support requests from researchers.

Our proposed solutions involve protocols across the entire protocol stack, necessitating interventions in several of the built-in ns-2 models. From these interventions we gained valuable insight into the intricate workings of the involved protocols. Namely, we closely studied and modified code related to the WNIC of a mobile device (physical layer), the queue of an outgoing link (link layer), the 802.11 agent (MAC layer), and the underlying transport agent (transport layer), while we created the new bundle layer (between transport and application layers).

The rendezvous mechanism, which enables energy conservation for mobile devices, was fitted into the DTN agent so that it may be enabled between the base station and the mobile receiver. The DTN agent employs dynamic bundle fragmentation at the base station supporting the creation of data bursts that contain bundle fragments from multiple sources. The DTN model has also been extended to support simulation for space-data dissemination schemes. This work has been supported by ESA and the resulting simulation tools have been delivered to the Agency, along with accompanying documentation material for future use. The simulation model was part of an extended framework containing an STK model and numerous auxiliary scripts that allow the detailed simulation of satellite missions. Finally, the DTN simulation model was further extended to support a number of peripheral functionality such as:

- Passive bandwidth estimation for 802.11 WLANs.
- Bundle acceptance mechanisms for storage-constrained, DTN routing.
- Cryptographic operations and key exchange algorithms for DTN.

During the development of the DTN model, and its central entity the DTN agent, we confronted several practical issues related to the TCP and UDP convergence layers, the routing functionality, the custody transfer procedure, and the retransmission mechanism. Additionally, we identified the need for certain cross-layer interactions between the DTN agent and the TCP convergence layer. Cross-layer interactions allow the DTN agent to limit redundant data transmission and, also, make wiser decisions with respect to the custody transfer and retransmission mechanisms. A few indicative such practical issues we confronted are:

- Timing issues related to switching the wireless interface to the sleep mode and back to active with respect to the transition time of the device.
- Monitoring state for the 802.11 MAC protocol.
- Data segmentation, reordering and aggregation at the transport layer.
- Forwarding strategies (store-and-forward vs. cut-through) at the bundle layer.
- Retransmission timeout issues between the transport and the bundle layers.

- Overlay DTN routing on top of an IP network with established routes.

Open issues with respect to the DTN model development involve extending the current routing capabilities to support full-fledged, DTN routing, and enable experimentation with contact graph or opportunistic routing algorithms. Also, some work could be invested in making the deployment and expansion of the DTN model more user-friendly for the networking researcher, which could give greater adoption from the research community.

References

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Dust, K. Scott, K. Fall, H. Weiss, "*Delay-Tolerant Networking Architecture*," IETF RFC 4838. [Online]. Available: http://tools.ietf.org/html/rfc4838.txt, 2007.

[2] J. Postel, "*Internet Protocol*," IETF RFC 791. [Online]. Available: http://tools.ietf.org/html /rfc791.txt, 1981.

[3] J. Postel, "*Transmission Control Protocol*," IETF RFC 793. [Online]. Available: http://tools.ietf.org/html /rfc793.txt, 1981.

[4] J. Postel, "*User Datagram Protocol*," IETF RFC 768. [Online]. Available: http://tools.ietf.org/html /rfc768.txt, 1980.

[5] Telecommunications and information exchange between systems, "*Local and metropolitan area networks - Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*," IEEE Standard for Information technology, 2007.

[6] S. A. Pentland, R. Fletcher, A. Hasson, "*DakNet: Rethinking Connectivity in Developing Nations*," IEEE Computer Society, vol. 37, no. 1, pp. 78-83, 2004.

[7] R. Morris, J. Jannotti, F. Kaashoek, J. Li, D. Decouto, "*CarNet: a scalable ad hoc wireless network system*," in Proc. 9th ACM Special Interest Group on Operating Systems (SIGOPS) European workshop, New York, NY, USA, 2000.

[8] J. Ott, D. Kutscher, "*From Drive-thru Internet to Delay-tolerant Ad-hoc Networking*," in: Conti, M., Crowcroft, J., Passarella, A., (Eds.), Mobile Ad-hoc Networks: From Theory to Reality. Nova Science Publishers, Inc., pp 241-277, 2007.

[9] X. Li., W. Shu, M. Li, H. Huang, M. Wu, "*DTN Routing in Vehicular Sensor Networks*," in Proc. IEEE Global Communications Conference (GLOBECOM), New Orleans, LA, 2008.

[10] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, C. Diot, "*Pocket-Switched Networking: Challenges, Feasibility and Implementation Issues*," in Proc. 2nd IFIP Workshop on Autonomic Communications 2005, Athens, Greece.

