The MicroOppnet Tool for Collaborative Computing Experiments with Class 2 Opportunistic Networks

Zille Huma Kamal, Ajay Gupta, Leszek Lilien,[1] and Zijiang Yang
The WiSe (Wireless Sensornets) Lab
Department of Computer Science
Western Michigan University
Kalamazoo, Michigan, USA
{zkamal, gupta, llilien, zijiang}@cs.wmich.edu

Abstract—**Class 2 opportunistic networks (oppnets) are a new paradigm for Collaborative Computing that aims at integrating communication, computation, sensing, actuation, storage, and other resources and services. Oppnets achieve global tasks and goals through the collaboration and coordination of their nodes (some of which join an oppnet dynamically). We describe the concept of oppnets, discuss related work, and present the standard API framework for oppnets name Oppnet Virtual Machine (OVM). We present the design and implementation details of a small-scale proof-of-concept system, named MicroOppnet, in terms of the OVM primitives. We describe both the design and implementation of MicroOppnet, which not only is a proof of concept but also constitutes our tool for experiments in collaborative computing with oppnets. We are currently working on extending MicroOppnet into a larger oppnet prototype and an oppnet testbed.**

Keywords-opportunistic networks; ad hoc networks; collaborative computing; collaborative communication; pervasive computing

I. Introduction

Over the years, technologies enabling pervasive computing have been researched and experimented with to meet demands for ubiquitous communication, computation, data processing, storage, etc. Various technologies have emerged, such as grid networks, mesh networks, ambient networks, and, most recently, opportunistic networks.

In what we call *class 1 opportunistic networks*, opportunism is quite restricted, usually limited to opportunistic connectivity that is, establishing communications when devices are within each other's range. In contrast, we proposed a new paradigm and a new technology called *class 2 opportunistic networks* or *oppnets* to enable not only opportunistic communications but also an opportunistic growth of networks and opportunistic use of resources gained by this growth [1]. (Oppnets evolved from our initial idea of opportunistic sensor networks [2]).

Effectively, oppnets leverage their capabilities by exploiting the wealth of resources available on all kinds of pervasive devices that are within their reach—crossing communication, hardware and software barriers. This integration of resources, fundamental to oppnets, can only be achieved through collaboration. Hence, oppnets are a new paradigm for collaborative computing networks.

In this paper we discuss oppnets and present design and implementation of MicroOppnet v.2.2, a small-scale experimental tool for oppnets, which is also a proof of concept for them.

The paper is organized as follows. Section II discusses the basic components and operations of oppnets. Section III includes an overview of work related to opportunistic networks. Section IV briefly describes an oppnet implementation framework, called the Oppnet Virtual Machine (OVM), which provides a standard API for oppnet-based systems. Section V presents the structure and operation of MicroOppnet v.2.2. Section VI shows how an oppnet, even as small as MicroOppnet, can assist in emergency response operations. Sections VII and VIII discuss the design and implementation of MicroOppnet v.2.2, respectively. Section IX gives a list of collaborative computing experiments (incl. collaborative communication experiments) for the MicroOppnet tool (and for the future testbed). Section X concludes with summarizing our contributions and showing plans for planned work on the MicroOppnet tool and the future testbed.

II. Basic Components and Operations of Oppnets

Each oppnet grows from a *seed oppnet*, or simply a *seed* (cf. Fig. 1), which is a set of nodes employed together at the time of the initial oppnet deployment. The seed is predesigned (so it can be viewed as a network in its own right). It can contain just a few nodes, in the extreme having a single node.

A seed can be wireless—with nodes communicating via radio channels, and ad hoc—with nodes not carefully pre-positioned but, for instance, randomly deployed during an emergency to achieve specific emergency response goals. The goals include providing connectivity—which is the only goal for class 1 opportunistic networks—as well as computing capabilities, sensing or monitoring, storage, and any imaginable

[1] Affiliated with the Center for Education and Research in Information Assurance and Security (CERIAS), Purdue University, West Lafayette, IN.

resources or services (e.g. a capability to recognize victims in images captured by surveillance cameras).

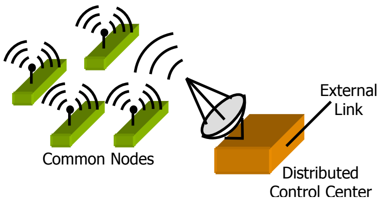

Figure 1. Seed oppnet

After the seed self-configures and becomes operational, its first task is to detect a set of both reserve and outside entities—devices, node clusters, networks, and other systems—which it deems useful. Reserve entities or *reservists* are those that are "trained" (in an analogy to, e.g., Army Reserve), i.e., preconfigured with oppnet primitives and protocols. Such "training" facilitates their discovery and use by any oppnet that might need them. In contrast, outside entities or outsiders are those that do not have any helpful primitives or protocols assisting in their discovery and use by oppnets.

Identified reservists or detected outsiders are *candidates* for becoming oppnet helpers. Each viable candidate must have a potential to provide an oppnet with some communication, computing, sensing, or other capabilities or resources. Candidates are evaluated by an oppnet considering their use, and those that are seen by the oppnet as potentially helpful are either invited to join the oppnet, or may be ordered to join in some cases.

The controversial issue of ordering entities to join is discussed elsewhere [4, 17]. We can briefly explain here that: (i) any reservists can be ordered at any time; and (ii) any non-reservist candidate can be ordered but only in life-or-death emergencies. Not following an order to join means going AWOL with all accompanying legal and other consequences. In contrast, an invited candidate can either accept or refuse the invitation to join, risking at most being labeled as selfish.

.

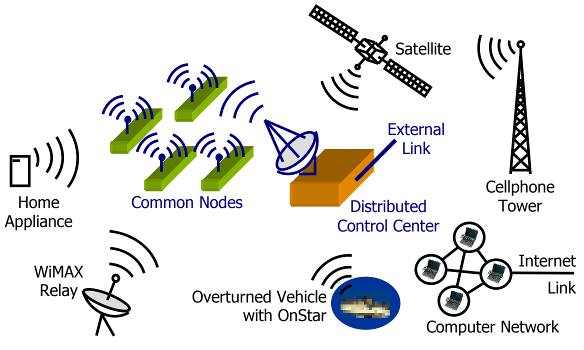

Figure 2. Expanded oppnet

A candidate that joins an oppnet becomes its *helper*. All helpers collaborate with their oppnet on realization of the oppnet's goals. They can be deployed to execute all kinds of tasks even though, in general, they were not designed to become elements of the oppnet that invited them. A helper may be allowed by an oppnet manager to invite other entities. With more helpers allowed to invite outsiders, the oppnet can grow faster.

