Numi: A Framework for Collaborative Data Management in a Network of InfoStations*

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
(oratsi2, kodeswar, joshi, finin, yeyesha) @cs.umbc.edu

Abstract

A network of infostations offering high bandwidth islands of data connectivity has often been suggested as a viable alternative to cellular WAN for providing network connectivity for mobile devices. Current data management models proposed for such networks treat these infostations mainly as passive entities that offer devices in range, access to needed information. In this paper, we propose a novel approach wherein infostations actively participate in delivering content even to devices that are not in range. Other participating mobile devices that have excess spare capacity are used as carrier nodes to transport data towards remote devices. Short range wireless technologies enabling creation of ad-hoc peer-to-peer networking capabilities between devices are used to form transient communities to exchange information. The infostations are responsible for selecting appropriate carrier devices to use so that these carriers meet the device in need at the appropriate time to participate in this collaboration. Through this, dependency on the more expensive WAN networks is minimized. We present a design of our proposed framework and its components along with a prototype implementation.

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(802). Also, there has been a lot of advances in the field of short-range wireless communication technologies like Bluetooth(Paper), homeRF(Negus & Landsford 2000) 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.

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 this migration. These include limitations on power consumption, smaller display, processing power etc. Also, most mobile devices depend on a wireless infrastructure for their communication needs. Such wireless infrastructures suffer from limitations of restricted range, low bandwidth, limited coverage, higher costs etc. Services that are offered on these mobile devices must be aware of these limitations and must efficiently overcome them(Chen & Kotz 2000).

Data management to support applications on a mobile device 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 resource sharing among peer users to maximize their efficiency as a whole in a network of infostations. We believe that by coordinating such ad-hoc collaboration among mobile users, individual user data needs can be satisfied in an efficient manner. We present an intelligent data caching strategy that utilizes user mobility patterns to efficiently manage data demands in the network. Using infostations to facilitate and coordinate caching of selective future data needs on a device, more efficient usage of the available communication infrastructure can be attained. We present a novel communication mechanism that uses collaboration among mobile devices to transfer data to the user in need. Using mobility patterns, data intended for a specific user can be piggy-backed on other devices heading towards that user. Through this mechanism, infostations can now actively service remote users, thereby reducing the dependency of those users on the expensive WAN. We have built a working prototype of our

*This work was supported in part by NSF awards IIS 9875433 and CCR 0070802, and the Defense Advanced Research Projects Agency under contract F30602-00-2-0 591 AO K528.

system which is presented in this paper.

Related Work

Cellular voice and data networks facilitate "any time, any where" connectivity but they are expensive and offer low bandwidth. Infostation networks(R.H. Frenkiel, B.R. Badrinath, J. Borras, and R. Yates, 2000) 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. 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 infostation(G. Wu, C. W. Chu, K. Wine, J. Evans, and R. Frenkiel 1999).

Several models have been suggested for an infostation type system. In (Iacono & Rose 2000), the infostations are owned by small enterprises with low speed wireline connection between them. Assuming that a mobile users 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 to facilitate data hoarding on mobile devices(U. Kubach and K. Rothermel 2001). 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(WICAT), 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. (Ye, Jacobsen, & Katz 1998) 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 using 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 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 infostation 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 is no facility for the infostations to attempt to deliver content to these stranded users (forcing them to use the WAN and hence the associated costs). 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.

Numi Framework Components

Service Portals, Mobile Hosts and Services

The key components of our Numi framework include Service Portals (SPs), Mobile Hosts (MHs), and Services. SPs are infostations running the Numi platform and hosting services that can be used by MHs. The SPs are connected by high speed links to the rest of the wireline network (unlike the model proposed in (Iacono & Rose 2000)) 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.

Mobile Hosts (MHs) are wireless mobile devices running the Numi Platform. MHs can communicate both with SPs and neighboring MHs that are within range. 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, it is assumed that MHs move along predetermined routes (like highways). 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 the actual applications that are hosted by our framework and are available to the MHs. Each Service in our system consists of a Service Specification, a Service Agent and Service Data Volume (SDV). The Service Speci-

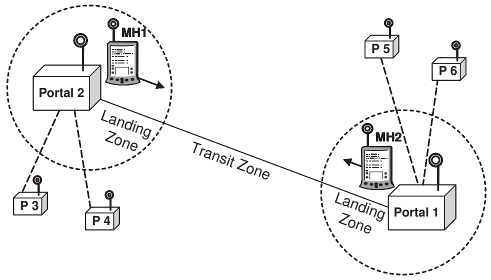

Figure 1: Network Model

fication is a meta description of the service. This description could include features that are offered to the user, version, minimum capabilities of a device to support this service etc. We envision the usage of semantic languages like RDF(Lassila & Swick 1999) or DAML(Language) to efficiently describe a service. A Service Agent is the code that is capable of executing on the MH so as to realize the service to the user. A single service may in fact be comprised of several service agents, each agent targeting a specific platform (PalmOS, PocketPC, Linux). Also, each service will have a portal service agent running on each SP in the network that advertises the presence of this service in the network and assists corresponding service agents running on MHs. Service Data comprises the actual data that is used by this service. A Service Data Unit (SDU) is the smallest indivisible unit of data for a service (individual songs for a music service or html pages for a newspaper service). A collection of such SDU forms a SDV. Data updates to MHs takes place in terms of these aggregate units to maximize communication efficiency.