[11] S.-A. Lenas, V. Tsaoussidis, "*Enabling free Internet access at the edges of broadband connections: a hybrid packet scheduling approach*", ACM SIGMOBILE Mobile Computing and Communications Review (MC2R), vol. 18, no. 1, pp.55-63, 2014.

[12] S. McCanne, S. Floyd, "*Network Simulator*," [Online], Available: http://www.isi.edu/nsnam/ns/, 1997.

[13] K. Fall and S. Farrell, "*DTN: An Architectural Retrospective*," IEEE Journal on Selected Areas in Communications, vol. 26, no. 5, pp.828-836, 2008.

[14] R. C. Shah, R. Sumit, S. Jain, W. Brunette, "*Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks*," Intel Research Tech Report IRS-TR-03-001, January 2003.

[15] K. Scott, S. Burleigh, "*Bundle Protocol Specification*," IETF RFC 5050. [Online]. Available: http://tools.ietf.org/html/rfc5050.txt, 2007.

[16] D. Vardalis, V. Tsaoussidis, "*Energy-efficient internetworking with DTN*," in Proc. 9th International Conference on Wired/Wireless Internet Communications (WWIC 2011), Vilanova i la Geltrú, Barcelona, Spain, 2011.

[17] A. Acquaviva, T. Simunic, V. Deolalikar, S. Roy, "*Remote power control of wireless network interfaces*," in Proc. 13th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS), Torino, Italy, 2003.

[18] C. Jones, K. Sivalingam, P. Agrawal, J. Chen, "*A Survey of Energy Efficient Network Protocols for Wireless Networks*," ACM Journal on Wireless Networks, vol. 7, no. 4, pp. 343-358, 2001.

[19] NASA Goddard Space Flight Center, Greenbelt, MD, USA, "*Interface Description Document for Eos Aqua X-Band Direct Broadcast*," June 2002.

[20] R. Torres, P. Snoeij, D. Geudtner, D. Bibby, M. Davidson, E. Attema, et al., "*GMES Sentinel-1 Mission*," Remote Sensing of Environment, vol. 120, pp. 9–24, 2012.

[21] R. Sandau, H. Röser, A. Valenzuela, "*Small Satellite Missions for Earth Observation*," in Proc. 7th IAA Symposium on Small Satellites for Earth Observation, Springer, 2010.

[22] E. Gill, P. Sundaramoorthy, J. Bouwmeester, B. Zandbergen, R. Reinhard, "*Formation Flying within a Constellation of Nano-satellites: The QB50 Mission*," Acta Astronautica, vol. 82, no. 1, pp. 110-117, 2013.

[23] S. Burleigh, V. Cerf, J. Crowcroft, V. Tsaoussidis, "*Space for Internet and Internet for Space*," Ad Hoc Networks, vol. 23, pp. 80-86, 2014.

