MoB: A Mobile Bazaar for Wide-area Wireless Services

Rajiv Chakravorty, Sulabh Agarwal*, Suman Banerjee
Department of Computer Sciences
University of Wisconsin-Madison
Madison, WI, USA
{rajiv,sulabh,suman}@cs.wisc.edu

Ian Pratt
University of Cambridge
Computer Laboratory
Cambridge, UK
ian.pratt@cl.cam.ac.uk

ABSTRACT

We introduce MoB, an infrastructure for collaborative wide-area wireless data services. MoB proposes to change the current model of data services in the following fundamental ways: (1) it decouples infrastructure providers from services providers and enables fine-grained competition, (2) it allows service interactions on arbitrary timescales, and, (3) it promotes flexible composition of these fine-grained service interactions based on user and application needs.

At the heart of MoB is an open market architecture in which mobile users can opportunistically trade various services with each other in a flexible manner. In this paper we first describe the overall architecture of MoB including various enablers like user reputation management, incentive management, and accounting services. We next present our experience from both simulations as well as our prototype implementation of MoB in enhancing application performance in multiple different scenarios — file transfers, web browsing, media streaming, and location-enhanced services.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication, Network communications*

General Terms

Design, Experimentation

Keywords

Wireless Services, Incentives, Reputation, Wide-area Wireless

1. INTRODUCTION

Mobile devices such as hand-held PCs, personal digital assistants (PDAs), and smart cellular phones, are increasingly gaining popularity worldwide. In order to satisfy the needs of this growing population of mobile users, cellular data networks are being universally upgraded to higher data rates and 802.11-based public WLAN hotspots are mushrooming around the globe at various opportunistic locations.

Despite the promise of ubiquitous connectivity based on these encouraging developments, many wireless devices lack access to the Internet infrastructure (either through WLANs or cellular data networks) in various wide-area mobility scenarios. There are various reasons that contribute to such intermittent connectivity. WLAN coverage is usually spotty and is limited to specific public hotspots; hence mobile devices need to rely on cellular data networks to acquire greater degree of continuous coverage. However, providing adequate cellular coverage in any region requires a sufficient number of (cellular) base stations which can sometimes be prohibitively expensive. Based on the degree of such investments made by individual cellular providers in different geographic regions, corresponding customers experience good connectivity in certain locations and poor (or no) connectivity in others. Even in areas of good connectivity, cellular links are sometimes plagued with problems of high latencies, relatively low bandwidths, and occasional link-stalls that lead to poor user experience. Such connectivity problems always lead to poor performance of ‘staple’ Internet protocols and applications running on the mobile devices.

To overcome this existing impasse in mobile applications and services, we present Mobile Bazaar or MoB, an open market, collaborative architecture to improve data services for wide-area wireless users. MoB changes the model of wide-area wireless data services in the following fundamental ways: (1) it decouples infrastructure providers from services providers and enables *fine-grained competition,* (2) it allows service interactions on arbitrary timescales, and (3) it promotes flexible composition of these fine-grained service interactions based on user and application needs.

Fine-grained competition

Wide-area wireless data services today are primarily available through a number of cellular service providers. In most typical scenarios, each user chooses *only one* of these cellular providers and signs a relatively long-term contract with that provider for all wireless data services. (By long term we mean a time duration in the order of hours, days, weeks, or months.) Although customers can choose between cellular providers they exercise their choice with large time gaps. We call this coarse-grained competition. As described above, the wireless coverage of different cellular providers vary in different regions. Hence, it is not uncommon for customers to be unable to access the Internet through their existing providers over certain periods of time. In contrast, MoB defines mechanisms to enable fine-grained competition, through which users have the flexibility to choose and change providers at arbitrarily small timescales. The key advantage of such an architecture is that it allows each user the ability to choose the “best” provider based on his current location and on the characteristics of his immediate wireless environment. Additionally, it allows the user the ability to temporarily choose multiple providers simultaneously in order to meet the performance requirements of high-bandwidth applications.

Finally, users in MoB are not required to acquire all necessary services directly from the cellular providers. Any user in the system is permitted to *resell* his unused resources. For example, an idle cell-phone with a fast connection to its provider’s network can sell bandwidth to the user of a laptop that is experiencing a slow

*Sulabh Agarwal is currently unaffiliated.