A typical usage scenario of our framework would be initiated by a user on a mobile device in a *landing zone*. The device contacts the corresponding SP for a list of relevant services that can be offered to the user. The selected service agent is activated on the MH and an initial SDV is transferred to the host. The user can now start using the service. Additional SDVs needed for the service are then transferred to the device in an intelligent manner utilizing a combination of inter-portal scheduling and data delivery using other MHs present in the network.

Numi Platform Agents

We designed our framework with the goal of reusability. We have developed a set of agents 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 in our framework are also implemented as agents. Service providers can implement agents conforming to our specifications and these agents can be seamlessly introduced into our 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. These framework agents include the heartbeat generator, location monitor, message handler, logger, task scheduler, data handler and a service manager.

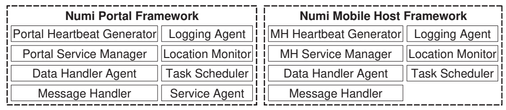

Figure 2: Numi Portal and Mobile Host Platforms

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. 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.

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 agent uses the FTP protocol.

Service agents run on top of our Numi platform 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 SP service manager notifies the corresponding service agent. This service agent then transmits the initial SDV to the MH and schedules subsequent volume updates throughout the network. Our current implementation can schedule volume updates only up to the next portal on the route. We are working on extending this to include all SPs on any given devices route. In addition, whenever a volume is scheduled at an SP, the service agent at that portal is responsible for ensuring that this SDV gets delivered to the MH. In case a device does not show up at a *landing zone* at it's prescribed time, the service agent at that SP then actively starts looking for other devices whose routes indicate that they are heading in the direction of that MH and attempts to use them to deliver the next data volume. The service agents continue to track the MH till it arrives and obtains a new SDV. Once this is done, the SP service agent notifies the next SP(s) on the route to schedule future volumes for this MH. A service agent on an MH offers a service to the user. Whenever a device enters a *landing zone*, the service agents on that MH receive SDV 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 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. In some cases, these queries cannot be handled by neighbors. However, since these interactions are being logged by the logger, 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 uses each service agents' service specification to generate a list of available services with their description. This list is maintained in HTML format and is served out to MHs through a web server that is run on each portal. The service manager at the MH, can activate, suspend or terminate other service agents. Also, this manager monitors system usage by each service agent including statistics like the amount of memory used, running time, messaging overhead incurred etc. MH service manager is also responsible for maintaining a location dependent bookmark that points to the URL being advertised by a nearby SP. This URL contains a list of services that are available at that portal.

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 SDV for currently running services (transparent to the user).

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 others heartbeats (Step 1). The SP stores

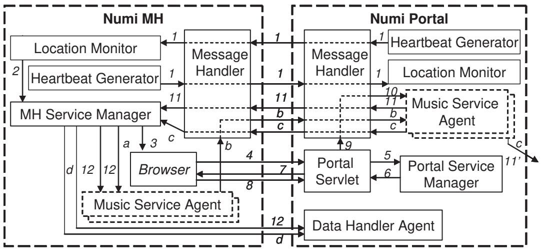

Figure 3: Mobile Host to Service Portal Interaction

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 SDV and notifies the MH Service Manager (Step 11). Several factors are used in formulating SDVs 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 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 SDV needed for each selected service (Step 12) and activates corresponding MH service agents.

In the case of request for additional SDV 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 SDV needed for this service (Step c). Using the data handler agent, these volumes are transferred to the MH (Step d).

Portal to Portal Interaction

This interaction is used between SPs to efficiently handle a mobile users 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.

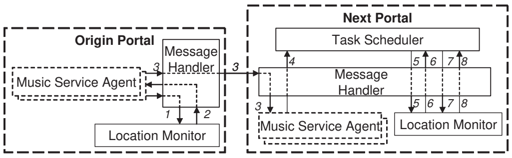

Figure 4: Portal to Portal Interactions

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 SDV for a new service that the MH has requested or can be continuation SDV 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 SDV 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 SDV. 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 SDV 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 SDV is given to one of these MHs with spare capacity. The SDVs 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

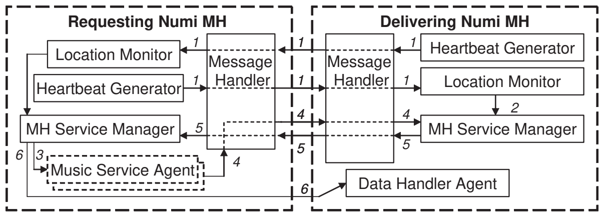

Figure 5: Mobile Host to Mobile Host Interactions

The Node to Node interaction is employed by an MH to obtain any additional SDV 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 SDV, a "No Such Volumes" message is sent to the requesting MH. The requesting MH continues the search for the next SDV by initiating Node to Node interaction with other MHs in its range.

