Using peer-to-Peer Data Routing for Infrastructure-Based Wireless Networks [*]

Sethuram Balaji Kodeswaran, Olga Ratsimor, Anupam Joshi, Tim Finin, Yelena Yesha
Department of Computer Science and Electrical Engineering,
University of Maryland Baltimore County
1000 Hilltop Circle, Baltimore, MD 21250
(kodeswar, oratsi2, joshi, finin, yeyesha) @cs.umbc.edu

Abstract

A mobile ad-hoc network is an autonomous system of mobile routers that are self-organizing and completely decentralized with no requirements for dedicated infrastructure support. Wireless Infrastructure in terms of base stations is often available in many popular areas offering high-speed data connectivity to a wired network. In this paper, we describe an approach where infrastructure components utilize passing by mobile nodes to route data to other devices that are out of range. In our scheme, base stations track user mobility and determine data usage patterns of users as they pass by. Based on this, base stations predict the future data needs for a passing mobile device. These base stations then collaborate (over the wired network) to identify other mobile devices with spare capacity whose routes intersect that of a needy device and use these carriers to transport the needed data. When such a carrier meets a needy device, they form ad hoc peer-to-peer communities to transfer this data. In this paper, we describe the motivation behind our approach and the different component interactions. We present the results of simulation work that we have done to validate the viability of our approach. We also describe, Numi, our framework for supporting collaborative infrastructure and ad hoc computing along with a sample application built on top of this highlighting the benefits of our proposed approach.

1. Introduction

Recent years have seen a tremendous proliferation of mobile computing devices. Devices such as Personal Digital Assistants (PDAs) have undergone constant improvements and are today full fledged computers with improved processing power, multimedia capabilities etc. Fueled by such advances, new and improved services and applications are beginning to be offered on them. Wireless networking has also witnessed remarkable growth in recent years starting from traditional voice-centric cellular technologies such as TDMA to more recent data-centric wireless LAN technologies like 802.11b. Also, there has been a lot of advances in the field of short-range wireless communication technologies facilitating personal area networks (PANs) like Bluetooth, homeRF, IEEE 802.15 etc. These technologies have enabled the creation of ad-hoc peer-to-peer networking capabilities on mobile devices. Now it is possible for these devices to discover peers, form transient peer communities to exchange information and gracefully handle changes in their neighborhood.

Ad-hoc networks do not require any infrastructure support. However, such infrastructure in the form of base stations is already available in many popular areas. In addition, there is growing popularity in community wireless networks. Companies like T-Mobile, Personal Telco, Seattle Wireless and Consume maintain *Community Networks* that offer network connectivity to mobile devices in metropolitan areas. For example, T-Mobile provides network access in over a thousand locations, including airports, airline clubs and Starbucks (Borders Bookstores will soon join the long list). This results in the creation of wireless networks with pockets of network connectivity (close to access points) surrounded by regions of no connectivity (or very expensive WAN connectivity).

A great majority of currently available mobile devices have restrictions such as limitations on power consumption, smaller display, processing power etc. that hinders a smooth migration of PC based applications to such devices. In addition, current wireless infrastructures suffer from limitations of restricted range, low bandwidth, limited coverage, higher costs etc. Services that are offered on wireless mobile devices must be aware of these limitations and must efficiently overcome them [4]. With the increased popularity of multimedia, financial applications etc., the amount of data required by these services is also on the rise. Managing these

[*]This work was supported in part by NSF awards IIS 9875433, ISS 0209001 and CCR 0070802, and the Defense Advanced Research Projects Agency under contract F30602-00-2-0591 AO K528.

users' data needs requires intelligent data transfer capabilities to and from the device and effective utilization of both the device's capabilities and communication infrastructure. In a wireless environment, constant network access cannot be guaranteed and using the cellular network as a WAN is associated with prohibitive costs.

In this paper, we present our approach to managing the data needs of services running on wireless mobile devices by using peer-to-peer data routing. The network model considered is one with islands of high-speed network connectivity surrounded by regions with no network access [cellular WAN is available but too expensive]. In our scheme, mobile devices within the range of an access point, can obtain relatively high-speed network access through it, while other mobile devices out-of-range, resort to ad-hoc peer-to-peer collaboration to satisfy their data needs. Our distinguishing feature is that our access points, by analyzing mobility patterns of users, can determine the future data needs of these users along with the time and the location of the user when such a need arises. Using this information, our access points collaborate among themselves (over a high-speed wired infrastructure) to identify other mobile devices whose mobility patterns indicate that they are likely to be in the vicinity of the needy device at that point in the future. If such a carrier device can be identified and if it has excess capacity, then the access points attempt to piggyback the data intended for the needy device onto this carrier. We claim that by doing this, in the future, when the needy device does in fact require the data and is not near any access point, peer-to-peer collaboration with neighboring peer devices will be able to satisfy the data needs. The access points in our approach, thus use mobile devices with spare capacity to route data to their peers who may not be in range of any access point.