[24] Space Internetworking Center, DUTH, Xanthi, Greece. ESA Statement of Work. [Online]. Available: http://www.spice-center.org/files/various/BitTorrent-like_Study_-_SoW.pdf.
[25] Telespazio VEGA UK, A Finmeccanica/Thales Company. [Online]. Available: http://www.telespazio-vega.com/.
[26] L. Sundararaj, P. Vellaiyan, "*Delay Tolerant Networking routing as a Game Theory problem – An Overview*," International Journal of Computer Networks, vol. 2, no. 3, 2010.
[27] U. Shevade, H. H. Song, L. Qiu, Y. Zhang, "*Incentive-Aware Routing in DTNs*", in Proc. The 16th IEEE International Conference on Network Protocols, Orlando, FL, USA, 2008.
[28] A. Keränen, J. Ott, T. Kärkkäinen, "*The ONE simulator for DTN protocol evaluation*," in Proc. The 2nd Simtools, Article 55, 2009.
[29] A. Varga. (1997). OMNet++. [Online]. Available: http://www.omnetpp.org/.
[30] D. Vardalis. "*DTN agent for ns-2, Space Internetworking Center,"* [Online]. Available: https://www.spice-center.org/dtn-agent/, 2012.
[31] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, D. Wetherall, "*Reducing Network Energy Consumption via Sleeping and Rate-Adaptation*," in Proc. of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI), San Francisco, CA, USA, 2008.
[32] D. Vardalis, S.-A. Lenas, S. Diamantopoulos, V. Tsaoussidis, F. Belli, P. Beavis, D. Patterson, K. Nergaard, "*Decentralized Space-Data Dissemination for Low-Cost, Dense Satellite Networks*," IEEE Transactions on Aerospace and Electronic Systems (to appear), 2015.
[33] B. Cohen, "*The BitTorrent Protocol Specification*," [Online]. Available: http://www.bittorrent.org/beps/bep_0003.html, 2008.
[34] AGI, Exton, PA, USA. Systems Tool Kit. [Online]. Available: http://www.agi.com/products/stk/.
[35] S. A. Menesidou, V. Katos, "*Authenticated Key Exchange (AKE) in Delay Tolerant Networks*," Information Security and Privacy Research (IFIP), Advances in Information and Communication Technology, vol. 376, pp. 49-60, 2012.
[36] K. Fall, "*A delay-tolerant network architecture for challenged internets*," in Proc. ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), Karlsruhe, Germany, 2003.
[37] S. Farrell,V. Cahill, D. Geraghty, I. Humphreys, P. McDonald, "*When TCP Breaks, Delay- and Disruption-Tolerant Networking*," IEEE Internet Computing, vol. 10, no. 4, pp. 72-78, 2006.
[38] H. Ochiai, H. Ishizuka, Y. Kawakami, H. Esaki, "*A field experience on DTN-based sensor data gathering in agricultural scenarios*," in Proc. IEEE Sensors, Kona, HI, USA, 2010.
[39] A. Tovar, T. Friesen, K. Ferens, B. McLeod, "*A DTN wireless sensor network for wildlife habitat monitoring*," in Proc. The 23rd Conference on Electrical and Computer Engineering (CCECE), Calgary, AB, Canada, 2010.
[40] H. Ochiai, H. Ishizuka, Y. Kawakami, H. Esaki, "*A DTN-Based Sensor Data Gathering for Agricultural Applications*," IEEE Sensors Journal, vol. 11, no. 11, pp. 2861-2868, 2011.
[41] J. Burgess, B. Gallagher, D. Jensen, B. Levine, "*MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks*," in Proc. 25th IEEE International Conference on Computer Communications. Proceedings (INFOCOM), Barcelona, Spain, 2006.
[42] A. Mashhadi, S. Mokhtar, L. Capra, "*Habit: Leveraging Human Mobility and Social Network for Efficient Content Dissemination in DTNs*," in Proc. IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks & Workshops (WoWMoM), Kos, Greece, 2009.
[43] C. Caini, H. Cruickshank, S. Farrell, M. Marchese, "*Delay- and Disruption-Tolerant Networking (DTN): An Alternative Solution for Future Satellite Networking Applications*," Proceedings of the IEEE vol. 99, no. 11, pp. 1980-1997, 2011.
[44] P. Muri, J. McNair, J. Antoon, A. Gordon-Ross, K. Cason, N. Fitz-Coy, "*User Datagram and Bundle Protocol for Distributed Small Satellite Topologies*," Journal of Wireless Networking and Communications, vol. 3, no. 3, pp. 19-28, 2013.
[45] M. Chitre, S. Shahabudeen, M. Stojanovic, "*Underwater Acoustic Communications and Networking: Recent Advances and Future Challenges*," Marine Technology Society Journal, vol. 42, no. 1, pp. 1-10, 2008.
[46] J. Partan, J. Kurose, B. Levine, "*A survey of practical issues in underwater networks*," ACM SIGMOBILE Mobile Computing and Communications Review, vol. 11, no. 4, pp. 23-33, 2007.

[47] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, H. Weiss, "*Delay-tolerant networking: an approach to interplanetary Internet*," IEEE Communications Magazine, vol. 41, no. 6, pp. 128-136, 2003.

[48] T. Berners-Lee, R. Fielding, L. Masinter, "*Uniform Resource Identifier (URI): Generic Syntax*," IETF RFC 3986. [Online]. Available: http://tools.ietf.org/html/rfc3986.txt, 2005.

[49] L. Clare, S. Burleigh, K. Scott, "*Endpoint Naming for Space Delay / Disruption Tolerant Networking*," in Proc. of IEEE Aerospace Conference, Big Sky, MT, USA, 2010.

[50] S. Patra, S. Saha, V. Shah, S. Sengupta, K. G. Singh, S. Nandi, "*A Qualitative Survey on Multicast Routing in Delay Tolerant Networks*," in Proc. The 3rd International Conference on Recent Trends in Wireless and Mobile Networks (WiMo) and on Computer Networks & Communications (CoNeCo), Ankara, Turkey, 2011.

[51] Y. Wang, X. Li, J. Wu, "*Multicasting in Delay Tolerant Networks: Delegation Forwarding*," in Proc. IEEE Global Telecommunications Conference (GLOBECOM), Miami, FL, USA, 2010.

[52] K. Srinivasan, P. Ramanathan, "*Reliable Anonymous Multicasting in Disruption Tolerant Networks*," in Proc. IEEE Global Telecommunications Conference (GLOBECOM), New Orleans, LO, USA, 2008.

[53] W. Gao, Q. Li, B. Zhao, G. Cao, "*Multicasting in Delay Tolerant Networks: A Social Network Perspective*," in Proc. 10th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc), New Orleans, LO, USA, 2009.

