Combining Infrastructure and Ad hoc Collaboration For Data Management in Mobile Wireless Networks

Olga Ratsimor, Sethuram Balaji Kodeswaran,
Anupam Joshi, Tim Finin, Yelena Yesha
Department of Computer Science and Electrical Engineering,
University of Maryland Baltimore County
1000 Hilltop Circle, Baltimore,MD 21250
+1 410 4553971
oratsi2, kodeswar, 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. However, such infrastructure in terms of base stations is available in many popular areas already. These base stations often offer high-speed data connectivity to a wired network. In this paper, we describe an approach where a combination of infrastructure and ad hoc communication can be used to better manage the data needs of a mobile wireless device. In our approach, base stations track user mobility and determine data usage patterns of users passing by. Based on this, future data needs of a mobile device can be predicted. Base stations can 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 data. When such a carrier meets the needy device, they can form ad hoc peer-to-peer communities to transfer the carried data. We also present the motivation, architecture and design of NUMI, our framework for supporting combined infrastructure and ad hoc peer-to-peer computing and a prototype application built on top of this framework.

Keywords
Ad-hoc networks, collaboration, proactive data delivery, infostations, mobile data management, peer-to-peer, community networks.

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[1]. Also, there has been a lot of advances in the field of short-range wireless communication technologies facilitating personal area networks (PANs) like Bluetooth[8], homeRF[7], IEEE 802.15[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.

Although ad hoc networks do not require any infrastructure support, infrastructure in the form of base stations is already available in many popular areas. In addition, there is growing popularity in terms of community wireless networks [13]. Such community networks aim to offer network connectivity to mobile devices in metropolitan areas. Often such networks are characterized by pockets of network connectivity (close to the access point) surrounded by regions of no network access. Several projects are underway to commercialize this notion of creating islands of high-speed network connectivity that are spread throughout a metropolitan region [14].

We are seeing numerous applications available on desktop PCs making their way into mobile devices. A great majority of currently available mobile devices have restrictions that hinder a smooth migration such as limitations on power consumption, smaller display, processing power etc. 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[2]. Data management to support applications on these devices is a challenging task. With the increased popularity of multimedia, financial applications etc., the amount of data required by these services is also on the rise. Mechanisms must be designed to efficiently ration out the available resources on a device. Also, more efficient approaches are required to maximize the utilization of device capabilities and the communication infrastructure that these devices rely on. Management of a user's data requires intelligent data transfer capabilities to and from the users device. 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. 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. Once they move out of range, the mobile devices resort to ad hoc peer-to-peer collaboration with other mobile devices to satisfy their data needs. The distinguishing feature here is that our access points have been enhanced to improve the "quality" (in terms of a mobile devices requests being satisfied by its peers) of the ad hoc collaborations of devices that are out of range. By analyzing mobility patterns of users, our access points can determine the future data needs of these users. In addition, our access points can also predict the time and location that the user will be when such a need arises. Using this information, access points can 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. The claim here is 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 devices will be able to satisfy the data needs. In essence, our approach proposes using access points to coordinate and facilitate ad-hoc collaborations between mobile devices such that the ad hoc communities that are dynamically formed have a higher probability of satisfying the requests of their participating peers instead of leaving this completely to chance.

To validate this approach, we have designed and built a prototype system, NUMI, which can be deployed on both mobile devices and access points. NUMI provides all necessary framework functionality on top of which applications can be built. NUMI allows the data needs for such applications to be satisfied through a combination of infrastructure and ad hoc collaboration. We have built a intelligent music service on top of NUMI which allows a mobile user to play his favorite MP3 playlists on his PDA. Instead of downloading all songs on his playlist into the PDA (limited memory), only a needed subset are downloaded (the ones that are next on the playlist). Songs that have been heard are transparently swapped out with subsequent songs on the playlist through interactions with access points that the user encounters and other mobile devices that the user's PDA happens to collaborate with on its route.

In the following sections we will discuss related work in this area, describe our network model, present our NUMI framework and its component interactions, and finally, a prototype application built on top of it.