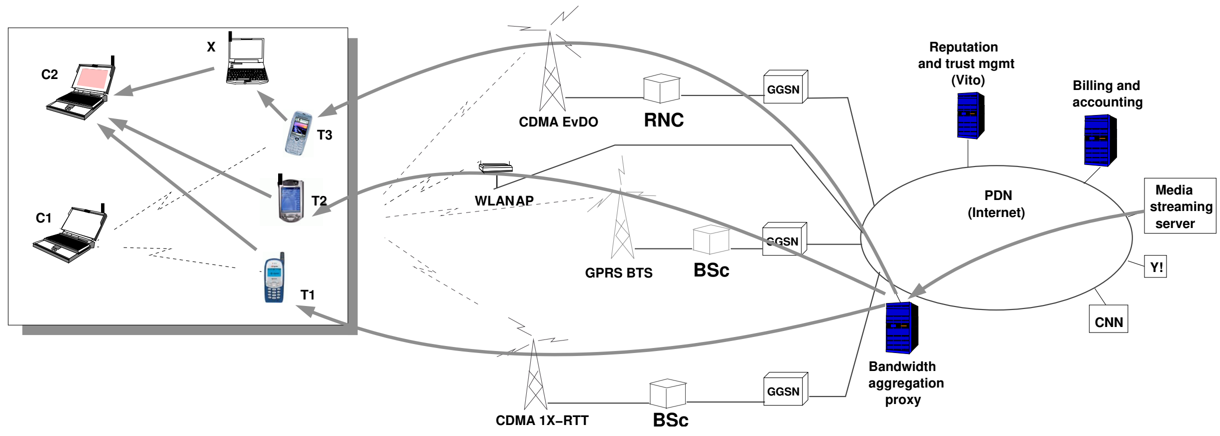


Figure 1: MoB system architecture for incentive-induced collaborations and an example of a bandwidth aggregation service interacting with a media-streaming application.

connection to its provider's network. A payment system is used to manage such resource trades. There are a number of advantages of this open market structure. A user in need of additional resources can purchase idle resources from nearby users for small time periods, thus boosting application performance on-demand. This model of open resource trading also decouples the provider of the wireless access infrastructure from the provider of the service. Therefore, users are no longer limited to the services and rates offered by their infrastructure provider. We believe that this architecture can have far reaching implications for the entire wide-area wireless industry. It will open this industry to greater competition, as happened in the long distance telephony market in the US in the mid 1990s. (A new Telecommunications Act came into effect in the US in 1996, which required that incumbent phone companies to allow their competitors access to their infrastructure with fair fees. Under this new structure, telephone subscribers were no longer tied to their local phone company for fixed rates and services, and instead had the freedom to move their long distance calls to providers offering better service or lower rates.)

Services in MoB — An Application-layer approach

The goal of MoB is to enable incentive-induced service collaborations between independent mobile devices. A *bandwidth aggregation* service is a simple example of such collaboration (shown in Figure 1). Consider a wireless user (C_1) in a static public environment (e.g., a coffee shop or a shopping mall) or a mobile environment (e.g., a moving bus or train). Let us call this user's device the *customer device.* Typically there are a large number of other users in these environments carrying other network-enabled devices, e.g., cell-phones, laptops, and PDAs. Each of these devices has its own mechanisms to access the wide-area Internet infrastructure. For example, a 3G-enabled cell-phone (T_1) can connect through a 3G-capable cellular provider's network, while a PDA with an 802.11 wireless interface (T_2) can connect through a WLAN-based service infrastructure like Boingo Wireless. Let us call these devices *trader devices.* (The rationale behind the terminology will be apparent.) At any instant many of these trader devices are idle. The goal of MoB is to define mechanisms that allow customer devices to harvest available resources and obtain necessary data services from such in-range, idle trader devices. In the example in Figure 1, customer C_1 first discovers a number of trader devices — T_1, T_2, T_3, that are available in its vicinity. Subsequently it chooses a subset of these trader devices, T_1, T_3, connects to them and uses them simultaneously as if they were its own wireless interfaces. Thus C_1 achieves significant bandwidth aggregation while T_1 and T_3 can recover their costs through monetary payments. In the rest of this paper we will also refer to these devices as customers and traders.

High-bandwidth connectivity is not the only service that traders in MoB can offer to their customers, though it certainly is a natural one. We envision MoB creating an open marketplace among participating traders and customers, in which a variety of advanced, application-layer services will be traded. The following are a few examples of such services:

Location determination: Consider a mobile user carrying a wireless PDA that is equipped with appropriate street map software and database. In order to function as a navigational tool, the PDA needs to be attached to a GPS (or any other location-tracking) device. Unfortunately the user may not have an appropriate GPS service available to him. In the MoB architecture, this user can purchase such information from any in-range MoB trader that has the relevant information through its own mechanisms, e.g., using GPS, manually configured, or by purchasing this from another trader in turn.