Application Scenario: *Walking by a Starbucks, Bob decides to listen to some music. He turns on his PDA and connects to the nearby Service Portal (SP) at Starbucks and downloads a suitable playlist. The Starbucks SP determines from Bobs PDA (through Bob's appointment book) that Bob is walking to his work, four blocks south. The Starbucks SP determines that Bob's PDA can hold only the first five songs in the playlist, which will last Bob only for the next two blocks. The Startbucks SP also infers that there is another Starbucks along Bobs predicted route, three blocks down. The first Starbuck provides Bob with his initial set of songs and informs the second Starbucks of Bob's imminent data needs. The second Starbuck waits for few minutes and starts looking for a device with excess capacity heading north and finds Susan's PDA. A few minutes later Bob and Susan cross each other but at this time, Bob's PDA detects that it is running out of songs and starts querying its peers for the needed songs. In this instance, Bob's PDA conveniently finds Susan's PDA carrying the songs needed and downloads through a peer-to-peer exchange. Bob, meanwhile, is completely unaware of these interactions and continues to listen to his songs uninterrupted. (The second Starbucks SP waits a few minutes before looking for a carrier so that the chosen carrier runs into Bob just about the time when Bob's PDA is going to start querying its neighbors).*

In the following sections we will discuss related work, our network model, present Numi, our framework for peer-to-peer data routing along with a prototype application and the simulation work that we have done study the viability of our approach.

2. Related Work

Infostation networks[14] have often been suggested as a viable alternative to meet the needs of mobile applications. An infostation network consist of a set of towers offering short-range high bandwidth radio coverage, which is inherently low cost. Network access is available to users that are passing in close proximity. In this sense, the infostation is similar to a basestation coupled with an info-server such that the basestation provides the network connectivity while the info-server handles the data requests. A mobile device thus experiences areas of connectivity (when close to a infostation) and areas of disconnection (when there is no infostation nearby). Specialized data link protocols have been suggested for allowing devices to communicate with such Infostations[6].

Several data management models have been suggested for these types of infrastructure based systems. In [7], the infostations are owned by small enterprises with low speed wireline connection between them. Assuming that a mobile user's path is known, the data management issues tackled deal with identifying how to divide data into segments and how to transmit different segments to different infostations along the path so that a users data demands are satisfied. As users move with constant or variable velocity, they are in range of a particular infostation only for a short duration of time. Data segments need to be correctly sized so that an infostation can deliver its segment to a passing user before the user goes out of range. The segments are also sized taking into consideration time it takes to deliver them to their respective infostation. Other models have suggested using infostations (or access points distributed throughout the network) to facilitate data hoarding on mobile devices[15]. Knowing a users path, a mobile device at an infostation attempts to cache as much data as needed till the device reaches the next infostation. Once the device leaves the infostation, subsequent user data needs are satisfied by its local cache. On a cache miss, the device needs to connect to the WAN (or cellular network) to retrieve the desired data. Using user's usage profiles and context, hoarding decisions can be made so that only the most relevant data is hoarded on the mobile device and the number of hoard misses is minimized.

Infostations have often been thought of as information kiosks. In WICAT system[1], users select items of interest on their mobile device and when their device passes by an infostation, it attempts to download items available at that infostation that match a users preferences. Other applications deal with downloading context aware information. [16] deals with a map-on-the-go application. Here as users move between infostations, each infostation serves out relevant maps to these users. Other applications have focused on using infostations to advertise local attractions.

Existing approaches to mobile data management using infrastructure support such as infostations do not take into account a mobile device's capabilities when offering services. Data hoarding mechanisms dictate that a device should cache enough data until the device reaches the next access point or infostation. This is not feasible for devices with limited storage capabilities (cell phones, PDAs etc). Also many popular applications (like multimedia applications) deal with sufficiently large data volumes that could be too large to handle for mobile devices. Existing schemes treat infostations purely as oasis of information. Devices check-in with the infostation upon arrival and obtain the relevant data (enough to last until the next infostation). Such a model is intolerant to changes in a users expected travel plans like route deviations and travel delays. Also, current models do not attempt to share load among devices that are traveling together or in close proximity. Each device in that group may end up hoarding the same data. This is inefficient and wasteful especially considering devices with limited capabilities and amounts of information used by todays' applications.