RELATED WORK

Cellular voice and data networks facilitate ``any time, anywhere" connectivity but they are expensive and offer low bandwidth. Infostation networks[9] have often been suggested as a viable alternative to meet the needs of mobile applications. An infostation network would consist of a set of towers offering short-range high bandwidth radio coverage. They offer high-speed discontinuous 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 base station coupled with an information server such that the base station provides the network connectivity while the information 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[3].

Several data management models have been suggested for these types of infrastructure based systems. In [4], 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 for that matter, access points distributed throughout the network) to facilitate data hoarding on mobile devices[10]. In this model, by 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[11], 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. [12] 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 such that the information needed to be stored even until the next access point may be too large for the device to handle. Existing schemes treat infostations purely as oasis of information. Devices check-in with the infostation upon arrival and obtain the relevant data. Such a model is intolerant to changes in a users expected travel plans. For example, in a hoarding scheme, if a user is held-up on the way to the next infostation, it is most likely that hoard misses will occur (as enough data would not have been cached). There are no facilities for the existing infrastructure to predict and take corrective action on behalf of this user simply because this user is out of range. Mobile devices in such schemes have no option but to use the costly WAN or wait until they arrive at the next Infostation to receive the needed data. 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 in nature. A characteristic of such networks is the dynamic creation of communities between peer devices which enables these peers to interact and exchange data with no dependence on any preexisting infrastructure. These impromptu collaborations are the key to data management in such 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 these ad hoc collaborations to trade tasks[19]. In systems such as [26], 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". Although this offers a mechanism for a mobile device to satisfy its data needs, it comes at a cost. It is very possible that communities that this needy device participates in do not satisfy the needs of this device purely because its peers did not consider the needed information as relevant to them and never carried it even though they may have had enough storage capacity to carry this information. Most models assume that devices carry only information needed by the user and are focused purely on the owner. There is no "community network" type of notion prevalent in ad hoc networks where users with spare capacity and willing to carry data for other users with limited capacity. In addition, most MANET routing protocols assume that a user's mobility pattern is purely random[20,21,22,23,24]. 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 components (access points) to monitor a user's mobility patterns (even if a users personal information is not presented, a device could specify an expected route to the access point), the infrastructure components can predict where and when a user may require some data. This information can be used to identify other mobile devices that are likely to meet this device at that time. Such 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 this data in its neighborhood and the carrier would "conveniently" show up carrying that needed data. We are proposing a "community ad hoc network" notion on mobile devices, i.e. excess capacity on a mobile device can be used to carry data needed for others. 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 carrying 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).

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.

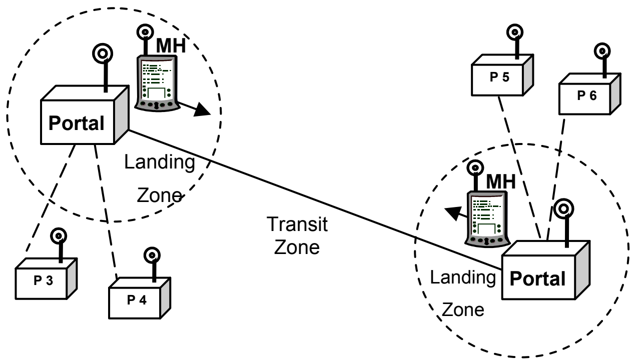

Figure1: Network Model

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[6] or DAML[5] 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.).

Typical Application of our Approach

Bob is walking by a Starbucks and decides he would like to listen to some music. He turns on his PDA which immediately connects to the nearby Service Portal (SP) at Starbucks and downloads a suitable playlist. The Starbucks SP determines from Bobs PDA (which infers this through his appointment book) that Bob is walking to his work, which is 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 now 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 would have detected that it is running out of songs and would be querying its peers for the needed songs. In this instance, Bob's PDA would conveniently find 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 mainly so that the chosen carrier runs into Bob just about the time when Bob's PDA is going to start querying its neighbors)

NUMI Framework

To study the viability of our approach, we have designed and built NUMI, a framework for supporting infrastructure coordinated ad hoc collaborative applications. The NUMI framework can be run both on a mobile device and a