Figure 2 shows an expanded oppnet that grew from the seed shown in Figure 1. The growth involved detection and admitting the following candidates that became helpers: (a) a computer network, contacted via a wired Internet link; (b) a cellphone infrastructure (represented in Fig. 2 by Cellphone Tower), contacted via Bluetooth-enabled oppnet's cellphone peripheral; (c) a satellite, contacted via a direct link; (d) a home area network (HAN), contacted via an intelligent appliance (e.g., a refrigerator) with a wireless link; (e) a WiMAX network, contacted via a WiMAX relay; (f) BANs (body area networks) on or within bodies of occupants of a car, contacted via an OnStar™ network.

The mechanism to leverage resources and services available in the environment, in order to enhance effectiveness or efficiency of an oppnet, is the salient and fundamental feature of oppnets, i.e., class 2 opportunistic networks.

Such an expanded network would be beneficial for oppnets deployed in emergencies or disaster response and relief. In such situations, HANs and BANs, for example, could be used mostly to provide sensing in search for survivors, and other helpers could be used to provide connectivity, computing power, specialized capabilities (e.g., image recognition), etc.

The basic sequence of oppnet operations is summarized in Figure 3. The operations are supervised by a distributed controller that can be autonomous or human-controlled.

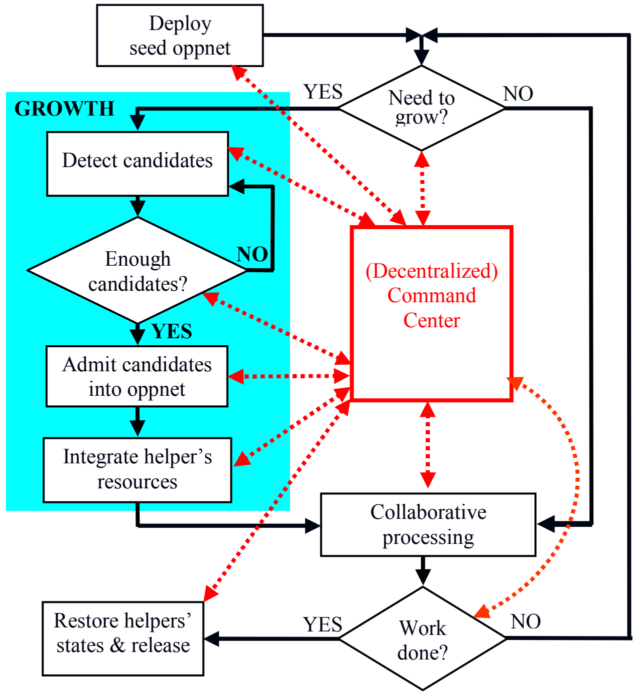

Figure 3. Basic operations of an oppnet

The oppnet command center presides over the operations of the oppnet throughout its life. If the oppnet needs more resources for to achieve its goal, the cycle of oppnet growth (including detecting, evaluating, admitting, and integrating candidates) is repeated. Once the goal of an oppnet is achieved, the helpers are assisted in restoring the state they were in before joining oppnet and are released.

III. Work Related to Opportunistic Networks

In this section, we review some collaborative computing technologies—such as resource-sharing, connectivity-based and other emerging 3G networks—to show, on the one hand, their relevance and similarity to oppnets and, on the other hand, to delineate distinctions between these technologies and oppnets.

A. Resource-sharing Technologies

There is no clear or standard definition for grid or peer-to-peer (P2P) systems [24, 25]. However, in the broader sense grid computing refers to the hardware and software infrastructure that enables aggregation of distributed computing resources in pursuit of common goals [24]. There is a thin line that demarcates grid networks from P2P systems, which share content—such as files—between devices, networks or systems (peers) that interact via an "appropriate communication and information channel" [25].

Grid and P2P systems are typical resource-sharing technologies. We have examined the relationship between P2P and oppnets [5]. The fundamental differences between these resource-sharing systems and oppnets are: (a) the collaboration of devices, networks, or systems in the oppnet to achieve a global goal, rather than meeting individual node requirements; and (b) the integration of all sorts of resources, such as communication, sensing, actuation, storage, etc. rather than the typical resource-sharing of computing cycle and information.

B. Connectivity-based Technologies

Mesh networks are geared towards providing continuous communication with redundant paths to overcome node failures while sharing the communication media to provide cost-effective Internet access via distributed gateways [31]. Research in area of mesh networks focuses on interaction between devices that have been configured a priori and in a deterministic manner. This distinguishes mesh networks from oppnets, since in general at least the helper oppnet nodes are not preconfigured, and there is no determinism in the manner that the helper nodes are called to service.

Amongst other connection-oriented networks/systems are dynamic interconnection networks (DINs), spontaneous networks [26], and delay-tolerant networks or disruption-tolerant networks (DTNs) [27]. Providing connectivity is essential in any functioning network, including DINs and DTNs that provide connectivity in the most challenging circumstances. However, the oppnets accomplish a much larger task. In addition to providing connectivity they integrate resources of the component networks/systems. They grow into larger networks in order to collaboratively achieve their goals.

Class 1 opportunistic networks (e.g., [3]), a proper subset of oppnets, can be viewed as a generalization of the mobile ad hoc networking (MANET) paradigm, in which the assumption of complete paths between data senders and receivers is relaxed [16]. This enables stations to communicate in disconnected environments, in which island of connected stations appear, disappear, and reconfigure dynamically [16]. In class 1 opportunistic networks, there is no notion of utilizing resources of the nodes in a network to perform a network task. In contrast, class 2 opportunistic networks (oppnets) not only provide a communication backbone but can provide computing, sensing, actuating, storage, or other resources or services. Also, oppnets can grow dynamically by admitting needed helpers, which facilitates execution of more challenging tasks. Such tasks would either be beyond capabilities of systems based on traditional networks, or would be much more difficult (even in class 1 opportunistic networks). As will be shown in Section IV, oppnets provide high-level primitives facilitating building of complex applications.