Unlike infrastructure networks, ad-hoc networks are completely decentralized, require no infrastructure support and are autonomous and dynamic in nature. Spontaneous collaborations in the ad-hoc communities are used for data management in ad-hoc networks. Many models have suggested that ad-hoc communities share data to achieve common goals on behalf of a user. Other models have suggested using ad-hoc collaborations to trade tasks[5]. In systems such as [9], whenever different devices meet, they exchange personal profiles on behalf of their users. Through this, devices can selectively share data based on users' interests and characteristics. A common theme in most of the models of pure ad-hoc interactions seems to be "ask around when data is needed and hopefully a peer has the needed data". However, in these models, devices carry only information needed by the owner. The needs of other peer devices are not explicitly acknowledged. There is no "community network" type of notion prevalent in traditional ad-hoc networks. In addition, most MANET routing protocols assume that a user's mobility pattern is purely random[12, 11, 10, 8]. This assumption is not really valid in real life as users usually do have more or less predictable mobility patterns and in addition, using a user's personal information (such as appointment calendar), we can infer future user movements.

Our approach is essentially based on a merger of the two types of networks. We believe that by using the infrastructure(access points) to monitor a user's mobility patterns, the infrastructure components can predict where and when a user may require some data. This information can be used to identify peer mobile devices that are likely to meet this device at that time. Peer devices are asked to carry this data (provided they have storage capacity available for this request). This way, when these devices meet, the needy device would most likely be looking for the data in its neighborhood and the carrier would "conveniently" show up carrying that needed data. We are proposing a "community adhoc network" notion on mobile devices, i.e. excess capacity on a mobile device can be used to route data to other devices in need. Through this, even rather simple devices with limited data storage can offer advanced data intensive services to a user by virtue of the fact that its peers are routing the data needed by this device and this whole collaboration is being orchestrated by the fixed infrastructure that has been tracking a device's mobility and usage patterns. (A simple charge/credit model can be built on top of this to encourage users to offer their devices for use by the network).

3. Network Model

The network model that we are considering are islands of high-speed wireless connectivity surrounded by regions of low or no network access. We envision that devices that are within these islands have access to an infrastructure component (access point) while in surrounding areas, only ad-hoc communication is possible between neighboring peer devices. The key components of our network include *Service Portals* (*SPs*), *Mobile Hosts* (*MHs*), and *Services*. *SPs* are infostations offering high-speed network connectivity and hosting services that can be used by nearby *MHs*. These *SPs* are connected by high speed links to the rest of the wireline network as broadband connectivity has become cheaper and more ubiquitous. The *SPs* use their wireless capabilities to interact with *MHs* that are in range and use their wireline connectivity to communicate among themselves. This model of disjoint areas of coverage is quite realistic. In fact, increasing popularity of *community networks* and their commercial deployments (Starbucks offer connectivity in their kiosks and in most metropolitan areas, one encounters a Starbuck every few blocks thereby creating a network of pockets of network access separated by a distance of a few blocks) is adding more credence to the viability of this network model. Our *SPs* are more intelligent than conventional infostation systems and can predict a users future data needs and through collaboration with other *SPs* in the network, data can be scheduled to be piggy backed on other devices to support this device in a completely distributed manner.