Service portal. Services can be implemented to run on top of our platform and take full advantage of the available ad hoc and infrastructure mode communication support. We designed NUMI with the goal of reusability. NUMI is essentially a set of agents and an agent runtime that could be used on both MH and SP. 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 may be more suitable for a cell phone while a laptop could support a much heavier configuration). Services on top of NUMI are also implemented as agents. Service providers can implement agents conforming to our specifications and these agents can be seamlessly introduced into a network. Service agents offer services that a user would need (a music service for example) while framework agents handle the lower level tasks that are needed for our framework.

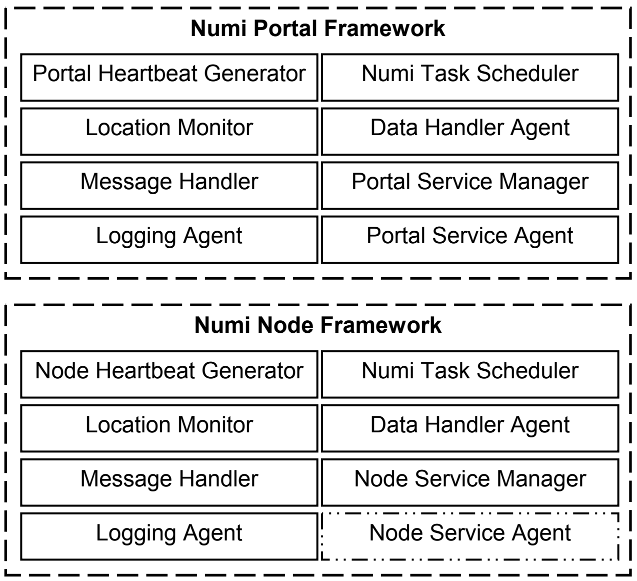

Figure2: Numi Portal and Mobile Host Platforms

The above figures illustrate typical configurations for NUMI on SPs and MHs. A heartbeat generator agent is responsible for broadcasting device presence messages. These heartbeats are broadcast periodically (every *t* seconds) and are used for identifying other devices that are in range. The SP heartbeat includes a unique platform identifier and a URL pointing to the list of services that are currently available to the user at that SP. The MH heartbeat contains a unique platform identifier, type of device and the route that this MH is traversing.

A location monitor agent is responsible for location dependent issues. On an MH, this agent identifies whether or not that device is currently in a *landing zone* or a *transit zone* (based on whether or not a portal presence message has been received in the last *t* seconds). Also, this agent tracks the other peer devices that are currently in the neighborhood (based on the peers' device presence message).

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. Each message in our system contains a source, a destination, a message type and a message content. Our message handler also allows agents to register for specific types of messages. In this case, whenever the message handler receives a message that is not addressed to a distinct agent on that platform, that message can now be routed to agents that have registered for that message type (to handle advertisements for example).

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. On MHs with limited capacity, this agent may minimize logging or not log at all.

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. Currently, our framework uses the FTP protocol.

Portal Service agents run on top of our Numi platform on SPs and offer services to a user. These include services such as a music jukebox, newspaper service, stock quote service etc. These agents conform to the Numi agent specification and have access to different features offered by the platform. Service agents at an SP actively wait for a user's request for a service. When a user requests a new service, the service agents determine the amount the data that user needs and capacity available on the device. Using this, the SP downloads an initial set of data to the device and informs the next SP on the devices route. This can be generalized to notify the next *n* SPs on a devices route. (Current implementation supports only one hop scheduling). The service agents at the subsequent portals determine when the device is likely to need data and take proactive measures to get the data to this device through other carriers. It is also possible that a MH was given enough data to last until the next SP but was moving slower than usual. In case a device does not show up at a *landing zone* at it's prescribed time, the service agents at that SP then actively starts looking for other devices whose routes indicate that they are heading in the direction of that

MH and attempt to use them to deliver the next data segments. These service agents continue to track the MH until it arrives and a new set of data. Once this is done, the SP service agent notifies the next SP(s) on the route to schedule future data for this MH.