DTNs can be viewed as the superclass including all kinds of wireless networks, the broadest networking paradigm. DTNs are characterized as those with intermittent communication due to mobility or use of energy-conserving sleep states, with high latency paths and low allowed data rates [15, 18], and, foremost, use of store-and-forward message routing [30].

Summarizing, we view opportunistic technologies as ranging from class 1 opportunistic networks [3] to class 2 opportunistic networks or oppnets [1, 5, 17]. Opportunistic data dissemination techniques [10, 11, 12] might be considered "class 1.5" opportunistic networks.

Connection-oriented networks can form the underlying layer for oppnet operations.

C. Specialized and Other Networks

Ambient Networks (AN) [22, 28, 29] are sponsored by the European Commission, as an interconnection network geared towards 3G networks and beyond. The vision of AN is to enable "cooperation of heterogeneous networks belonging to different operator or technology domains." The ability of ANs to include smaller ANs and manage resources through a "control space" resembles how oppnets will integrate devices, networks or systems and manage resources and oppnet through a distributed command center. However, the following features distinguish the two efforts:

- AN is a global, universal network intended as a replacement for the Internet and all communication networks, whereas oppnet is a local/wide area network which is serving few specific applications.
- AN requires heavyweight primitives whereas oppnet requires no primitives or only lightweight ones.
- AN is completely predesigned and is aware of the location of all sub-ANs, all its facilities are built-in or add-on, only networks that have the needed primitives can be "composed" into ambient networks, whereas oppnet is mostly ad hoc that has to discover helpers, with a possibility of lightweight components being built-in or add-on.

- ANs contact each other mutually, so that any sub-AN can initiate connection, whereas in oppnets the push mechanism (where seed oppnet nodes initiate discovery of devices) is predominant.

Wireless sensor networks (WSNs)—in which sensor network nodes are equipped with sensors, processors and transceivers—are well-established specialized networks. Their characteristics can be summarized as follows:

- resource-constrained – in terms of memory, computation power and energy,
- random or deterministic deployment,
- configuration and reconfiguration, in face of dynamic topology,
- power management critical since sensor network nodes are often battery-powered.

The major differences between oppnets and WSNs include the following. First, oppnet nodes are heterogeneous devices that can be powerful computing devices, not just resource-constrained nodes. Second, communications in oppnets is not over a single frequency channel, which is typical in sensornets.

IV. Oppnet Virtual Machine

From the point of view of an application architect and a developer wishing to use oppnets, the ultimate goal of our work is to provide a standard implementation framework, which we called the Oppnet Virtual Machine (OVM) [4]. OVM will allow developing standard library routines and APIs to be used for implementing all kinds of oppnet-based systems. OVM will not only facilitate application development but will also assure interoperability among different oppnet implementations and third-party oppnet products.

A subset of seed nodes constitutes a distributed Control Center (CC). CC can grow admitting other nodes as helpers, and can shrink expelling any of its nodes. We can have both regular helpers and *lightweight helpers* or *lites* (such as a smoke detector). Lites are oppnet-enabled, that is equipped with inexpensive, simple means of standard oppnet communications. In this way, even lites can be triggered to operate in the oppnet mode when needed and commanded to do so by a CC Regular helpers, but not lites, can discover and may admit other helpers.

For illustration, Tables I–IV delineate selected OVM primitives defined by us for four categories of oppnet nodes, namely for control center (CC) nodes, seed nodes, helpers and lites.

TABLE I. PARTIAL LIST OF OVM PRIMITIVES FOR CC NODES

Name of the Primitive	Functions of the Primitive
CTRL_initiate	Initiate oppnet
CTRL_terminate	Terminate oppnet
CTRL_command	Send commend to seed nodes

TABLE II. PARTIAL LIST OF OVM PRIMITIVES FOR SEED NODES

Name of the Primitive	Functions of the Primitive
SEED_scan	Scan communication spectrum to detect devices that could become candidate helpers
SEED_discover	Discover candidate helpers with a specific communication mechanism
SEED_listen	Receive and save messages in buffer
SEED_validate	Verify the received command
SEED_isMember	Checks if a device is already an oppnet node (oppnet member)
SEED_evaluateAdmit	Evaluate a device and admit it into oppnet if the device meets criteria for admittance
SEED_sendTask	Send a task to other oppnet device
SEED_delegateTask	Delegate a task that requires a permission from the delegating entity
SEED_release	Release a helper when no longer needed
SEED_processMsg	Process a message from buffer
SEED_report	Report information to control center/coordinator
SEED_update	Update a device in the oppnet with new expectations
SEED_receiveTask	Receive task from control center or another seed
SEED_wait	Wait for a certain amount of time become take another action
SEED_barrier	Block the caller until all devices specified in the input parameter have called it

TABLE III. PARTIAL LIST OF OVM PRIMITIVES FOR HELPERS

Name of the Primitive	Functions of the Primitive
HLPR_isMember	Test if a helper is already a member of oppnet
HLPR_joinOppnet	Join oppnet
HLPR_scan	Scan communication spectrum to detect devices that could become candidate helpers (regular or lites)
HLPR_discover	Discover candidate helpers with a specified communication mechanism
HLPR_validate	Verify the received command
HLPR_switchMode	Switch between helpers' regular application and oppnet application
HLPR_report	Send information/data to specified device
HLPR_selectTask	Select a task from the task queue to execute
HLPR_listen	Receive message and save it
HLPR_evaluateAdmit	Evaluate a candidate helper and admit it into oppnet if it meets criteria defined by oppnet
HLPR_runApplication	Execute application indicated by authorized oppnet seed or helper node
HLPR_release	Release a helper (unless delegated a release task, a helper H can release only helpers admitted by H)
HLPR_processMsg	Process a message from buffer
HLPR_sendData	Send information/data to specified authorized oppnet node
HLPR_leaveOppnet	Inform a seed that the caller will quit oppnet
HLPR_strongTask	Respond to the request sent from device and express the willingness to join oppnet. By accepting this task, the device will abort previous task
HLPR_weakTask	Respond to the request sent from device and express the willingness to join oppnet. By accepting this task, the device will put the task in a queue
HLPR_assignStrongTask	Assign tasks to a device. If accepted, the task will interrupt the previous task at the device
HLPR_assignWeakTask	Assign tasks to a device. If accepted, the task will be queued