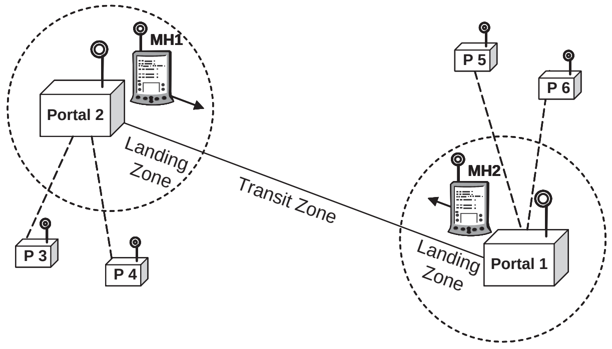


Figure 1. Network Model

Mobile Hosts (*MHs*) are wireless mobile devices that can communicate both with *SPs* (infrastructure mode) and neighboring *MHs* that are within range (ad-hoc mode). These *MHs* travel through the geographical area populated with *SPs*. Our network can be thought of as comprised of two distinct types of zones: *landing zones* and *transit zones*. A *landing zone* is essentially an island of connectivity around a *Service Portal* limited only by that *Portals* wireless range. An *MH* can communicate with an *SP* when it is in a *landing zone*. In a *transit zone*, an *MH* can communicate only with peer *MHs* that are within its immediate neighborhood. Furthermore, we assume that *MHs* move along predetermined routes (like highways or based on their personal information such as appointments, place of work and habits etc.). An *MH* can request services from *SPs* that it encounters as well as other *MHs*. We assume a heterogeneous mix of mobile devices in our network with differing capabilities. Devices also vary in their level of participation in our system and the amount of resources that they are willing to share. Services are actual applications that are hosted on *SPs* and available to the *MHs*. In this sense, our *SPs* can be considered as a combination of access point (for providing network support) and application server for hosting popular services. We envision the usage of semantic languages like RDF or DAML to efficiently describe a service so that it can be made available to *MHs* that may have a need for them (Newspaper service, stock quotes, local interests guide etc.).

4. Proposed Approach

We present our approach in terms of interactions that mobile devices and *Service Portals* have in our network.

Portal - MH Interaction: An *MH* that comes in range of a *Portal* communicates with the *Portal* to obtain data. Based on this interaction, *SPs* can track the service usage patterns for a user as well as the proposed route for the user. This information is used by the *Portal* to intelligently transfer only as much data as needed by the device until it reaches the next *Portal*. Also, *Portals* know about the impending data needs of other nodes that this *MH* is likely to meet en-route. By strategically downloading right amounts of data, any additional memory that this *MH* has can be utilized to route data for other nodes.

Portal - Portal Interaction: Once a *Portal* services a particular *MH*, it informs its neighboring *Portals* of this *MH* and its future data needs. Other *Portals* can then determine if this *MH* has enough info to last till it arrives at the next *Portal* or if proactive action is required by the *Portals* to transfer data to this *MH*. Through this interaction, *Portals* perform distributed scheduling to intelligently transfer data to even nodes that are not in the immediate range. Currently we are considering that a *Portal* interacts with its 1-hop neighbors to intelligently service *MHs*. This can be generalized to *n-hop* neighbors.

MH - MH Interaction: *MHs* traveling in a *transit zone* can form ad-hoc communities with neighboring peers. In the event of needing data, an *MH* can query its neighboring peers to see if they have the data that is needed. If a peer has the desired data, the peer notifies the *MH* and the data can be transferred. If no peer responds, the *MH* continues its querying process and as its neighborhood changes, other peers get involved in this collaboration. The process continues until a response is received or the *MH* arrives at a *landing zone* (where the *SP* can definitely provide the required data). It is through this interaction that a *MH* can obtain data that has been routed to it by a neighboring *Portal* that predicted this impending data need.

5. Numi Framework

Numi is our prototype framework for supporting infrastructure coordinated ad-hoc collaborative applications. *Numi* is essentially an agent runtime and a set of agents. By abstracting functionality into distinct agents, our framework is highly modular and loosely coupled. It is possible to pick and choose agents that a device needs to run thereby allowing different configurations of our framework (a lighter configuration is more suitable for devices like cell phones). All the services are also implemented as agents. Service providers can implement agents conforming to our specifications and these agents can be seamlessly introduced into a network. *Numi* differs from other peer-to-peer models such as JXTA and Gnutella as these are intended more for the wireline networks (suitable for interaction between *SPs* but not between a *SP* and a *MH*). Other approaches such as LEAP[3] are still missing support for ad-hoc collaboration (work in progress). *Numi* shares some similarities with PROEM[9]. However, unlike PROEM, *Numi* is intended for operation both on *SPs* (peer-to-peer through wireline) and *MHs* (ad-hoc peer-to-peer) whereas PROEM deals only with ad-hoc peer-to-peer interactions. In addition, *Numi* has extensive built in support for distributed scheduling which is essential for our proposed approach.