[54] M. Motani, V. Srinivasan, P. Nuggehalli, "*PeopleNet: engineering a wireless virtual social network*," In Proc. 11th ACM annual international conference on Mobile computing and networking (MobiCom), Cologne, Germany, 2005.

[55] M. Ramadas, S. Burleigh, S. Farrell, "*Licklider Transmission Protocol – Specification*," IETF RFC 5326. [Online]. Available: http://tools.ietf.org/html/rfc5326, 2008.

[56] G. Papastergiou, I. Psaras, and V. Tsaoussidis, "*Deep-Space Transport Protocol: A Novel Transport Scheme for Space DTNs*," Computer Communications (COMCOM), Special Issue on Delay-/Disruption-Tolerant Networks, vol. 32, no. 16, pp. 1757–1767, 2009.

[57] C. Samaras, V. Tsaoussidis, "*Design of Delay-Tolerant Transport Protocol (DTTP) and its evaluation for Mars*," Acta Astronautica, vol. 67, no. 7-8, pp. 863–880, 2010.

[58] M. Sarkar, K. Shukla, K. Dasgupta, "*A Survey of Transport Protocols for Deep Space Communication Networks*," International Journal of Computer Applications, vol. 31, no. 8, pp. 25-32, October 2011.

[59] C. Caini, R. Firrincieli, M. Livini, "*DTN Bundle Layer over TCP: Retransmission Algorithms in the Presence of Channel Disruptions*," Journal of Communications, vol. 5, no. 2, pp. 106-116, 2010.

[60] R. Wang, X. Wu, T. Wang, X. Liu, L. Zhou, "*TCP Convergence Layer-Based Operation of DTN for Long-Delay Cislunar Communications*," IEEE Systems Journal, vol. 4, no. 3, pp. 385-395, 2010.

[61] S. Jain, K. Fall, R. Patra, "*Routing in a Delay Tolerant Network*," in Proc. ACM conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), Portland, OR, USA, 2004.

[62] J. Dias, J. Rodrigues, J. Isento, P. R. Pereira, J. Lloret, "*Performance assessment of fragmentation mechanisms for vehicular delay-tolerant networks*," EURASIP Journal on Wireless Communications and Networking, 2011:195, 2011.

[63] M. Pitkanen, A. Keranen, J. Ott, "*Message Fragmentation in Opportunistic DTNs*," in Proc. International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Newport Beach, CA, USA, 2008.

[64] M. Demmer, J. Ott, S. Perreault, "*Delay Tolerant Networking TCP Convergence Layer Protocol*," IETF Internet Draft (Work in Progress). [Online]. Available: http://tools.ietf.org/html/draft-irtf-dtnrg-tcp-clayer-04, 2012.

[65] E. Kohler, M. Handley, S. Floyd, "*Datagram Congestion Control Protocol (DCCP)*," IETF RFC 4340. [Online]. Available: http://tools.ietf.org/html/ rfc4340.txt, 2006.

[66] H. Kruse, S. Jero, S. Ostermann, "*Datagram Convergence Layers for the DTN Bundle and LTP Protocols*," IETF Internet Draft (Work in Progress). [Online]. Available: http://tools.ietf.org/html/draft-irtf-dtnrg-dgram-clayer-02, 2013.

[67] F. Warthman, "*Delay- and Disruption-Tolerant Networks (DTNs), A Tutorial*," version 2, [Online] at: http://ipnsig.org/.

[68] D. Vardalis, V. Tsaoussidis, "*Achieving energy-efficiency with DTN: A proof-of-concept and roadmap study*," in Proc. The 10th Wired/Wireless Internet Communication (WWIC), Santorini, Greece, 2012.