Time synchronization: Consider a distributed set of wireless devices that are participating remotely in a collaborative application, such as a mobile multi-player game. In many cases such gaming devices may require accurate time synchronization. While such time synchronization is possible using the Network Time Protocol (NTP) [25], such an operation can be fairly expensive due to high variability on end-to-end network paths, especially involving multiple wireless links.

However, MoB allows the following simple technique for efficient time synchronization. Each wireless gaming device independently locates a corresponding cellphone-based trader that is willing to announce the current time. If the cellphones themselves are synchronized accurately to a global time (which they usually are), the gaming devices will automatically be synchronized.

Web proxy caching: A user browsing web content over relatively slower and more expensive cellular links may first choose to locate cached copies of the same content within its wireless vicinity. A MoB trader device with the appropriate web content in its local cache can serve as a proxy on-demand in such scenarios, thereby improving browsing performance and reducing overall cost.

Bandwidth aggregation for media streaming: In order to receive a high-quality video streams in a wide-area cellular environment, a bandwidth-constrained user (say, using a GPRS network) may request multiple MoB bandwidth traders (using different 3G networks) to serve as wide-area interfaces. Using an application-level network proxy, the video stream can then be intelligently striped over multiple such interfaces and lead to overall improved user perception.

Peer-to-peer data search: A user conducting a Gnutella or Kazaa-style peer-to-peer data search and download operations in the wide-area environment may suffer from loss of connectivity and poor performance. In order to mitigate such loss of performance (and also potentially avoid monetary costs of downloads over cellular links), the user may first attempt to locate and download the data within his physical neighborhood. Only if such a search is unsuccessful, the user may attempt a download across wide-area cellular links.

Traffic filtering: Consider a resource-constrained wireless user that is currently obtaining bandwidth services from one MoB trader as described above. If the MoB trader is suitably capable, it can also serve as traffic filter that detects and eliminates malicious content, e.g., worms, targeted at the unsuspecting user.

Such advanced application layer services in MoB are advertised by traders and discovered by customers using the Service Location Protocol (SLP) [12]. In this paper we present experimental results for four such application layer services that we have developed through a prototype implementation, namely — web downloads, location determination, file transfers (both peer-to-peer style and specific location based), and bandwidth aggregation for media streaming.

It is possible to achieve service interactions in MoB using both single hop as well as multi-hop paths. In Figure 1, we show an example of a multi-hop path based interaction in which customer C_2 has requested for web proxy caching services from any trader in its vicinity and trader T_3 has offered this service to C_2 by relaying it through an intermediary, X. However, management of such service interactions over multi-hop paths requires more coordination. For example, the service cost needs to be appropriately distributed between X and T_3. Also since the service responsibility is divided between multiple entities, the customer may not immediately know whom to hold responsible (and penalize) during a failure.

We avoid this in MoB, by requiring that all service interactions be pair-wise (or single-hop). In Figure 1 we would therefore require customer C_2 to purchase the web proxy caching service from X who in turn would negotiate a similar service for this purpose from T_3. There is no direct interaction between C_2 and T_3 and the service charge in each of the two interactions can be independently set (though if X is a strategic participant, it will ensure an overall profit through the two interactions). Such a design simplifies various service management functionalities required in MoB. Additionally, we believe that most typical service interactions in MoB will be in environments where devices are in direct communication range of each other, e.g., within a coffee shop or a bus. Due to device proximity in these environments multi-hop interactions will be relatively rare.

A customer may compose different service interactions from multiple traders into one desired application. For example, customer

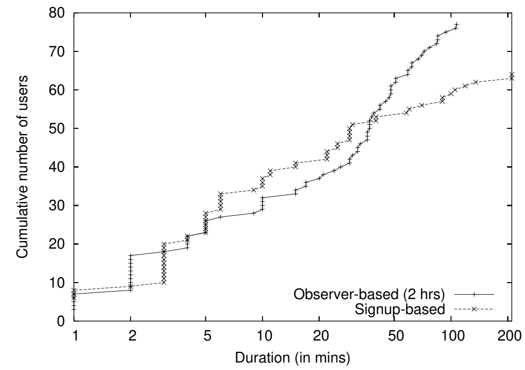


Figure 2: Distribution of user persistence in coffee-shop environments (x-axis on log-scale).