A *Numi* platform (running on a *SP* or a *MH*) is composed of a set of agents. A *Heartbeat Generator Agent* is responsible for periodically broadcasting *device presence messages* (every t seconds). A *Message Handler Agent* is responsible for handling the messaging needs of the framework. Agents on our platform use this component to send and receive messages to other agents on the same or on different platforms asynchronously. Messages are routed using a combination of agent identifier and platform identifier. A *Logger Agent* records every interaction that takes place on the local device. This includes user interaction with a specific service, messages that pass though the local *Message Handler*, peer encounters, peer queries for service etc. An *SP* uses these logs collected on an *MH* to extrapolate useful information such as service usage patterns, queries issued by other devices that this *MH* has encountered etc. Logging can be limited or even turned off on the devices with limited capabilities. A *Task Scheduler Agent* is responsible for scheduling prescribed tasks at various times. These tasks could be one-time tasks that need to be executed at a set time or repetitive tasks that occur at fixed durations. A *Data Handler Agent* is used for transferring data volumes between *MHs* and between an *MH* and an *SP*. The agent is required to implement a reliable protocol to exchange data volumes. *Portal Service Agents* run on top of our *Numi* platform on *SPs* and offer services to users. These include services such as a music jukebox, newspaper service, stock quote service etc. These *Service Agents* interact with the *Node Service Agent* running on a *MH* to provide a user with some useful service. A *Node Service Agent* also monitors service data usage and detects when the service is running out of data. When this occurs, the *Node Service Agent* publishes queries in its neighborhood to obtain the next set of data needed to keep that service running. *Service Agents* on peer devices that have this data acknowledge these queries and using the *Data Handler Agents*, data can be exchanged. In some cases, neighbors cannot handle these queries. However, since the *Logger* is logging these interactions, a neighboring peer device reaching an *SP* can trigger this *SP* to attempt to deliver the data to the *MH* that initiated the query. A *Service Manager Agent* is responsible for managing *Service Agents* on a platform. This *Manager* monitors system usage by each *Service Agent* including statistics like the amount of memory used, running time, messaging overhead incurred etc.

6. Numi Component Interactions

Mobile Host to Service Portal Interaction: An *MH*, in a *landing zone*, can request form *SP* a set of new services (upon user's request) or it can ask for additional data for currently running services (transparent to the user). In the case of a request for a new service, the *SP Service Agents* formulate the initial set of data needed by a user and notify the *MH Service Manager* to transfer this initial set. Several factors are used in formulating these data sets such as the *MH's* route (which is embedded in the node's heartbeat), storage capacity, service characteristics, neighborhood information (to take advantage of groups of users traveling in the same direction) etc. The *SP Service Agents* also initiate a *SP* to *SP* interaction to facilitate the *MH* data management needs at subsequent *SPs* that are on this devices path. A *MH Service Manager*, upon receiving notification messages from *SP Service Agents*, uses its *Data Handler Agent* to fetch the data needed for each selected service and activates corresponding *MH Service Agents*.

In the case of a request for additional data for currently active services, once the zone transition take place (from a *transit zone* to a *landing zone*) the *MH's Location Monitor* alerts the local *Service Manager* and all active *MH Service Agents* about it. Each *Service Agent* issues a *service continuation message* to the corresponding *SP Service Agents*. The *SP Service Agents* compiles the data volume and through the *Data Handler Agent*, transfers the volume to the *MH*. As in the case of requesting new service, several factors are used in formulating the data sets.

Portal to Portal Interaction: *Portals* use this mechanism to ensure that data that would be needed by *MHs* are properly scheduled to be available at *SPs* along an *MH's* route. The *Portals* select *MHs* with spare capacity that are moving towards the needy peer device and use the *MHs* to route the data. This key feature enables *Portals* to actively participate in service data routing instead of passively waiting for *Mobile Hosts*.

When an *SP Service Agent* (origin *SP*) offers some service to an *MH* (target *MH*) in its *landing zone* it could be an initial set of data for a new service or can be continuation data that are provided to a running service. The origin *SP's* location monitor determines the route for the target *MH* (the location monitor caches routes that are published within the *MH's heartbeat message*). Using this route, the origin *SP Service Agent* contacts its counterpart on the destination *SP* to notify what services have been provided to this *MH*. This is achieved through a *service notification message* that contains information like service name, last data unit provided, the devices route and its capabilities. The destination *SP Service Agent* then determines the time it will take for the target *MH* to reach this *Portal*. This can be obtained from static configuration information like network maps or can be dynamically learnt by each *SP* by tracking devices passing through (more adaptive). The destination *SP Service Agent* can determine if the *MH* has enough data to make it all the way to this *SP* or if that *MH* will require additional data somewhere in the neighboring *transit zone*. In the latter case, the destination *SP Service Agent* determines the optimal time to schedule this data to be carried towards the target *MH*. We currently use a simple heuristic, assuming α is the normal travel time from origin *SP* to destination *SP*, β is the time it will take for user of the target *MH* to consume the data that is currently available on the *MH*, then we define $\lambda = 2*\beta - \alpha$. The destination *SP* then waits for a *min*(α,λ) before starting to look for a carrier *MH* to deliver needed data to the target *MH*. The reason for this delay is so that the carrier *MH* ideally meets the target *MH* just as that device is about to use up its current data. Without this, the carrier *MH* may meet the target *MH* well in advance of when the target *MH* actually needs that data. This is not desirable as the target *MH* now needs to make precious room to store this future data. We assume that the network is sufficiently populated with *MHs* such that a *Portal* can find a carrier *MH* at the optimal time. Other alternative designs that do not make this assumption are possible. For example, an *SP* could start sending additional data through carrier *MHs* as soon as they become available with no delays. The target *MH* would then be responsible for determining the best time to refresh its data volumes.