[69] M. Viredaz, L. Brakmo, W. Hamburgen, "*Energy Management on Handheld Devices*," ACM Queue Magazine - Power Management, vol. 1, no. 7, 2003.
[70] D. Vardalis, V. Tsaoussidis, "*Exploiting the potential of DTN for energy-efficient internetworking*," Journal of Systems and Software, vol. 90, pp. 91-103, 2014.
[71] R. Scholtz, "*The Origins of Spread-Spectrum Communications*," IEEE Transactions on Communications, vol. 30, no. 5, pp. 822-854, 1982.
[72] R. Chang, "*Synthesis of band-limited orthogonal signals for multi-channel data transmission*," Bell System Technical Journal, vol. 45, no. 10, pp. 1775-1796, 1966.
[73] Wi-Fi Alliance. [Online] http://www.wi-fi.org/.
[74] A Jayasuriya, S Perreau, A Dadej, S Gordon, "*Hidden vs exposed terminal problem in ad hoc networks*," in Proc. Australian Telecommunication Networks and Applications Conference (ATNAC), Bondi Beach, NSW, Australia, 2004.
[75] S. Chandra, A. Vahdat, "*Application-specific network management for energy-aware streaming of popular multimedia formats*," in Proc. The General Track of the annual conference on USENIX, Monterey CA, USA, 2002.
[76] J. Adams, G.M. Muntean, "*Adaptive-Buffer Power Save Mechanism for Mobile Multimedia Streaming*," in Proc. IEEE International Conference on Communications, Glasgow, UK, 2007.
[77] A. Acquaviva, E. Lattanzi, A. Bogliolo, "*Design and Simulation of Power-Aware Scheduling Strategies of Streaming Data in Wireless LANs*," in Proc. The 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM), Venice, Italy, 2004.
[78] H. Zhu, G Cao, "*A Power-Aware and QoS-Aware Service Model on Wireless Networks*," in Proc. 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Hong Kong, 2004.
[79] G. Anastasi, M. Conti, E. Gregori, A. Passarella, L. Pelusi, "*A Power-Aware Multimedia Streaming Protocol for Mobile Users*," in Proc. The IEEE International Conference on Pervasive Services (ICPS), Santorini, Greece, 2005.
[80] J. Korhonen and Y. Wang, "*Power-efficient streaming for mobile terminals*," in Proc. The ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video, Skamania, WA, 2005.
[81] N. Balasubramanian, A. Balasubramanian, A. Venkataramani, "*Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications*," in Proc. of ACM/USENIX Internet Measurement Conference, Chicago, IL, USA, 2009.
[82] G. Anastasi, M. Conti, W. Lapenna, "*A Power-Saving Network Architecture for Accessing the Internet from Mobile Computers: Design, Implementation and Measurements*," The Computer Journal, vol. 46, no. 1, pp. 3-15, 2003.
[83] A. Bakre, B. R. Badrinath, "*Implementation and Performance Evaluation of Indirect TCP*," IEEE Transaction on Computers, vol. 6, no. 3, pp. 260-278, 1997.
[84] I. Batsiolas, I. Nikolaidis, "*Selective Idling: Experiments in Transport Layer Energy Conservation*," The Journal of Supercomputing, vol. 20, no. 2, pp. 101-114, 2001.
[85] L. Mamatas, V. Tsaoussidis, "*Exploiting Energy-Saving Potential in Heterogeneous Networks*," International Journal of Parallel, Emergent and Distributed Systems (IJPEDS), Taylor & Francis, vol. 23, no. 4, pp. 309-324, 2008.
[86] R. Prasad, M. Murray, C. Dovrolis, K. Claffy, "*Bandwidth estimation: metrics, measurement techniques, and tools*," IEEE Network, vol. 17, no. 6, pp. 27-35, 2003.
[87] S. Keshav, "*A control-theoretic approach to flow control*," in Proc. ACM Conference on Communications Architecture & Protocols (SIGCOMM), Zürich, Switzerland, 1991.
[88] M. Bredel, M. Fidler, "*A measurement study of bandwidth estimation in IEEE 802.11g wireless LANs using the DCF*," in Proc. The 7th International IFIP-TC6 Networking Conference, Singapore, 2008.
[89] K. Lakshminarayanan, V. Padmanabhan, J. Padhye, "*Bandwidth estimation in broadband access networks*," in Proc. The 4th ACM SIGCOMM Conference on Internet Measurement (IMC), Taormina, Sicily, Italy, 2004.
[90] C. Sarr, C. Chaudet, G. Chelius, I. G. Lassous, "*Bandwidth Estimation for IEEE 802.11-based Ad Hoc networks*," IEEE Transactions on Mobile Computing, vol. 7, no. 10, pp. 1228-1241, 2008.
[91] H. K. Lee, V. Hall, K. H. Yum, K. I. Kim, E. J. Kim, "*Bandwidth Estimation in Wireless Lans for Multimedia Streaming Services*," Hindawi Publishing Corporation Advances in Multimedia, vol. 2007.