C_2 may be interested in recommendations for Italian restaurants in his vicinity. He may avail the location information from trader T_2 and a blog on Italian restaurant recommendations from trader T_3. From the system's viewpoint, however, these service interactions are independent of each other.

Finally, we require that all service interactions in MoB be implemented in the application layer. This is because application layer mechanisms will be easier to deploy without requiring any change to underlying network protocol behavior.

Now consider a multi-hop service interaction in MoB — say customer C_2 is performing a peer-to-peer file download from T_3 via X. Based on our above requirements, there are two independent single-hop service interactions that enable this download — one between C_2 and X, and the other between X and T_3. We can imagine this download to be progressing using two independent TCP connections, one for each hop in the path. We use this example to highlight a key difference of such data downloads in MoB with that of data transfer mechanisms in various ad-hoc networking scenarios. Data transfers in ad-hoc networks use a (on-demand) routing protocol, e.g., DSR [13], AODV [28], to construct network layer end-to-end paths on which such transfers will proceed. In contrast, multi-hop interactions in MoB do not involve any routing protocols. In particular we do not define any such network layer mechanisms as part of MoB. All multi-hop interactions are composed of multiple single-hop application-layer service interactions. Although such multi-hop interactions maybe viewed as a single multi-hop path, the flavor of the interactions in MoB is significantly different.

Our approach of application-layer services in MoB is significantly different from multiple related and prior efforts, namely 7DS [26], UCAN [23], CAPS [19], ORION [16], and iCAR [35]. We present a detailed comparison between MoB and other such approaches in Section 6.

Pricing and Reputation

The open market in MoB is implemented in a *laissez faire* approach with no control or regulation on advertised services and their corresponding prices. All pricing and purchasing decisions are left to the individual users. As with any such free market system, it is expected that the system itself will dispense with inefficiencies in a more deliberate and quick manner than any regulatory body can. Although individuals in MoB can arbitrarily price their services, open market economics dictate that intelligent traders will price their services based on various competitive forces. In order to enable such an open market, we require (1) a reputation and trust management system, and (2) a billing and accounting system, both of which can ideally be implemented by independent providers as third-party services. In this paper we define one possible design and implementation of the reputation management and accounting system — *Vito.* Our design of this system is modeled on eBay (see http://www.ebay.com) — a large person-to-person online auction site (with more than 4 million open auctions at a time), which implements its own reputation management system. We present design rationale and details on Vito in Section 2.

Applicable environments

An environment like MoB is perfectly applicable to various scenarios where there are many opportunities of collaboration between in-range devices. A coffee-shop is a perfect example of such a scenario where users often spend tens of minutes in relatively close proximity of numerous other users. To evaluate resource sharing opportunities in the context of MoB in these environments, we conducted a study of user persistence in multiple neighborhood coffee-shops. The goal of this study was to collect data on how long a user stays in a coffee-shop. The data was collected using two different techniques: (1) a time-sheet left near the counter that allowed people to "sign-in" and "sign-out" (not all coffee-shop users participated), and (2) an observer spent two hours in a specific coffee-shop to monitor and collect such data. We present the results of this study in Figure 2 which shows that more than two-thirds of the users spent more than two minutes during their visit, and at least 50% of the users spent ten minutes or more. Additionally, there are a significant fraction of users who spend more than 30 minutes in each visit. It is clear that there are significant opportunities of relatively long-lived MoB interactions that are possible between devices carried by such users. Other examples of such environments include static scenarios, e.g., hotspots in shopping malls, and mobile scenarios, e.g., users in a bus or a train.

Salient features

The following are the salient features of MoB:

Open market architecture: MoB is an open market architecture that offers a common ground for integration of heterogeneous mobile terminals, networks, services and applications. MoB is open in the sense that any device can autonomously advertise services independent of all other devices. Each customer of an advertised service can have separate service level agreements with the traders that provide the service. These service level agreements can potentially last over small timescales. Additionally, the architecture allows customers the flexibility to resell idle resources to other customers.

Better Performance through Wireless Diversity: MoB allows users to better exploit the significant diversity in the wide-area wireless environment. For example, in the context of data forwarding (bandwidth) service, a mobile device can exploit the diversity in the wireless coverage of different technologies (e.g., CDMA and GPRS in cellular networks, 802.11 b/g in wireless LANs), networks (e.g., Sprint, AT&T, Boingo), and channels (defined by specific frequencies used for communication). Customers in MoB can take advantage of diversity by intelligently striping data across multiple wireless links with good bandwidth characteristics at the current location and under current conditions, thereby improving application performance.