A Node service agent runs within NUMI on an MH offering a service to the user. Whenever a device enters a *landing zone*, the node service agents on that MH receive data updates from their respective SP service agents. An MH 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 other devices that have this data acknowledge these queries and using the data handler agents, data can be exchanged. These impromptu ad hoc collaborations have a high probability of succeeding, as a needy device's neighbors have probably been pre-equipped with this needed data by an adjacent SP. In some cases, neighbors cannot handle these queries. However, since the logger is logging these interactions, a neighboring 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. The SP service manager hosts the service agents representing services available at that SP. This manager monitors system usage by each service agent including statistics like the amount of memory used, running time, messaging overhead incurred etc.

NUMI has several succinct features that differentiate it from other Peer-to-Peer frameworks. NUMI is very different from JXTA[16] and Gnutella[17] 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[25] are still missing support for ad hoc collaboration (work in progress). NUMI is similar to PROEM[18] but with some differences. 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. In addition, NUMI has extensive support for distributed scheduling built in which is essential for our scheme.

NUMI COMPONENT INTERACTIONS

Mobile Host to Service Portal Interaction

An MH, in a *landing zone*, can request a set of new services (upon user's request) or it can ask for additional data for currently running services (transparent to the user).

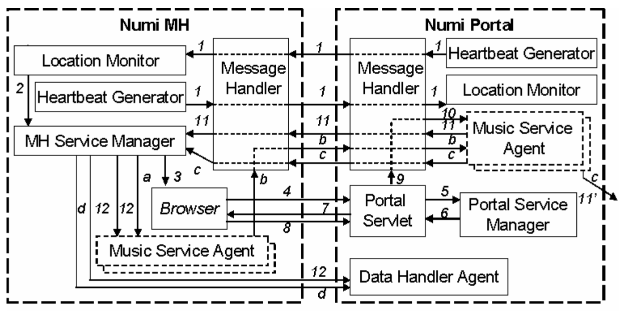

Figure3: Mobile Host to Service Portal Interaction

In the case of a request for a new service, the interaction begins when the location monitors of both the MH and the SP detect each other's heartbeats (Step 1). The SP stores the MH's route in it's location monitor to support the SP to SP interaction described in section. The MH service manager presents the URL from the SP's heartbeat to the user through a web browser (Steps 2 and 3). When the user follows this location dependent bookmark, the MH receives a dynamically generated list of services available at the local SP (Steps 4 to 7). The user can select a set of services from this list. User selections are handled by a servlet on the SP which forwards this to the corresponding SP service agents (Steps 8 to 10) through *ServiceRequestMessages*. Each service agent creates a set of data and notifies the MH Service Manager (Step 11). Several factors are used in formulating these data sets such as the MH's route, 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 (Step 11') to facilitate he 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 (Step 12) and activates corresponding MH service agents.

In the case of request for additional data for currently active services, once the location monitor on an MH detects an in-range SP, it notifies the local MH service manager. This zone transition (from a *transit zone* to a *landing zone*) (Step 1) is communicated to all active MH service agents (Step a).Each service agent issues a *ServiceContinuationMessage* to the corresponding SP service agents (Step b). This message contains the service name, last data unit used by the service and other service specific information like service execution plan (play list in case of a music service), usage profiles etc. The SP service agents use this information and other factors like device route, capacity, service characteristics, neighborhood information etc., to create the next set of data needed for this service (Step c). Using the data handler agent, these data volumes are transferred to the MH (Step d).

Portal to Portal Interaction

This interaction is used between SPs to efficiently handle a mobile user's movement through a network. 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. However, there may be cases where a device in a *transit zone* does not have sufficient data to support a service till this device reaches the next SP on it's route. In our framework, the SPs handle this by selecting other MHs with spare capacity that are moving towards this device and using them to route the needed data. This key feature enables SPs to actively participate in service data routing instead of passively waiting for MHs. This is essential to support devices with limited capabilities and deviations in a device's expected route.