Implementation and Experimental Results

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.

Figure 6: Mobility Coordinator

Experimental Setup

We set up network of 3 SP: P1, P2 and P3. *Transit zones* between SPs were divided into cells as shown in figure 8. There are three mobile users Susan, Bob and Jim with iPAQs. Susan has MH1; she starts her trip at P1 and plans to go to P3 through P2. Bob has MH2; he starts his 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. Susan is using Music Service and eBooks Service; Bob is using Newspaper Service; Jim is using eBooks Service.

"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 Susan requested the Music Service and the Newspaper Service through a MH to Portal interaction. MH1 gets two SDV and Susan starts her trip form P1 to P2. MH1 has enough data to last Susan 80% of her trip. She will run out of data in C3. P1 notifies P2 about service provided to Susan through portal to portal interaction. P2 determines that she does not have enough data to last her till she reaches P2. Meanwhile, Bob with MH2 is

Figure 7: Mobile Host Interface

getting ready to depart from the *landing zone* of P2 (towards P1). He is using the Newspaper Service. P2 gives MH2 the volumes for Bob's Newspaper Service and it also gives the next set of SDVs needed for Susan's Newspaper and Music Services. Bob passes by Susan in C3. MH1 contacts MH2 and obtains the next set of SDVs for the Newspaper Service and Music Service. Susan 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 Susan continues on her way from P2 to P3 though C5, C6, C7 and C8. She is continuing to use Music Service and Newspaper Service. MH1 has enough memory to store SDVs for her services to last her from P2 to P3. She departs P2's *landing zone* towards P3 with no plans for detours or delays. Along the way in C7 she sees a coffee shop and decides to stop for a cappuccino. As she stands in line to place her order she continues to read her newspaper and listen to her music. Meanwhile P3 determines that Susan 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 SDVs for Jim's Newspaper Service and also receives music and newspaper SDVs for Susan. By this time, service agents on Susan's iPAQ would have realized that they are running out of data and will start querying the neighborhood. As Jim passes by the coffee shop, MH1 and MH3 discover each other and MH1 uploads needed SDVs from

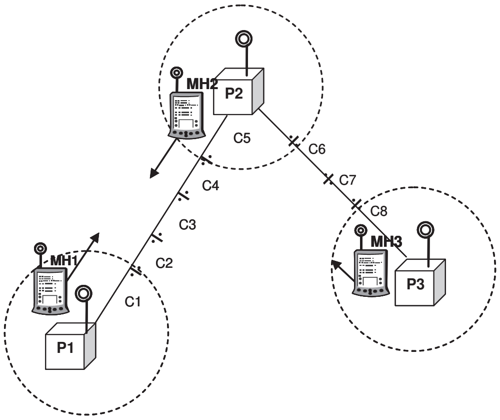

Figure 8: Experimental Network

MH3. Susan gets her cappuccino and resumes her trip. She reaches P3 with out 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 the 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.

Currently we are extending our framework to support *n*-hop scheduling of data on SPs on an MH's route. We are working on implementing intelligent services that can monitor usage patterns so as to adapt the data delivery mechanism to offer optimal efficiency. Ability for users to modify the level of collaboration for their MH is being developed. We are adding support in our framework to allow an MH to issue a query in a *transit zone*, whose result will be made available to that MH when it arrives at its next *landing zone* or will be carried to it by other MHs. We will also be migrating our framework to use Bluetooth protocol for ad-hoc interactions. We are also building a simulation model of our approach to study characteristics of our framework such as different scheduling mechanisms, scalability, performance when MH routes are not known and performance when there is a combination of MHs whose routes are known and others whose routes are not. Also, using mobility patterns, number of MHs passing through an SP and their levels of collaboration, we would like portals to generate an analytical estimate of the bandwidth that is available so as to offer QoS guarantees to its users.

References

IEEE Std 802.11-1997, Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Publications, 1997.

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

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

Iacono, A., and Rose, C. 2000. Bounds on file delivery delay in an infostations system.

Language, D. A. M. World Wide Web, `http://www.daml.org`.

Lassila, O., and Swick, R. 1999. Resource Description Framework. `http://www.w3.org/TR/1999/REC/rdf-syntax-19990222`.

Negus, K. J. Stephens, A. P., and Landsford, L. 2000. HomeRF: Wireless Networking for the connected home. *IEEE Personal Communications* 7:20–27.

Paper, B. W. World Wide Web, `http://www.bluetooth.com/developer/whitepaper`.

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

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

WICAT, Infostation Project, P. U. WWW, `http://wicat.poly.edu/infostation.htm`.

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