Incentive-based Collaboration: MoB enables incentive-based collaboration among in-range mobile devices. Traders in MoB provide services in return for monetary payments. Thus customers gain improved performance while traders profit by reselling idle resources.

Customization and Support for Diverse Applications: MoB allows applications with varying service requirements to be implemented in the wide-area wireless environment. For example, a user can utilize caching services from a specific trader (by paying for the service) only when he is browsing the web. Another user can obtain location services from multiple (location-aware) traders, e.g., E-911- capable cellphones [1], to customize his interactive navigation application, but only while he is driving. Similarly, a third user requiring high-bandwidth services for a download intensive application is no longer tied to his single service provider. Instead he can aggregate bandwidth resources from multiple traders to meet the application requirements. Such on-demand customization is not feasible in today's model, where each user of a navigational application needs a long-term (in the order of hours) subscription to a satellite-based, coarse-grained location-tracking system like GPS.

Roadmap

The rest of this paper is structured as follows. In the next section we present an overview of the MoB architecture including various components. In Section 3 we describe some implementation details. In Section 4 we present evaluation results from our implementation as well as some simulation based studies. In Section 5 we discuss some additional issues relevant to the MoB architecture. In Section 6 we present a detailed comparison of MoB with prior work in related projects and finally we conclude in Section 7.

2. MOB ARCHITECTURE

MoB defines a flexible architecture for service interactions between heterogeneous mobile devices with diverse wireless access mechanisms and technologies. There are three basic components of the MoB architecture: (1) an infrastructure which allows devices to connect to the Internet e.g. cellular networks, WLAN-based access infrastructure, or any combination thereof, (2) mobile devices, e.g., laptops, PDAs, cellphones, with the ability to communicate with each other and with the infrastructure, and (3) third-party services for accounting and billing as well as for reputation and trust management.

A key construct in the MoB architecture is the formation of a the dynamic network of the mobile devices within which resources are traded. As explained in the previous section, mobile devices in MoB can be either customers, traders or both. A customer device requests and acquires services from trader devices within its range. A mobile device may simultaneously be a customer for some specific service and a trader for another service. Additionally, a trader can provide a service to its customers by itself acquiring necessary services from one or more other traders (e.g., device X in Figure 1). In fact, it is also possible to model the infrastructure provider as a trader in the MoB framework. This way, the framework allows us to handle scenarios where the infrastructure provider can participate in the resource and service trading market. Service trading decisions in MoB are independently governed by user preferences and policies. Some policies can also be determined by device characteristics, (e.g., form factor, uplink bandwidth to the infrastructure and residual battery power). Hence, a user of MoB might allow his device to provide data forwarding services if and only if it is idle and has battery power above a configured threshold.

Finally there are two important third-party services in MoB that are centrally managed, namely (1) a reputation and trust management

[1] See http://www.fcc.gov/911/enhanced/

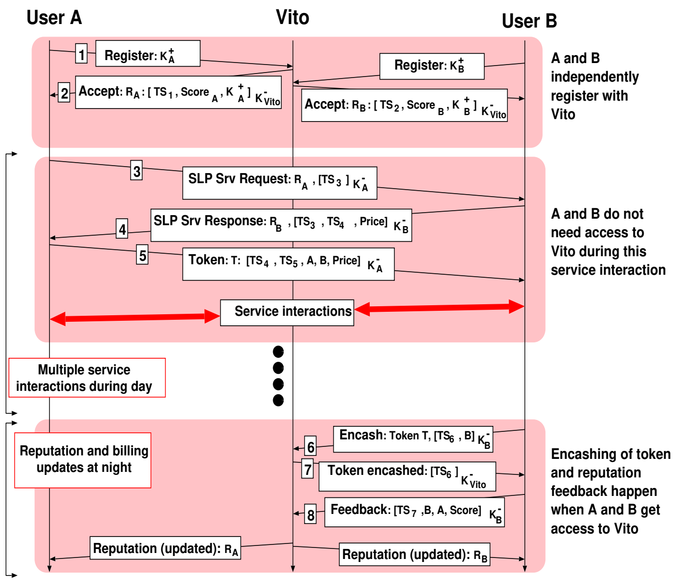


Figure 3: A sequence of operations for a successful service trade in MoB. K_X^+ and K_X^- denote the public and private key respectively of entity X. For a message, m, $[m]_K$ denotes the concatenation of the message, m, with the encryption of message digest (say using SHA-1) using the key, K. TS_i indicates a time-stamp. In a typical scenario it is likely that multiple service interactions happen during a day and reputation updates happen when the users connect their device back to the Internet (say at night). Note that the *Encash token* operation by B is an implicit positive reputation feedback for B.