TABLE IV. PARTIAL LIST OF OVM PRIMITIVES FOR LITES

Name of the Primitive	Functions of the Primitive
LITE_isMember	Test if a lit is already a member of oppnet
LITE_joinOppnet	Join oppnet
LITE_validate	Verify the received command
LITE_switchMode	Switch between lites' regular application and oppnet application
LITE_report	Send information/data to specified device
LITE_selectTask	Select a task from the task queue to execute
LITE_listen	Receive message and save it
LITE_runApplication	Execute application indicated by authorized oppnet seed or helper node
LITE_processMsg	Process a message from buffer
LITE_sendData	Send information/data to specified authorized oppnet node
LITE_leaveOppnet	Inform a seed that the caller will quit oppnet
LITE_strongTask	Respond to the request sent from device and express the willingness to join oppnet. By accepting this task, the device will abort previous task
LITE_weakTask	Respond to the request sent from device and express the willingness to join oppnet. By accepting this task, the device will put the task in a queue

V. OVERVIEW OF MICROOPPNET

The current version of MicroOppnet, v.2.2, is a small-scale proof of concept for *class 2* opportunistic networks, since it not only allows opportunistic communications but also opportunistically accesses sensornet nodes to perform sensing. It is, though, rudimentary in its class 2 opportunism, hence the prefix *micro* in the name MicroOppnet. (The prefix also indicates the small size of this implementation.)

MicroOppnet is a platform on which functional parameters, such as, oppnet components (including OVM primitives), protocols, and architectures are or will be implemented, tested, and fine-tuned. Non-functional parameters, including quality-of-service (QoS) parameters, such as throughput, delay, reliability, accuracy, scalability, etc., can also be measured or investigated on MicroOppnet.

MicroOppnet v.2.2 integrates three separate communication media and frequency ranges: (a) Bluetooth (*BT*) at 2.4 GHz; (b) a sensor network at 916/433MHz and (c) wireless Internet standards 802.11b and 802.11g, both working on the same 2.4 GHz [19] frequency as BT.

The seed oppnet in MicroOppnet, shown in Fig. 4 consists of Workstation *A* with a Bluetooth (*BT*) adapter and a serial port connection to Sensornet Base Station *BS_1*. All other indicated devices are candidates for helpers or actual helpers, depending on the scenario.

The seed searches for BT devices and initiates a connection with them. Alternatively, a BT-enabled device—a cellphone labeled *Victim* in our example—can find the seed and initiate a connection. Once a connection has been established, the *Victim* cellphone can send a message to the seed, for example, the `help` message. This message is then forwarded via Base Station *BS_1*, and then through the sensor network. The sensornet consists of 10 Mica2 Motes and 6 Stargates, which are sensornet gateways. Some of the gateways are also connected to Mica2 Motes.

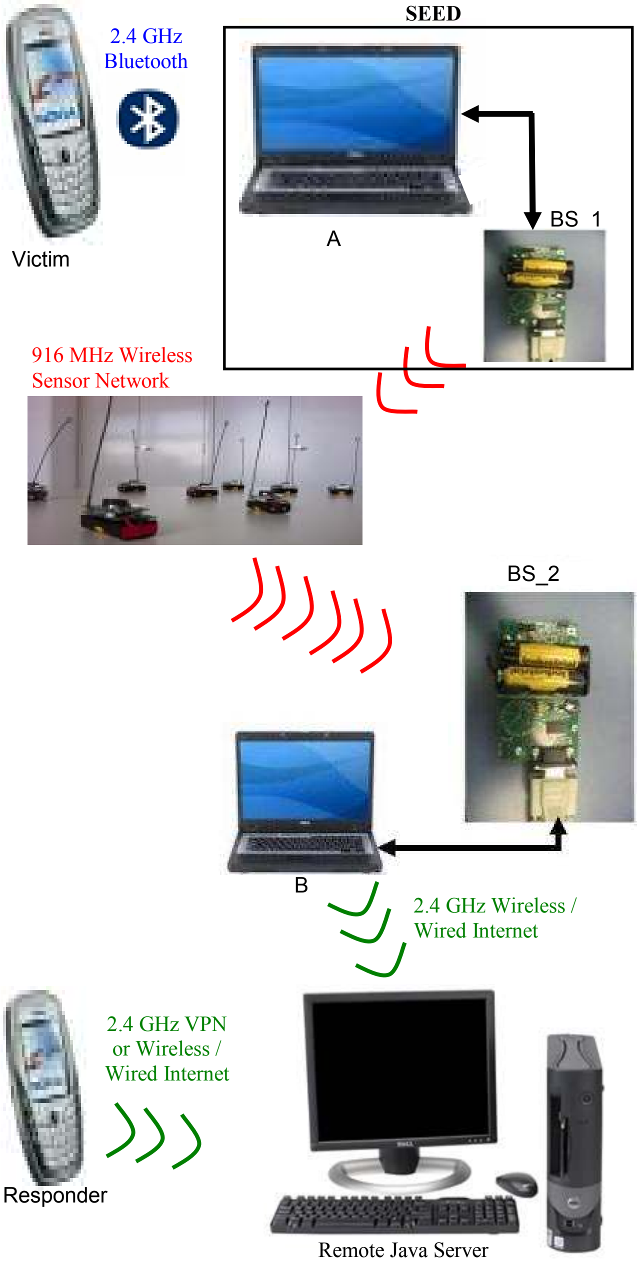

Figure 4. Structure of MicroOppnet v.2.2

Base Station *BS_2* at the other end of the sensornet is connected to Laptop *B*. Once the `help` message is propagated via *BS_2* to Laptop *B*, a Java TCP/IP client socket connection is initiated with a remote Java server. The `help` message and the location of the device that sent it are logged on this server.