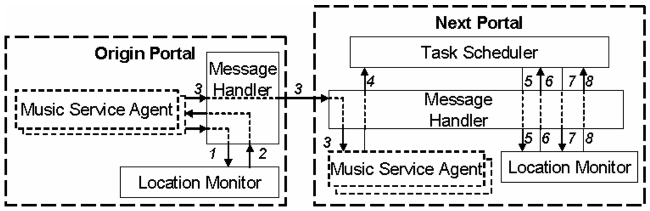

Figure4: Portal to Portal Interactions

This interaction is initiated when an SP service agent (origin SP) offers some service to an MH (target MH) in its *landing zone*. This could be an initial set of data for a new service that the MH has requested or can be continuation data that are provided to a running service. The origin SP service agent then contacts the local location monitor agent to determine the route for the target MH (Steps 1 and 2). The location monitor is able to provide this information as it caches routes that are published within the heartbeat messages periodically broadcast by the MH. Using this route, the origin SP service agent then contacts its counterpart on the next SP (destination SP) to notify what services have been provided to this MH (Step 3). This is achieved through a *ServiceNotificationMessage* 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 $min(\alpha,\lambda)$ 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. Currently, 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 local task scheduler to schedule delivery for the target MH (Step 4). At the specified time, the corresponding task is activated. This task contacts the local location monitor to determine if the target MH has arrived (Step 5). 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) (Step 6). If at that moment there are no such MHs then, the task is rescheduled with the task scheduler for later execution (Steps 7 and 8). The task will be activated in *t* seconds and process will be repeated. However, if there is one or more MHs that are heading into the *transit zone*, the data is given to one of these MHs with spare capacity. 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

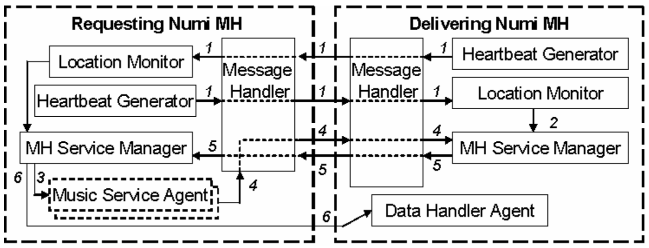

Figure5: Mobile Host to Mobile Host Interactions

The Node-to-Node interaction is employed by an MH to obtain any additional data from another MH. Such interactions occur only in *transit zones*. This interaction is initiated when an MH service agent detects that it is running out of data. This agent contacts the location monitor to see if there are any neighboring peers (Steps 1 to 3). The service agent uses the message handler to publish queries for the data units that it needs (Step 4). 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 MH needs and equipping carriers accordingly), the data handler agents on these devices interact to download the data into the requesting MH (Steps 5 and 6). 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 set of data by initiating Node-to-Node interaction with other MHs in its range.

PROTOTYPE APPLICATION IMPLEMENTATION

On top of NUMI, we have built a working music application that allows a user to listen to his favorite MP3 playlist on his PDA. On startup, the application downloads only a few songs on the playlist. As the user listens to his songs, previously heard songs are removed from the device and replaced with his other favorites.

We have implemented a prototype of our framework using Java programming language. We installed our platform on three PCs and three iPAQs. The PCs run the SP platform and the iPAQs run the MH platform. All devices used were equipped with 802.11b wireless LAN cards. The iPAQs were running the Jeode Embedded Virtual Machine. Each SP also runs a Tomcat Apache servlet engine.

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. Using this, control messages can be sent to any MH to change its current cell ID.