ment system, and (2) a service accounting and billing system. In addition, there are some optional third-party services that can also be deployed in MoB to enhance the performance of specific applications. One such example is a bandwidth aggregation service in which a third-party can deploy a bandwidth aggregation proxy in the wired Internet as shown in Figure 1 In order to receive a high quality media stream from the media streaming server, C_2 has requested (and purchased) bandwidth services from three different traders, T_1, T_2, and T_3, each potentially using a different wide-area wireless interface. The media streaming server itself is unaware of multi-path capabilities enabled by resource-sharing. Therefore, a bandwidth aggregation proxy is needed to *intelligently* stripe the media stream across the three trader devices employed by the customer depending on individual wireless path characteristics. As part of our MoB prototype we have implemented and deployed a bandwidth aggregation proxy and have used it for different wide-area services. Prior research has shown the benefits of various other proxy and caching services for wired as well as wireless end-hosts. Each such service can potentially be deployed as an independent third-party service in MoB.

Modes of Operations: In general, devices in a MoB can interact in multiple different ways — (1) *incentive-based with no trust assumptions,* where a trader provides services to a customer based on financial incentives, and both parties use a central reputation management system (like Vito) to examine past trade histories and derive trust for each other; (2) *incentive-based with trust assumption,* where the customer provides financial incentives for the trade and both parties in a trade directly trust each other (e.g., due to multiple successful interactions in the past the two parties have direct faith in each other without requiring a centralized reputation management entity to induce mutual trust); and 3) *altruistic,* where there is perfect trust between the participants and no financial incentives are required to enable resource and service sharing, e.g., a within a friends' network. Only the first of these three options require a centralized reputation management system; the first and second options require both a reputation management system and a billing and accounting system; while the third option just requires a service location and discovery technique.

Reputation and trust management

We will next describe the operations for MoB users employing the incentive-based mode of operation with no trust assumptions, in which reputation management and accounting support play a central role. (The sequence of operations in the remaining two modes are subsets of this mode, and hence it easy to infer their operations.)

Design rationale for Vito

We now explain the reasons for some of the decisions made in the sequence of operations as described above.

In general any reputation and trust management system can be deployed in MoB as a third-party service. In this section we focus on one such possible choice, *Vito.* We have designed and implemented Vito to serve both reputation management and accounting service functionalities for all its users. It is modeled on eBay's reputation and trust management system, that successfully manages more than 4 million person-to-person auctions at any time. Note that eBay offers no warranty for its auctions; it only serves as a listing service while the buyers and sellers assume the risks associated with transactions. There are fraudulent transactions for sure but the overall rate of successful transactions remains quite high for a market as "ripe with the possibility of large-scale fraud and deceit" [17].

eBay attributes its high rate of successful transactions to its reputation system. After a transaction is completed, the buyer and the seller have an opportunity to rate each other. On a successful trade, the buyer and the seller typically provide a positive reputation feedback for each other. Similarly an unsuccessful trade leads to negative feedback. While it is possible for a user to gain false reputation, it would cost a user money to do so (due to appropriate transaction fees). This financial barrier makes such a reputation system more reliable, and buyers trust it more as a result.

Vito Design: Like the eBay system, Vito is centralized and is hosted as a third-party service in the wired Internet. Each user registers himself with Vito and obtains a timestamped reputation certificate. The reputation certificate issued by Vito indicates both successful and unsuccessful transactions involving the specified user. During a service trade a customer and trader will typically examine each other's reputation certificate. They may may choose to ignore the other's certificate if the timestamp is very old. The negotiated price for a MoB trade can depend on the reputation of the participating parties.

The actual trades in MoB are conducted independent of Vito (potentially when the participants do not even have access to the Internet). As part of each trade, the customer and the trader exchanges certified reputation feedback scores for each other. At a later time, they independently upload these feedback scores to Vito, who verifies these certificates and periodically distributes updated reputation certificates to the users for future trades. We discuss various performance aspects of Vito in MoB in Section 4.