The Java server can be queried by remote users employing either traditional computing devices or Java-enabled devices. In our example, we employ cellphone with T-Mobile™ Virtual Private Network (VPN) connection, labeled as *Responder*, which can inquire the remote Java server via VPN.

The seed can broadcast to BS_1 belonging to the sensornet a variety of messages in addition to help—e.g., start_sensing, log_sensing, retrieve_log. The messages can be used, for instance, to start temperature sensing, to log temperature in the EEPROM of the sensor, or to retrieve the logged data from the sensor network. The retrieved temperature readings can be logged at the Java server. Then, they can either be queried by remote users via wireless Internet, or be broadcast by the seed on the BT channels.

VI. Sample Oppnet Emergency Scenario

To illustrate use of MicroOppnet, let us consider an emergency scenario, namely a fire in a large office building.

Suppose that a few workers were unable to evacuate. Most of them tried to use their cellphones to call for help. Many succeeded but many failed to get a connection since the cellphone infrastructure is overloaded with calls being made by thousands of workers still gathered outside of the building.

The firefighters can put MicroOppnet (or, maybe, MiniOppnet) to use. They start with deploying a MicroOppnet seed around the office building. It consists of laptops and networks connecting them. The Bluetooth Class 1 connectivity becomes an essential communications capability, with the MicroOppnet using it to discover all kinds of BT-enabled helpers. (BT Class 1 has nothing to do with class 1 opportunistic networks. Devices of BT Class 1 have the range of approx. 100meters [19].) An owner of any such helper, that is an owner of a BT-equipped cellphone, PDA, laptop, etc., is now able to communicate with the firefighters via the extended MicroOppnet (consisting of the seed MicroOppnet plus all helpers that joined it).

We have so far used only class 1 opportunistic capabilities of the MicroOppnet. To show how class 2 opportunistic capabilities of the MicroOppnet can be used, suppose that the MicroOppnet is now commanded to contact and query for temperature readings all sensing nodes within the building. These temperature readings, aggregated at a Java server, are used to plot the heat profile for the building. The profile, together with location information gathered by BT-enabled helpers before, can be used by the firefighters to find the best routes for reaching the workers trapped by fire in the building.

Please note that many other pervasive communications technologies could be used in parallel with BT (but our example should be complete enough without discussing them).

VII. Design of MicroOppnet

In this section, we present the flow of control for the MicroOppnet of Figure 4 in terms of the OVM primitives of Section IV. This flow of control, illustrated in Figure 5, can begin with three modes of operation: (a) DISCOVER – an active discovering of candidates—using SEED_discover; (b) LISTEN – with a passive wait—using SEED_listen, when candidates search for and initiate connection with the seed; or (c) SENSORNET OPERATION – with dispatching a task for the sensornet—using SEED_sendTask. In v.2.2, communication for (a) and (b) is only over the Bluetooth medium.

Messages received from nodes wishing to use MicroOppnet are processed, and tasks are delegated to the appropriate helpers. In v.2.2, there are only two sets of helpers: the set of nodes in the sensornet, and the remote server.

Messages from a user such as Victim in Figure 4 can be forwarded from the seed's sensornet base station (SBS) BS_1 to the helpers using the SEED_sendTask primitive. The nodes in the sensornet process the message using HLPR_processMsg and then perform the task (currently, only sensing or communication) using HLPR_runApplication. If the task is sensing, then the sensornet nodes (SNNs) will start or stop sensing as required. Otherwise, they will forward either the received message or their temperature sensor readings as directed. When the message is received by another sensornet gateway or another base station (e.g., by BS_2), it is logged on a remote server. If the task was to retrieve sensor-measured temperature, then BS_2 aggregates sensornet readings and floods the result back through the sensornet to BS_1. Since SNNs lack a display mechanism, to observe the flow of control/operation/messages we use the blinking of the three LEDs—yellow, red, and green—on the SNNs to signal receipt of messages and the types of messages received.

Devices such as *Responder* (cf. Fig. 4) can send the message retrieve_log to the remote helper server, which is in a listening mode with the HLPR_listen primitive. This allows the remote server's log to be queried for specific tasks and retrieve the appropriate messages. The server can process any TCP/IP socket connection.

Summarizing, MicroOppnet v.2.2 supports only the following tasks: (i) communication tasks – flooding messages and retrieving sensor readings; and (ii) sensing tasks – starting and stopping sensing. All these tasks rely on opportunism. In more detail, the following is the exhaustive list of all tasks using resources opportunistically:

i. Communication in the BT medium
ii. Communication in the sensornet medium
iii. Communication using TCP/IP in wired or wireless Internet
iv. Temperature sensing using sensornet nodes

The first three tasks use class 1 opportunism, and only the last task relies on class 2 opportunism—by leveraging the sensing resources of MicroOppnet helpers. Thanks to the last task, we can claim that MicroOppnet is a *class 2* opportunistic network, albeit a rudimentary one (exploiting only *one* type of non-communication resources).

VIII. Implementation of MicroOppnet

A USB Bluetooth dongle equips the seed with a BT infrastructure. To exploit the BT communication framework, we use the BT software protocol stack provided by Atinav