[92] D. Gupta, D. Wu, P. Mohapatra, C. Chuah, "*Experimental Comparison of Bandwidth Estimation Tools for Wireless Mesh Networks*," in Proc. The 27th IEEE Conference on Computer Communications. (INFOCOM), Rio de Janeiro, Brazil, 2008.
[93] M. Nielsen, K. Ovsthus, L. Landmark, "*Field trials of two 802.11 residual bandwidth estimation methods*," in Proc. the IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), Vancouver, BC, Canada, 2006.
[94] G. Wu and T. Chiueh, "*Passive and accurate traffic load estimation for infrastructure-mode wireless LAN*," in Proc. The 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems (MSWiM), Chania, Greece, 2007.
[95] H. J. Kramer, "*Earth Observation Remote Sensing. Survey of Missions and Sensors*," Springer-Verlag, 1992.
[96] M. Folkman, J. Pearlman, L. Liao, P. Jarecke, "*EO1/Hyperion hyperspectral imager design, development, characterization and calibration*," in Proc. The 40th SPIE Conference on Hyperspectral Remote Sensing of the Land and Atmosphere, Sendai, Japan, 2001.
[97] D. Vane, G. L. Stephens, "*The CloudSat Mission and the A-Train: A Revolutionary Approach to Observing Earth's Atmosphere*," in Proc. The IEEE Aerospace Conference, Big Sky, MT, USA, 2008.
[98] F. Colomb, C. Varotto, "*SAC-C and the AM constellation: three years of achievements*," in Proc. IEEE International Conference on Recent Advances in Space Technologies, Istanbul, Turkey, 2003.
[99] C. Iacopino, P. Palmer, N. Policella, A. Donati, A. Brewer, "*Self-Organizing MPS for Dynamic EO Constellation Scenarios*," in Proc. International Workshop on Planning and Scheduling for Space (IWPSS), NASA-Ames, CA, USA, 2013.
[100] G. Tyc, J. Tulip, D. Schulten, M. Krischke, M. Oxfort, "*The Rapideye Mission Design*," Acta Astronautica, vol. 56, no. 1-2, pp. 213–219, 2005.
[101] L. Wood, "*Satellite Constellation Networks*," Chapter 2 of "Internetworking and Computing over Satellite Networks," Kluwer Academic Publishers, Y. Zhang, pp 13-34, 2003.
[102] M. Witting et al., "*Status of the European Data Relay Satellite System*," in Proc. International Conference on Space Optical Systems and Applications (ICSOS), Ajaccio, France, 2012.
[103] "PACKET TELEMETRY," CCSDS 102.0-B-5, Blue Book, November 2000.
[104] "TELECOMMAND", CCSDS 201.0-B-3, Blue Book, June 2000.
[105] "AOS SPACE DATA LINK PROTOCOL", CCSDS 732.0-B-2, Blue Book, July 2006.
[106] "CCSDS FILE DELIVERY PROTOCOL (CFDP)," CCSDS 727.0-B-2, Blue Book, October 2002.
[107] W. Ivancic, W. Eddy, L. Wood, D. Stewart, C. Jackson, J. Northam, A. Curiel, "*Delay/Disruption-Tolerant Network Testing Using a LEO Satellite*," in Proc. NASA Earth Science Technology Conference (ESTC), University of Maryland, MD, USA, 2008.
[108] "*Call for CubeSat Proposals for QB50*," [Online], https://www.qb50.eu/call_proposals_QB50.pdf
[109] K. Leveque, J. Puig-Suari, C. Turner, "*Global Educational Network for Satellite Operations (GENSO)*," in Proc. The 21st AIAA/USU Conference on Small Satellites (SSC), Technical Session XI: Educational Programs, Logan, UT, USA, 2007.
[110] Innovative Solutions In Space. [Online]. http://www.isispace.nl.
[111] H. Bedon, C. Miguel, A. Fernandez, J. S. Park, "*A DTN System for Nanosatellite-based Sensor Networks using a New ALOHA Multiple Access with Gateway Priority*," Smart Computing Review, vol. 3, no. 5, pp. 383-396, 2013.
[112] T. Paila, R. Walsh, M. Luby, V. Roca, R. Lehtonen, "*FLUTE - File Delivery over Unidirectional Transport*," IETF RFC 6726. [Online]. Available: http://tools.ietf.org/rfc/rfc6726.txt, 2007.
[113] M. Luby, M. Watson, L. Vicisano, "*Asynchronous Layered Coding (ALC) Protocol Instantiation*," IETF RFC 5775. [Online]. Available: http://tools.ietf.org/rfc/rfc5775.txt, 2010.
[114] B. Adamson, C. Bormann, M. Handley, J. Macker, "*NACK-Oriented Reliable Multicast (NORM) Transport Protocol*," IETF RFC 5740. [Online]. Available: http://tools.ietf.org/rfc/rfc5740.txt, 2009.
[115] P. Basu, K. Kanchanasut, "*A reliable multicast protocol for unidirectional satellite link*," in Proc. Symposium on Applications and the Internet (SAINT), Orlando, FL, USA, 2003.
[116] European Space Agency (ESA). Earth Online. [Online]. Available: https://earth.esa.int/web/guest/pi-community.
[117] National Aeronautics and Space Administration (NASA). NASA Earth Observations (NEO). [Online]. Available: http://neo.sci.gsfc.nasa.gov.