It is, however, not necessary that all MoB devices use Vito as the reputation and trust management system. In fact there might be multiple reputation management systems that co-exist in the MoB architecture, each implemented as a separate third-party service with its own user base. Each user, A, can independently decide to register with one or more of these reputation services and perform trades with any other user, B, who trusts A's reputation certificates.

Operations in MoB

In order to participate in the MoB architecture, typically each user, A, has to register with both the reputation and trust management system and the service accounting and billing system, e.g., Vito, using its public key, K_A^+ (message 1 in Figure 3).

Once Vito's reputation management system accepts a user's registration, it issues a reputation certificate, R_A, appropriately signed by Vito (message 2). R_A includes a time-stamp, and a separate count of all positive and negative feedback for A (indicated as $Score_A$). At this initial instant, the user has no reputation state at Vito. Equipped with the reputation certificate, A is able to perform subsequent trades with other users.

All services in MoB are discovered and advertised using the Service Location Protocol [12]. To request any service in its wireless vicinity, an *SLP User Agent* in A sends a *Service Request* to the SLP multicast address (239.255.255.253) and port 427 with a TTL of 1. The choice of the TTL stems from our pair-wise requirement for service interactions in MoB. We explain all service interactions in MoB with the following simple scenario.

A data forwarding service scenario: Consider the scenario, where A seeks 30 Kbps data forwarding service from any trader in its vicinity. In such a case, A will include this information in the broadcasted *Service Request* message (message 3). A also includes its own reputation certificate in its service request. The *SLP Service Agent* of any in-range device, B, that is willing to provide the desired service can respond back to A. The response (message 4) includes B's reputation certificate, the service description (say, it is willing to provide only 25 Kbps), and a price quote. A can potentially receive multiple such responses. On receiving these responses, A can choose a subset of devices, $\mathcal{S}$, as traders, based on user-configured policies, and send a *Service Acceptance Notification* to each such trader, $B \in \mathcal{S}$ (message 5). This notification includes a time-stamped *token* signed by A using its private key that indicates the payment amount for B. Subsequently B configures itself as a data forwarder and starts accepting data traffic from A analogous to an Internet router. In this example, B will operate as an Internet NAT device for A as it forwards traffic.

At a later time, B will present this token to Vito, which appropriately charges and credits A and B respectively for the payment amount (messages 6 and 7). We also assume that the token serves as a positive feedback for B from A for this trade. Therefore as B encashes the token at Vito, it gains positive feedback points. Vito charges B a transaction fee (some percentage of the trade price) for the positive reputation that B gains through token encashing.

Once B receives credit from the accounting and billing system for this trade, it will typically choose to report a positive feedback for A (message 8). Thus in this proposed system, B is responsible for reporting both its own positive feedback (in form of the token from A) and an explicit positive or negative feedback for A. A is not required to report positive feedback for B. However, if A is dissatisfied with the transaction operation, it will explicitly report a negative feedback for B. Such a negative feedback will automatically cancel the prior positive feedback that B had gained by encashing A's token and instead add to its negative reputation score.

In this whole process, Vito charges a single transaction fee — from the trader (B). This charge is made when the seller encashes the customer's token and improves its positive reputation score.

On data integrity, confidentiality, and their complexity: In general data integrity and security in a MoB environment is no worse than in any other wireless environment that lacks any security mechanisms. Therefore, a user who wants additional data confidentiality and integrity will have to employ appropriate security mechanisms.

However, we require appropriate security mechanisms for the reputation certificates and service tokens that are exchanged as part of a trade. We use public key cryptography, e.g., 3DES or RC4, and message digests, e.g., MD5 or SHA-1, to generate digital signatures of reputation information. For example, the reputation certificate of a user includes the reputation information in plaintext followed by an encryption of the message digest of the same information (message digests are regularly employed to speed up the signature generation and verification operations). Such an approach ensures integrity of reputation data, but not confidentiality.

Figure 3 indicates that in each trade, the customer performs two private key encryptions on message digests and one public key verification of a message digest. Similarly, the trader performs two public key verifications on message digests and one private key encryption of a message digest. Prior work by Freeman et.al. [9] from 1999 had studied the time complexity of such operations on multiple low-end platforms. For a modulus length of 512 bits, their results show that the signature generation and verification operations take 40 ms and 2.5 ms respectively on a Pentium II 266 MHz machine running Linux and take 25 ms and 2.3 ms respectively on a MVME-2600 333 MHz board running VxWorks (a real-time OS). Based on these results we believe that these mechanisms to prevent tampering of reputation information can, therefore, be executed even on low-end devices such as cell-phones and PDAs. We comment on various other practical implications of such mechanisms in Section 5.