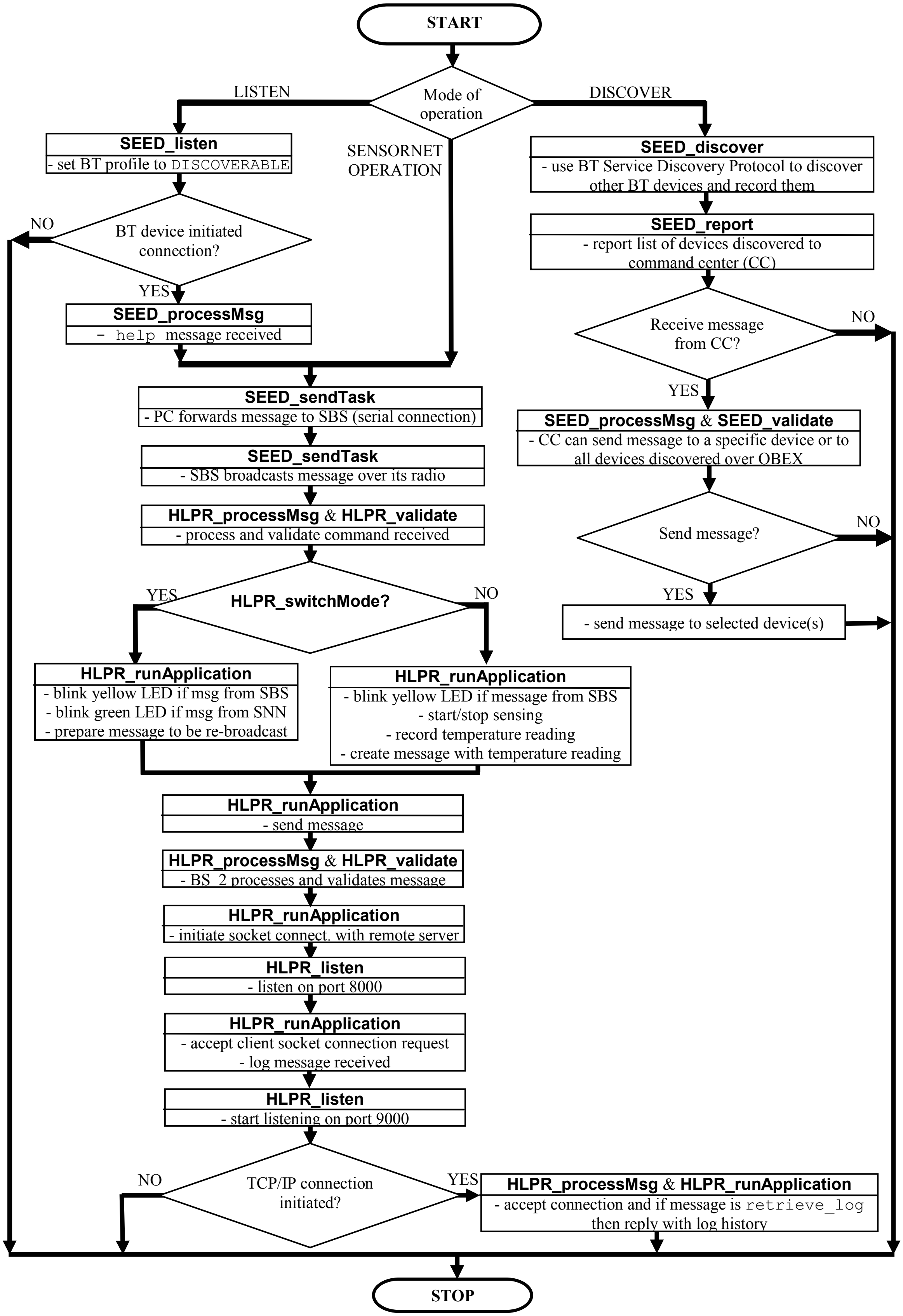
START
Mode of operation
LISTEN
DISCOVER
SENSORNET OPERATION
SEED_listen
- set BT profile to DISCOVERABLE
BT device initiated connection?
NO
YES
SEED_processMsg
- help message received
SEED_discover
- use BT Service Discovery Protocol to discover other BT devices and record them
SEED_report
- report list of devices discovered to command center (CC)
Receive message from CC?
NO
YES
SEED_processMsg & SEED_validate
- CC can send message to a specific device or to all devices discovered over OBEX
Send message?
NO
YES
- send message to selected device(s)
SEED_sendTask
- PC forwards message to SBS (serial connection)
SEED_sendTask
- SBS broadcasts message over its radio
HLPR_processMsg & HLPR_validate
- process and validate command received
HLPR_switchMode?
YES
NO
HLPR_runApplication
- blink yellow LED if msg from SBS
- blink green LED if msg from SNN
- prepare message to be re-broadcast
HLPR_runApplication
- blink yellow LED if message from SBS
- start/stop sensing
- record temperature reading
- create message with temperature reading
HLPR_runApplication
- send message
HLPR_processMsg & HLPR_validate
- BS_2 processes and validates message
HLPR_runApplication
- initiate socket connect. with remote server
HLPR_listen
- listen on port 8000
HLPR_runApplication
- accept client socket connection request
- log message received
HLPR_listen
- start listening on port 9000
TCP/IP connection initiated?
NO
YES
HLPR_processMsg & HLPR_runApplication
- accept connection and if message is retrieve_log then reply with log history
STOP

Figure 5. Flow of control in MicroOppnet v.2.2

AveLink [6]. In this way, we can invoke the BT Service Discovery Protocol (SDP) using the API from the protocol stack to detect BT devices, and to either initiate connections with BT devices or to receive connections from BT devices.

The BT communication infrastructure consists of profiles that are built on top of layers/protocols to define further high-level functionality. There are numerous profiles that exist and, moreover, there are close dependencies between profiles. The lowest-level profile that most common BT Profiles are dependent on is the Generic Access Profile, which is used to establish a basic connection. After establishing an initial connection, we use Generic Object Exchange Profile, which uses the Object Exchange (OBEX) layer to exchange objects. Alternatively, we can use Logical Link Control and Adaptation Protocol (L2CAP) and RFCOMM protocol (uses Serial Port Profile) for packet and stream data, respectively [19].

Our sensor network consists of Crossbow's Mica2 Motes and Stargate gateways [7]. The Mica2 Motes run UC Berkley's TinyOS [8] operating system, and are programmed with nesC [9]. The nesC code is compiled on a workstation and is flushed onto the Motes using Crossbow's programming boards.

The remote server, developed in Java using socket connections, runs on a Linux machine. Its flow of control is illustrated in Figure 6.

Cellphone programming is accomplished with Java MicroEdition (J2ME) and JSR-118 Mobile Information Device Profile (MIDP) 2.0 for resource-constrained devices, such as cellphones and PDAs. Java applications for such devices are called *MIDlets*. We use Java-enabled phones: Nokia 6600 (equipped with Symbian OS), Nokia 6103, and Motorola RAZR.