In our design, an *SP Service Agent* uses the task scheduler to plan delivery for the target *MH*. When task activates it contacts the *SP's* location monitor to determine if the target *MH* has arrived. If that *MH* already has passed though or is currently in the *landing zone* then the task terminates. Otherwise, the location monitor replies with a list of *MHs* that are heading into the *transit zone* towards the target *MH* (using the device routes learned). If at that moment there are no such *MHs* then, the task is rescheduled for later execution. However, if there is one or more *MHs* that are heading into the *transit zone*, the data is given to just one. The data requests are handled in the order in which they are scheduled. Adding a priority queue allows *Portals* to offer differentiated levels of service.

Mobile Host to Mobile Host Interaction: This interaction take place in *transit zones* when an *MH* tries to obtain additional data from another *MH*. Once *MH Service Agent* detects that it is running out of data it contacts the location monitor to see if there are any neighboring peers. The *Service Agent* uses the *Message Handler* to publish queries for the data units that it needs. If the passing *MH* contains the needed service data (the *SP* to *SP* interaction attempts to make these queries succeed most of the time by predicting devices needs and equipping carriers accordingly), the *Data Handler Agents* on these devices interact to download the data into the requesting *MH*. If the passing *MH* does not have the needed data, a *"No Such Data" message* is sent to the requesting *MH*. The requesting *MH* continues the search for the next data set. Detailed description of thises tree interactions could be found in [13].

7. Experimental Results

We built a simulation model of our approach to validate its viability. We used Glomosim[2] as our modeling tool. *MHs* were assumed to move randomly between *SPs* that were uniformly placed throughout a geographic region of ten square kilometers. Nodes had access to a finite set of service data. 802.11 was used as MAC protocol. We compared our approach against a conventional data hoarding scheme (*MH* downloads as much of data as can/needed till next *Portal*, *MHs* cannot communicate with each other) and a conventional ad-hoc querying scheme (*MHs* in a *transit zone* can communicate with peers to request data as well as download data from *SPs* that they visit). Our work mainly focused on modeling percentage of simulation time a nodes spends without of data. We consider this to be a measure of expected service disruption in a network.

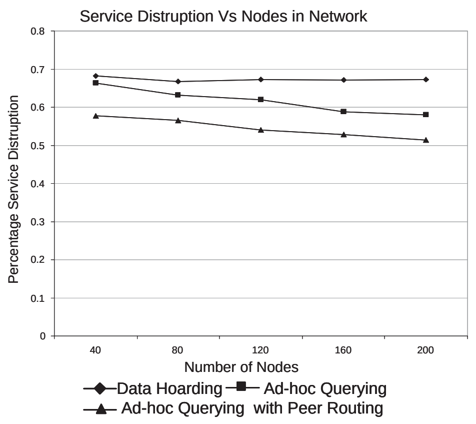


Figure 2. Simulation Results

We compared the three schemes considering small devices (<1MB capacity) traveling randomly through a network (speed is 20mt/sec, data packet size <0.2MB, 5 *Portals*). The conventional data hoarding scheme performs the worst of the three. Increase in the number of nodes in the network does not seem to have any significant impacts in such a scheme. Increasing the number of *Portals* (keeping number of nodes as 100) benefits this scheme, as nodes are more likely to be in *SPs landing zone*. Ad-hoc querying scheme performs better in networks with limited *Portals*. As number of *Portals* increases, the ad-hoc component of the interactions between nodes reduces as more nodes can rely on a *SP* to answer their requests instead of resorting to ad-hoc peer queries. Increasing number of nodes in the network has a positive effect as there are more peers that a *MH* can query and the likelihood of finding a response is higher. Our scheme consistently out-performs the other two. Increasing nodes allows the *Portals* to more easily find carrier nodes while increasing *Portals* allows for more efficient scheduling within the network.