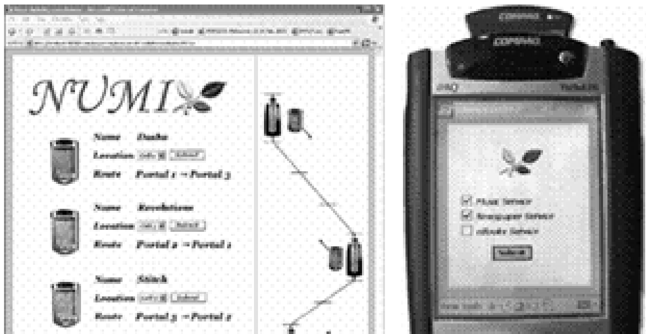

Figure6: Mobility Coordinator and Mobile Host Interface

Prototype Setup

We set up network of 3 SP: P1, P2 and P3. *Transit zones* between SPs were divided into cells as shown in figure 7. There are three mobile users Susan, Bob and Jim with iPAQs. Bob has MH1; he starts his trip at P1 and plans to go to P3 through P2. Susan has MH2; she starts her trip at P2 and plans to go to P1. Jim has MH3; he starts his trip at P3 and plans to go to P2. There are three services available on the network: Music Service, Newspaper Service and eBooks Service. Bob is using Music Service and eBooks Service; Susan is using Newspaper Service; Jim is using eBooks Service.

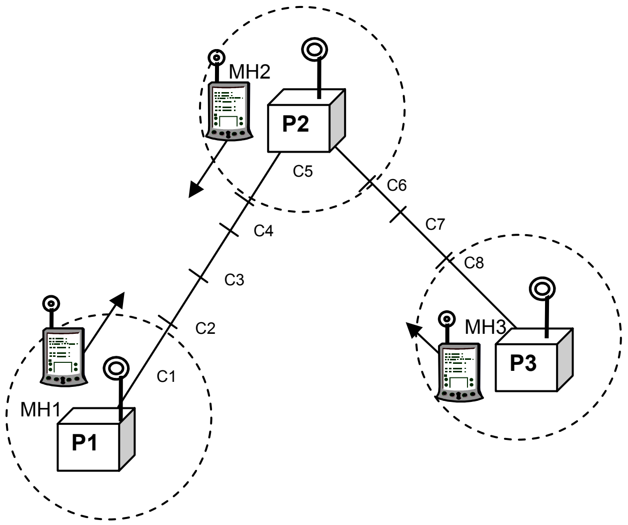

Figure7: Prototype Network

"Memory Shortage" Scenario

This scenario occurs when the user requests one or more services but his or her MH is unable to store all the needed data to last all the way form one *landing zone* to the next. Suppose Bob requested the Music Service and the Newspaper Service through a MH to Portal interaction. MH1 gets initial sets of data and Bob starts on his trip from P1 to P2. MH1 has enough data to last Bob 80% of his trip. He will run out of data in C3. P1 notifies P2 about service provided to Bob through portal-to-portal interaction. P2 determines that he does not have enough data to last him until he reaches P2. Meanwhile, Susan with MH2 is getting ready to depart from the *landing zone* of P2 (towards P1). She is using the Newspaper Service and has capacity to spare. P2 gives MH2 the data for Susan's Newspaper Service and it gives the next set of data needed for Bob's Newspaper and Music Services. Bob passes by Susan in C3. MH1 contacts MH2 and obtains the next set of data for the Newspaper Service and Music Service. Bob now can reach P2 with out any service disruptions.

"Delay" Scenario

This scenario deals with cases when a user takes longer then expected to pass thought the *transit zone*. The user is initially given enough data to last from one *landing zone* to the next *landing zone*. However for some unexpected reason the user takes longer than initially was expected. Suppose Bob continues on his way from P2 to P3 though C5, C6, C7 and C8. He is continuing to use Music Service and Newspaper Service. MH1 has enough memory to store data for his services to last from P2 to P3. He departs P2's *landing zone* towards P3 with no plans for detours or delays. Along the way in C7 he decides to take a rest break for a hotdog. As he stands in line to place his order, he continues to read his newspaper and listen to his music. Meanwhile P3 determines that Bob did not arrive on time to P3's *landing zone*. P3 receives a request for a Newspaper Service from Jim who is planning to depart P3's *landing zone* and head towards P2. His MH3 gets data for Jim's Newspaper Service and also receives music and newspaper data for Bob. By this time, service agents on Bob's iPAQ would have realized that they are running out of data and will start querying the neighborhood. As Jim passes by resting Bob, MH1 and MH3 discover each other and MH1 uploads needed data from MH3. Bob finishes his hotdog and resumes his trip. He reaches P3 without any service interruptions.

CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel approach to manage data needed by a mobile device in an infostation 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 infostations in our model facilitate this collaboration by equipping devices that are likely to meet with data that the other may require. This allows our infostations to offer services even to devices that are not in range. Unlike existing schemes that do not gracefully handle deviations in a devices expected route, we provide a mechanism for our infostations to detect such deviations and react by actively trying to route needed data to these 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 current work is focused on building a simulation model for our approach to study various characteristics of our scheme such as different scheduling mechanisms, scalability, performance and build cost models. We are also working on extending our existing NUMI implementation to support multi-hop scheduling and are adding support for a security framework.

ACKNOWLEDGMENTS

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

REFERENCES

1. IEEE Std 802.11-1997, Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Publications, 1997.
2. Guanling Chen and David Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.
3. G. Wu, C. W. Chu, K. Wine, J. Evans, and R. Frenkiel. Winmac: A novel transmission protocol for infostations., 1999.
4. Iacono and C. Rose. Bounds on file delivery delay in an infostations system, 2000.
5. DARPA Agent Markup Language. World Wide Web, http://www.daml.org.
6. O. Lassila and R. Swick. Resource Description Framework. http://www.w3.org/TR/1999/REC/rdf-syntax-19990222, 1999.
7. P. Negus, K. J. Stephens and L. Landsford. HomeRF: Wireless Networking for the connected home. IEEE Personal Communications, 7:20{27, Feb 2000.
8. Bluetooth White Paper, World Wide Web, http://www.bluetooth.com/developer/whitepaper.
9. 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.
10. U. Kubach and K. Rothermel. A map-based hoarding mechanism for location-dependent information.In Proceedings of the Second International Conference on Mobile Data Management (MDM 2001), January 2001.
11. Polytechnic University WICAT, Infostation Project. WWW, http://wicat.poly. edu/infostation.htm.
12. Tao Ye, Hans-Arno Jacobsen, and Randy H. Katz. Mobile awareness in a wide area wireless network of info-stations. In Mobile Computing and Networking, pages 109-120, 1998.
13. WirelessCommunities - The Personal Telco Project, World Wide Web http://www.personaltelco.net/index.cgi/WirelessCommunities
14. T-Mobile HotSpot, World Wide Web http://www.t-mobile.com/hotspot/

15. IEEE 802.15 Working Group for Wireless Personal Area Networks (WPANs), World Wide Web http://grouper.ieee.org/groups/802/15/
16. JXTA Project,World Wide Web http://www.jxta.org
17. Gnutella Project, World Wide Web http://www.gnutella.com
18. Gerd Kortuem, Jay Schneider. An Application Platform for Mobile Ad-hoc Networks. 2001 Ubiquitous Computing Conference, Workshop on Application Models and Programming Tools for Ubiquitous Computing
19. Gerd Kortuem, Jay Schneider, Jim Suruda, Steve Fickas, Zary Segall When Cyborgs Meet: Building Communities of Cooperating Wearable Agents Proceedings Third International Symposium on Wearable Computers
20. Charles E. Perkins and Elizabeth M. Royer. "Ad hoc On-Demand Distance Vector Routing." Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100.
21. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers (1994) Charles Perkins, ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications
22. Vincent D. Park and M. Scott Corson, A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks, Proceedings of IEEE INFOCOM '97, Kobe, Japan
23. The Zone Routing Protocol (ZRP) for Ad Hoc Networks <draft-ietf-manet-zone-zrp-o4.txt>, Zygmunt J. Haas, Marc R. Pearlman and Prince Samar
24. Y.Ko, N.H.Vaidya, “Location Aided Routing (LAR) mobile ad hoc networks”, MOBICOM98
25. Lightweight Extensible Agent Platform, World Wide Web, http://leap.crm-paris.com
26. Gerd Kortuem, Jay Schneider, Dustin Preuitt, Thaddeus G.C. Thompson, Stephen Fickas, Zary Segall, “When Peer-to-Peer comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad hoc networks”.