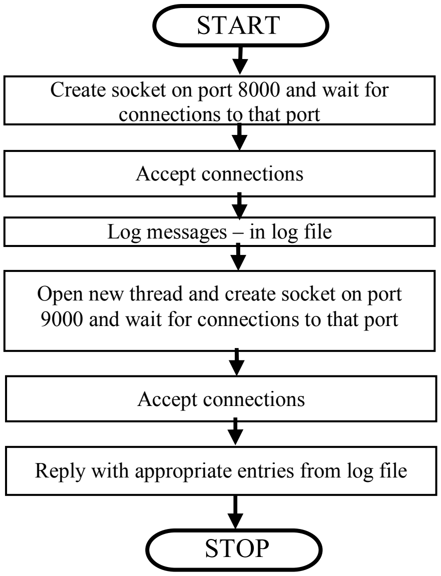
START
Create socket on port 8000 and wait for connections to that port
Accept connections
Log messages – in log file
Open new thread and create socket on port 9000 and wait for connections to that port
Accept connections
Reply with appropriate entries from log file
STOP

Figure 6. Flow of control for the remote Java server

Figure 7 illustrates the flow of control in the MIDlet of the *Victim* and *Responder* stations (cf. Figure 4). We found that Nokia 6600 is stronger than the other two models when it came to initiating BT connections with the seed or TCP/IP connections with the server.

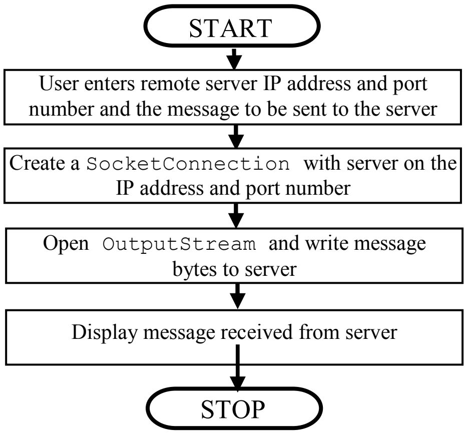
START
User enters remote server IP address and port number and the message to be sent to the server
Create a SocketConnection with server on the IP address and port number
Open OutputStream and write message bytes to server
Display message received from server
STOP

Figure 7. Flow of control for cellphone MIDlets

Currently, MicroOppnet v.2.2 uses MANET routing, in which every node in the oppnet is a router. We gained valuable insights on vulnerabilities of MANET routing in an opportunistic network. Consequently, we plan to scrutinize routing protocols developed for opportunistic networks and DTNs, e.g., the ones described in References [3, 13, 14, 15, 18] to mention a few. We will evaluate them against the set of vulnerabilities and a number of very specific and low-level criteria that we have identified while implementing MicroOppnet and experimenting with it.

During the development of MicroOppnet, we learnt that Nokia 6600 was a much more robust cellphone (probably due to its Symbian OS), since MIDlets that created sockets or streams were not allowed on Nokia 6103 (locked by T-Mobile) and Motorola RAZR was not equipped to receive text messages over Bluetooth. We observed that T-Mobile WAP and GPRS connections do not allow unrestricted access to the Internet. Instead T-Mobile Virtual Area Network (VPN) was used to allow unrestricted MIDlet access to the Internet. Furthermore, we observed that the remote server should not be behind any firewall to allow MIDlet access to the server.

IX. MicroOppnet Experiments in Progress

The following simulations and experiments in the area of collaborative computing (incl. collaborative communication) are in progress or planned for the MicroOppnet v.2.2 tool (and the future testbed):

1. Studying the impact of interference from environmental factors on oppnet connectivity (incl.

collaborative connectivity) and collaborative networking tasks.

2. Studying the impact of link failures on connectivity and routing in oppnets, and providing algorithms for collaborative means of counteracting them.
3. Devising algorithms for collaborative detection and localization of candidate nodes, clusters, or networks.
4. Proposing algorithms for collaborative identification of suspicious or inefficient oppnet nodes or clusters and removing them when necessary (even the members of the original seed oppnet can be "fired").
5. Devising collaborative controls for selected aspects of helper privacy and oppnet security.

X. Conclusions

We have described *class 2 opportunistic networks* or *oppnets* and contrasted them with less opportunistic *class 1 opportunistic networks*. We detailed the design and implementation of a small-scale oppnet named MicroOppnet. It not only serves as a proof of concept but is also being extended to become a prototype as well as a testbed for designing, testing and implementing oppnet primitives; routing, privacy and security protocols; and architectures.

Oppnets can be used as a new paradigm of collaborative computing, supporting and facilitating collaborations in a vast variety of applications, ranging from the most critical disaster recovery to the most mundane domestic applications.

Our future work on MicroOppnet and its successors will include: (a) extending its rudimentary class 2 opportunism of MicroOppnet to a substantial class 2 opportunism—by implementing the opportunistic growth mechanisms of class 2 opportunistic networks fully and for multiple types of resources; (b) developing comprehensive privacy and security controls [20, 1, 17]; (c) incorporating opportunistic routing protocols (e.g. [3, 14, 15, 18]); (d) extending MicroOppnet to a medium scale either by increasing the number of communication media used (adding, among others, WiMAX), or by increasing the number of resource kinds that can be leveraged (e.g., including computation and storage); (e) stress-testing MicroOppnet on the DETER Lab facilities (funded by HSARPA and operated by the Information Sciences Institute at USC), and, later, as a part of GENI [32]; and (f) developing a Rapid Application Development (RAD) [21] environment for oppnets and testing it on our testbed.

Acknowledgment

This research was supported in part by the NSF Grants IIS-0242840 and IIS-0209059, and in part by the U.S. Department of Commerce under Grant BS123456. The authors would also like to acknowledge Computational Science Center and Information Technology and Image Analysis (ITIA) Center. Any opinions, finding, conclusions or recommendation expressed in the paper are those of the authors and do not necessarily reflect the views of the funding agencies or institutions.

References

[1] L. Lilien, Z. H. Kamal, V. Bhuse, and A. Gupta, "Opportunistic Networks: The Concept and Research Challenges in Privacy and Security," *Proc. International Workshop on Research Challenges in Security and Privacy for Mobile and Wireless Networks (WSPWN '06)*, Miami, FL, Mar. 2006, pp. 134-147.

[2] B. Bhargava, L. Lilien, A. Rosenthal and M. Winslett, "PervasiveTrust," *IEEE Intelligent Systems*, vol. 19(5), Sep./Oct.2004, pp. 74-77.