[118] M. Goetzelmann, V. Tsaoussidis, S. Diamantopoulos, I.A. Daglis, T. Amanatidis, B. Ghita, "*Space Data Routers for the Exploitation of Space-Data*," in Proc. The 12th International Conference on Space Operations (SpaceOps), Stochholm, Sweden, 2012.
[119] C. Perkins, E. Royer, "*Ad hoc on-demand distance vector routing*," in Proc. The 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), New Orleans, LA, USA, 1999.
[120] D. Johnson, D. Maltz, "*Dynamic source routing in ad hoc wireless networks*," Chapter 5 in "Mobile Computing", T. Imielinski, H. F. Korth, Kluwer Academic Publishers, pp. 153-181, 1996.
[121] A.Vahdat, D. Becker, "*Epidemic routing for partially connected ad hoc networks*," Technical Report CS-200006, Duke University, 2000.
[122] A. Lindgren, A. Doria, "*Probabilistic Routing Protocol for Intermittently Connected Networks*," Internet Draft. [Online]. Available: http://tools.ietf.org/html/draft-irtf-dtnrg-prophet, 2010.
[123] T. Spyropoulos, K. Psounis, C. Raghavendra, "*Spray and wait: An efficient routing scheme for intermittently connected mobile networks*," in Proc. ACM SIGCOMM workshop on Delay-tolerant networking (WDTN), Philadelphia, PA, USA, 2005.
[124] X. Chen, J. Shen, J. Wu, "*Improving routing protocol performance in delay tolerant networks using extended information*," The Journal of Systems and Software, vol. 83, no. 8, pp. 1301-1309, 2010.
[125] J. Ott, D. Kutscher, C. Dwertmann, "*Integrating DTN and MANET Routing*," in Proc. The 2006 SIGCOMM workshop on Challenged networks (CHANTS), Pisa, Italy, 2006.
[126] J. Whitbeck, V. Conan, "*HYMAD: Hybrid DTN-MANET Routing for Dense and Highly Dynamic Wireless Networks*," Computer Communications, vol. 33, no. 13, pp. 1483-1492, 2010.
[127] T. Small, Z. Haas, "*Resource and Performance Tradeoffs in Delay-Tolerant Wireless Networks*," in Proc. ACM SIGCOMM workshop on Delay-tolerant networking (WDTN), Philadelphia, PA, USA, 2005.
[128] M. Seligman, K. Fall, P. Mundur, "*Storage routing for DTN congestion control*," Wireless Communications & Mobile Computing, vol. 7, no. 10, pp. 1183-1196, 2007.
[129] R. El-Azouzi, F. De Pellegrini, V. Kamble, "*Evolutionary forwarding games in Delay Tolerant Networks*," in Proc. The 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), Avignon, France, 2010.
[130] R. Axelrod, "*Effective Choice in the Prisoner's Dilemma*," The Journal of Conflict Resolution, vol. 24, no. 1, pp. 3-25, 1980.
[131] L. Buttyan, L. Dora, M. Felegyhazi, I. Vajda, "*Barter-based cooperation in delay-tolerant personal wireless networks*," in Proc. IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Espoo, Finland, 2007.
[132] R. Lu, X. Lin, H. Zhu, X. Shen, B. Preiss, "*Pi: A Practical Incentive Protocol for Delay Tolerant Networks*," IEEE Transactions on Wireless Communications, vol. 9, no. 4, pp. 1483-1493, 2010.
[133] T. Henderson, M. Lacage, G. Riley, "*Network Simulations with the ns-3 Simulator*," ACM Special Interest Group on Data Communication (SIGCOMM), Seattle, WA, USA, 2008.
[134] J. Lakkakorpi, M. Pitkänen, J. Ott, "*Adaptive Routing in Mobile Opportunistic Networks*," in Proc. The 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems (MSWiM), Bodrum, Turkey, 2010.
[135] J. Lakkakorpi, P. Ginzboorg, "*ns-3 Module for Routing and Congestion Control Studies in Mobile Opportunistic DTNs*," in Proc. International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), Toronto, ON, Canada, 2013.
[136] A. Petz, J. Enderle, C. Julien, "*A Framework for Evaluating DTN Mobility Models*," in Proc. Workshop on Scenarios for Network Evaluation Studies (SCENES), Rome, Italy, 2009.
[137] T. Issariyakul, E. Hossain, "*Introduction to Network Simulator NS2*," Springer Science+Business Media, LLC, 2009.
[138] A. Varga, "*Parameterized Topologies for Simulation Programs*," in Proc. of the Western Multiconference on Simulation (WMC), Communication Networks and Distributed Systems (CNDS), San Diego, CA, 1998.
[139] G. van Rossum, "*Python tutorial*," Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, 1995.
[140] L. Franck, F. Potorti, "*GaliLEO: a simulation tool for LEO satellite constellations*," Transactions of The Society for Modeling and Simulation International (SIMULATION), vol. 78, no. 9, pp. 543-551, 2002.