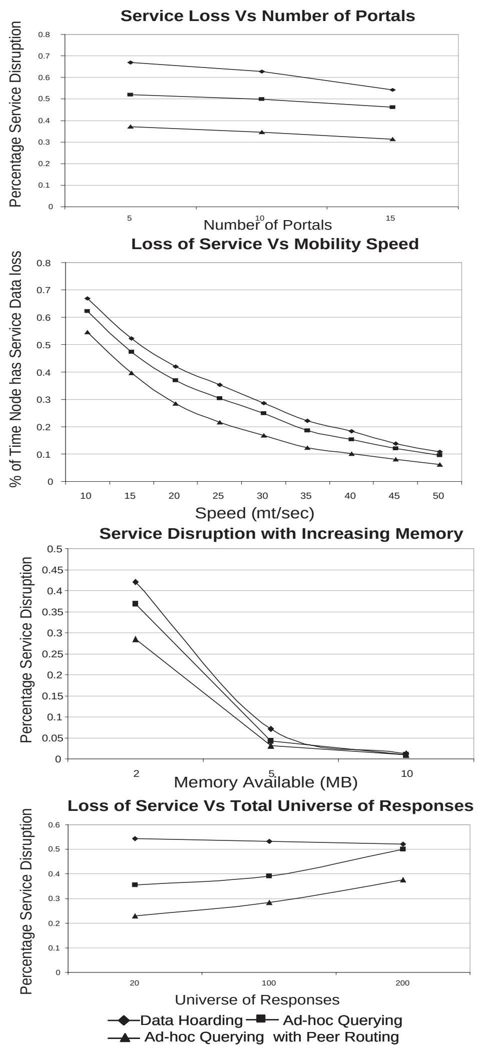


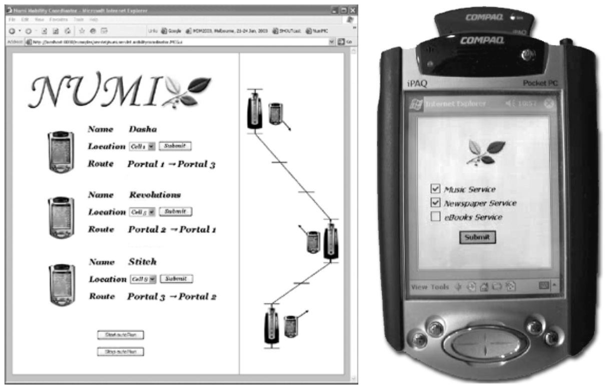
Figure 3. Simulation Results

We also conducted tests by changing the speed of nodes in the network (100 nodes, data packet size $<$0.2 MB). We found that as the speed increases, the level of service disruption appears to decrease for all schemes. For data hoarding schemes, this is due to the reduction in the time that a node spends in a *transit zone*. For ad-hoc querying schemes, this is due to the increased number of peers that a given *MH* can query. Again, our scheme outperforms the other two approaches. Increasing the node speeds allows carriers selected by *Portals* to reach a node in a need quicker thereby reducing service disruption time. We conducted tests by varying the memory capacities of mobile devices (node speed 20mt/sec). As expected, network comprised of devices with higher memory capacity suffered much less service disruption than the limited capacity network. Again, our approach results in less disruption in service than both the data hoarding and ad-hoc query schemes.

We varied the total universe of request/response pairs in our model (100 nodes, 5 *Portals*, 20 mt/sec, data packet size $<$0.2 MB). Conventional ad-hoc querying relies completely on chance that a *MHs* queries are heard by a passing peer that happens to have the desired data. We found that by increasing the universe of data in the network, there was less likelihood that a passing peers had the desired data. The ad-hoc schemes still performs better than simple data hoarding but as the universe of data grows, the difference between schemes becomes less significant. Our scheme performs better than the other two (the growth is slower then simple ad-hoc querying) as the *Portals* try to ensure that a peers passing by an *MH* in need, does in fact carry data that this *MH* would be needing.

8. Prototype Application Implementation

We have implemented a prototype of our framework using Java programming language. Our platform runs on three PCs and three iPAQs equipped with 802.11b wireless LAN cards. The PCs run the *SP* platform and a Tomcat Apache Servlet Engine. The iPAQs run the *MH* platform. The iPAQs run the Jeode EVM. To simulate the mobility of the devices (moving in range and out of range of each other) we divided each *transit zone* into non-overlapping cells. Each cell has a unique cell ID. *MHs* are able to communicate with each other only if they are in the same cell. Messages have been augmented to carry a cell ID. Since we are using 802.11, broadcast messages will be heard by all devices. However, the *Message Handler* filters out all messages that do not match a device's current cell ID. By using this notion of cells, we can simulate neighborhoods and by changing a *MH's* cell ID, its neighborhood can be changed thereby simulating movement. We have developed an additional simulation component called the Mobility Coordinator (right figure below). Using this, control messages can be sent to any *MH* to change its current cell ID. More detailed description of prototype can be found in [13].