[3] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic Networking: Data Forwarding in Disconnected Mobile Ad Hoc Networks," *IEEE Communications*, Vol. 44(11), Nov. 2006, pp. 134-141.

[4] L. Lilien, A. Gupta, and Z. Yang "Opportunistic Networks for Emergency Applications and Their Standard Implementation Framework," *Proc. 1st Intl. Workshop on Next Generation Networks for First Responders and Critical Infrastructure (NetCri07)*, New Orleans, LA, Apr. 2007**,** pp. 588-593.

[5] L. Lilien, Z. H. Kamal and A. Gupta, "Opportunistic Networks: Research Challenges in Specializing the P2P Paradigm," *Proc. 3rd Int. W. on P2P Data Management, Security and Trust (PDMST'06)*, Kraków, Poland, Sep. 2006, pp. 722-726.

[6] Atinav, 2006. Online: http://www.atinav.com

[7] Crossbow Technology Inc., 2006. Online: http://www.xbow.com/

[8] UC Berkley, 2004. Online: http://www.tinyos.net/

[9] nesC: A Programming Language for Deeply Networked Systems. Dec, 2004. Online: http://nescc.sourceforge.net/

[10] P. Sistla, O. Wolfson and B. Xu, "Opportunistic Data Dissemination in Mobile Peer-to-Peer Networks," *9th Intl. Sym. on Advances in Spatial and Temporal Databases (SSTD 05)*, Angra dos Reis, Brazil, Aug. 2005

[11] O. Wolfson and B. Xu, "Opportunistic dissemination of spatio-temporal resource information in mobile peer to peer networks," *15th Intl. W. on Database and Expert Systems Applications (DEXA 04)*, Zaragoza, Spain, Aug. 2004

[12] B. Xu, A. Ouksel and O. Wolfson, "Opportunistic resource exchange in inter-vehicle ad-hoc networks," *IEEE Proc. Intl. Conference on Mobile Data Management (MDM 04)*, Berkley, California, Jan. 2004

[13] S. Jain, K. Fall and R. Patra, "Routing in a Delay Tolerant Network," *ACM conference of the Special Interest Group on Data Communication (SIGCOMM 04)*, Portland, Oregon, Aug. 2004

[14] J. Park, D. Lun, Y. Yi, M. Gerla and M. Medard, "CodeCast: A Network Coding based Ad hoc Multicast Protocol," *IEEE Wireless Communications*, Vol. 13(5), 2006.

[15] K. Fall, "A delay-tolerant network architecture for challenged internets," *ACM conference of the Special Interest Group on Data Communication (SIGCOMM 03)*, Karlsruhe, Germany, Aug. 2003

[16] Y. Wang, S. Jain, M. Martonosi and K. Fall, "Erasure-Coding Based Routing for Opportunistic Networks," *ACM Conf. of the Special Interest Group on Data Communication (SIGCOMM 2005)*, Philadelphia, PA, Aug. 2005

[17] L. Lilien, Z. H. Kamal, V. Bhuse, and A. Gupta, "The Concept of Opportunistic Networks and Their Research Challenges in Privacy and Security," book chapter in: Mobile and Wireless Network Security and Privacy ed. by K. Makki et al., Springer Science+Business Media, Norwell, Massachusetts, 2007 (to appear)

[18] E. Magistretti, J. Kong, U. Lee, M. Gerla, P. Bellavista and A. Corradi, "A Mobile Delay-tolerant Approach to Long-term Energy-efficient Underwater Sensor Networking," *IEEE Wireless Communications and Networking Conference (WCNC 07)*, Hong Kong, Mar. 2007

[19] B. Hopkins and R. Anthony, Bluetooth for Java, Apress, 2003

[20] W. Cheswick and S. Bellovin, Firewalls and Internet Security, 2nd ed., Addison-Wesley, 2002.

[21] RAPIDware: Component-Based Development of Adaptable and Dependable Middleware. Online: http://www.cse.msu.edu/~mckinley/rapidware/

[22] Ambient Networks. Online, last accessed July 15, 2007, http://www.ambient-networks.org/

[23] L. Lilien, "A Taxonomy of Specialized Ad Hoc Networks and Systems for Emergency Applications," *The First Intl. Workshop on Mobile and Ubiquitous Context Aware Systems and Applications (MUBICA 2007)*, Philadelphia, PA, August 2007, CD-ROM, 8 pages.

[24] A. Abbas, "Grid Computing: A Practical Guide to Technology and Applications," Charles River Media, Hingham, MA, 2004.

[25] R. Subramanian, and B. Goodman, "Peer-to-Peer Computing: The Evolution of a Disruptive Technology" Idea Group Publishing, Hershey, PA, 2005.

[26] L.M. Feeney, B. Ahlgren, and A. Westerlund, "Spontaneous Networking: An Application-oriented Approach to Ad Hoc Networking," *IEEE Communications Magazine*, 39 (6), June 2001, pp. 176-181.

[27] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, H. Weiss, "Delay-Tolerant Network Architecture," DTN Research Group Internet Draft, March 2003.

[28] N. Niebert, A. Schieder, H. Abramowicz, G. Malmgren, J. Sachs, U. Horn, C. Prehofer, H. Karl, "Ambient Networks – An Architecture for Communication Networks Beyond 3G," *IEEE Wireless Communications*, Special Issue on 4G Mobile Communications – Towards Open Wireless Architecture, April 2004.

[29] B. Ahlgren, L. Eggert, B. Ohlman, and A. Schieder, "Ambient Networks: Bridging Heterogeneous Network Domains," *The 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, Berlin, Germany, Sep. 2005.

[30] M. Seligman, K. Fall, and P. Mundur, "Storage Routing for DTN Congestion Control," *Journal on Wireless Communications & Mobile Computing*, Special Issue on Disruption Tolerant Networking for Mobile or Sensor Networks, January 2007.

[31] Networking Research Group, "Self-Organizing Neighborhood Wireless Mesh Networks." Online, last accessed on July 17, 2007, http://research.microsoft.com/mesh/

[32] Global Environment for Network Innovations. Online, last accessed on July 17, 2007, http://www.geni.net/index.html