[141] B. Smalarz, "*Cubesat Constellation Analysis for Data Relaying*," M.S. Thesis, Dept. Aerospace Eng., California Polytechnic State University, San Luis Obispo, CA, USA, 2011.
[142] MATLAB and Statistics Toolbox, The MathWorks, Inc., Natick, Massachusetts, United States.
[143] T. R. Henderson, R. H. Katz, "*Network Simulation for LEO Satellite Networks*," in Proc. The 18th AIAA International Communications Satellite Systems Conference (ICSSC), Oakland, CA, USA, 2000.
[144] J. Liang, N. Xiao, J. Zhang, "*Constellation design and performance simulation of LEO satellite communication system*," in Proc. International Conference on Applied Informatics and Communication (ICAIC), Xi'an, China, 2011.
[145] P. Muri, J. McNair, "*Topology design and performance analysis for networked earth observing small satellites*," in Proc. Military Communications Conference (MILCOM), Baltimore, MD, USA, 2011.
[146] P. Muri, J. McNair, "*Simulating Delay Tolerant Networking for CubeSats*," in Proc. The 5th Interplanetary CubeSat Workshop (iCubeSat), Boston, MA, USA, 2012.
[147] K. Fall, W. Hong, S. Madden, "*Custody Transfer for Reliable Delivery in Delay Tolerant Networks*," Intel Research Berkeley, Tech. Rep. IRB-TR-03-030, 2003.
[148] D. Vardalis, V. Tsaoussidis, "*Energy-efficient internetworking with DTN*," Journal of Internet Engineering, vol. 5, no. 1, pp. 345-354, 2012.
[149] C. K. Yeoa, B. S. Leea, M. H. Erb, "*A survey of application level multicast techniques*," Computer Communications, vol. 27, no. 15, pp. 1547-1568, 2004.
[150] A. Cockburn, "*Agile Software Development*," Addison-Wesley Longman Publishing Co., Inc. Boston, MA, 2002.
[151] E. Shih, P. Bahl, M. Sinclair, "*Wake on Wireless: An Event DrivenEnergy Saving Strategy for Battery Operated Devices*", in Proc. The 8th ACM International Conference on Mobile Computing and Networking (MobiCom), Atlanta, GA, USA, 2002.
[152] Y. Xiao, P. Savolainen, A. Karppanen, M. Siekkinen, A. Ylä-Jääski, "*Practical power modeling of data transmission over 802.11g for wireless applications*," in Proc. The 1st ACM International Conference on Energy-Efficient Computing and Networking (e-Energy), Passau, Germany, 2010.
[153] K. Jamieson, "*Implementation of a Power-Saving Protocol for Ad Hoc Wireless Networks*," Master Thesis, Dept. of Computer Science and Engineering, MIT, 2002.
[154] R. Krashinsky, H. Balakrishnan, "*Minimizing energy for wireless web access with bounded slowdown*," Journal of Wireless Networks, vol. 11, no. 1-2, pp. 135-148, 2005.
[155] P. McKenney, "*Stochastic fairness queueing*," in Proc. The 9th Annual Joint Conference of the IEEE Computer and Communication Societies (INFOCOM), San Francisco, CA, 1990.
[156] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, "*Hypertext Transfer Protocol -- HTTP/1.1*," IETF RFC 2616. [Online]. Available: http://tools.ietf.org/html/rfc2616.txt, 1999.
[157] E. Zegura, K. Calvert, S. Bhattacharjee, "*How to Model an Internetwork*," in Proc. The 15th Annual Joint Conference of the IEEE Computer Societies (INFOCOM), San Francisco, CA, 1996.
[158] D. Knuth, "*The Stanford GraphBase: A Platform for Combinatorial Computing*," New York: ACM Press, 1994.
[159] B.M. Waxman, "*Routing of multipoint connections*," IEEE Journal on Selected Areas in Communications, vol. 6, no. 9, pp. 1617-1622, 1988.
[160] D. Vardalis, "*On the Efficiency and Fairness of TCP over Wired/Wireless Networks*," Master Thesis, Dept. of Computer Science, State University of New York at Stony Brook, 2001.
[161] M. Sarkarati, M. Merri, K. Nergaard, P. Steele, "*How to plug-in your rover into a space mission to moon or mars*", in Proc. The 6th International Conference on Automation, Robotics and Applications (ICARA), Queenstown, New Zealand, 2015.
[162] A. Jenkins A, S. Kuzminsky, K. Gifford, R.L. Pitts, K. Nichols, "*Delay/Disruption-Tolerant Networking: Flight test results from the international space station*," IEEE Aerospace Conference, Big Sky, MT, USA, 2010.