Name Dasha
Location Cell 1 Submit
Route Portal 1 → Portal 3

Name Revolutions
Location Cell 5 Submit
Route Portal 2 → Portal 1

Name Stitch
Location Cell 8 Submit
Route Portal 3 → Portal 2

[✓] Music Service
[✓] Newspaper Service
[ ] eBooks Service

Figure 4. Prototype Application on iPAQ

9. Conclusion and Future Work

In this paper, we have presented a novel approach to route data needed by a mobile device through peers in an infrastructure network. Our model augments hoarding schemes with the ability of users to share load among themselves so that collectively they can satisfy individual user data needs. Through this, we attempt to minimize the amount of interaction a mobile device needs with an expensive WAN network. Also, by sharing load, sophisticated applications can be offered even on less capable devices as long as their neighborhood is sufficiently resource rich to satisfy the applications needs. The infrastructure in our model facilitates this collaboration by equipping devices that are likely to meet with data that the others may require. Unlike existing schemes that do not gracefully handle deviations in a devices expected route, we provide a mechanism for our infrastructure to detect such deviations and react by actively trying to route needed data to such needy devices. Through our framework, data needs for mobile devices can be managed across a network of *Portals* in a highly distributed manner that scales and is cost efficient with little dependency on a cellular WAN. Our simulation results also show that our scheme consistently outperforms both traditional pure infrastructure based and a traditional pure ad-hoc collaboration based data management models.

Our current work is focused on extending our simulation models to study the benefits of n-hop scheduling in a network of *Portals*. We are also working on a cost model to determine an optimal *Portal* placement policy within a given network. *Numi* framework is also being augmented with a security component to handle privacy issues.

References

[1] WICAT, Infostation Project, Polytechnic University, WWW, `http://wicat.poly.edu/infostation.htm`.

[2] Global Mobile Information Systems Simulation Library, GLOMOSIM, WWW, `http://pcl.cs.ucla.edu/projects/glomosim/`.

[3] F. Bergenti and A. Poggi. Leap: A fipa platform for handheld and mobile devices. In *presented at ATAL*, 2001.

[4] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

[5] S. Fickas, G. Kortuem, J. Schneider, Z. Segall, and J. Suruda. When cyborgs meet: Building communities of cooperating wearable agents, October 1999.

[6] G. Wu and C. W. Chu and K. Wine, J. Evans and R. Frenkiel. Winmac: A novel transmission protocol for infostations., 1999.

[7] A. Iacono and C. Rose. Bounds on file delivery delay in an infostations system. In *In Proceedings of the IEEE Vehicular Technology Conference,*, 2000.

[8] Y.-B. Ko and N. H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 66–75, 1998.

[9] G. Kortuem, J. Schneider, D. Preuitt, T. G. Thompson, S. Fickas, and Z. Segall. When peer-to-peer comes face-to-face: Collaborative peer-to-peer computing in mobile ad hoc networks. In *Proceedings 2001 International Conference on Peer-to-Peer Computing*, Aug, 2001.

[10] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proc. IEEE INFOCOM.*, april 1997.

[11] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing(dsdv) for monbile computers. *Comp. Commun. Rev.*, pages 234–44, October 1994.

[12] C. Perkins and E. Royer. Ad-hoc on-demand distance vector routing. In *Proc. 2nd IEEE Workshop. Mobile Computing Systems and Applications*, pages 90–100, February 1999.

[13] O. Ratsimor, S. B. Kodeswaran, A. Joshi, T. Finin, and Y. Yesha. Combining infrastructure and ad-hoc collaboration for data management in mobile wireless networks. In *Workshop on Ad-hoc Communications and Collaboration in Ubiquitous Computing Environments*, Nov 2002.

[14] R.H. Frenkiel, B.R. Badrinath, J. Borras, and R. Yates,. The infostations challenge: Balancing cost and ubiquity in delivering wireless data. *IEEE Personal Communications*, 7(2):pp.66–71, April 2000.

[15] U. Kubach and K. Rothermel. A map-based hoarding mechanism for location-dependent information. In *In Proceedings of the Second International Conference on Mobile Data Management (MDM 2001)*, January 2001.

[16] T. Ye, H.-A. Jacobsen, and R. H. Katz. Mobile awareness in a wide area wireless network of info-stations. In *Mobile Computing and Networking*, pages 109–120, 1998.