Trader uploads its own positive reputation feedback: A positive reputation feedback for the trader (B) benefits itself in future trades. Hence we make the beneficiary responsible for performing the reputation feedback upload.

Trader uploads positive feedback for customer: The positive feedback for the customer (A) is contingent upon successful encashing of the token. In Vito, the service token is assumed to be a signed certificate from A which indicates the trade price. On receiving this token, Vito's billing service will verify that A has adequate credit in the system and appropriately informs B. Based on this response from Vito, B will choose to update either positive or negative feedback for A. Studies have shown that self-interest, specifically the expectation of a reciprocal positive rating from one's trading partner is the strong motivation behind high levels of voluntary feedback in such a system [8].

Customer uploads negative feedback for the trader: This was a natural decision since the trader has no incentive to report its own failure and reduce its positive reputation in the system. Note that in our proposed system, the trader has no recourse if a malicious customer always chooses to provide negative reputation feedback. This is a shortcoming that is present in a successful management system like eBay. The common assumption is that users in the system are selfish, but not malicious — they do not choose to maliciously reduce a trader's positive reputation when they received good service from them.

Customer pays prior to receiving service: We had a choice of requiring the customer to send the signed payment either prior to or after the service transaction. If the customer makes the payment after the service, there is a danger that a malicious customer can default the payment. In such a scenario, the trader has no proof of the transaction and has no further recourse. However, if the customer pays first, and the trader defaults in providing the service then the trader will be caught defaulting in case it attempts to encash the service token. The customer has the minimal recourse of providing negative reputation feedback in response corresponding to the encashed token.

Transaction fee charge: A transaction fee is the incentive for the reputation system. Additionally, having a transaction fee implies that no one can build up reputation for free (and hence misuse the system by constructing multiple colluding identities, performing transactions between these identities, and report positive feedback).

Further design considerations for Vito

Vito, as described, is based on eBay's reputation management model. We derive the feasibility of Vito based on eBay's success in managing more than 4 million simultaneous auctions, with fairly low rate of misuse. Resnick et.al. [30] explain why such reputation systems work in practice. In general there are three properties needed at minimum to make reputation systems work: (1) entities must be long-lived, so that there is an expectation of future interaction with other such entities, (2) feedback about current interactions is captured and distributed so that such information is visible in the future, and (3) past feedback guides buyer decisions.

However, as various studies have pointed out, reputation systems are not infallible and further work is needed to improve their resilience to malicious behavior. We describe a few such challenges to make reputation systems really robust. They include: (1) *Sybil attacks:* A user with bad reputation re-entering the system with a new identity. A consequence of such behavior might be that newcomers (with little or no reputation state) are always distrusted unless they have "somehow paid their dues, either through an entry fee or by accepting more risks or worse prices while building up a reputation" [30]. Another alternative is to prevent name changes either by requiring the use of real names or by preventing people from acquiring multiple pseudonyms, a technique called once-in-a-lifetime pseudonym [10]. (2) *Collusions:* A group of users collaborate and rate each other positively to accumulate positive feedback, artificially inflating their reputations — user collusion. Prior research has tried to address the collusion problem in various reputation systems. For example, it is possible to view Google's Page Rank algorithm as a reputation system in which a set of colluding web-pages try to artificially increase their page rank by carefully choosing their outgoing web-links. In [39] the authors illustrate that the problem of making such "eigenvector" based methods robust to collusions is NP-Hard, and propose some heuristic approaches. Authors in [14] examine a similar problem in the P2P scenario and demonstrate how collusions can be avoided if there exists a set of pre-trusted peers. While both these approaches are promising, we believe that using transaction fees for reputation-reporting adds a new mechanism that can be exploited to prevent collusion. We intend to consider all of these approaches in our future work for a theoretically robust reputation system. (3) *Decentralized reputation management:* Our current proposal for reputation management is centralized. Given the periodic nature of interactions between users and the reputation repository, such centralization is likely to be adequate. However, with growing popularity of the system, it is possible that load on a single centralized reputation management system maybe too high and the task of reputation management may need to be divided across multiple such repositories. Additionally, in many scenarios, it can be useful to define decentralized reputation management approaches. Some approaches to perform such decentralized reputation management has been proposed in recent literature, mostly in the context of P2P networks that exploit pre-trusted peers [14].

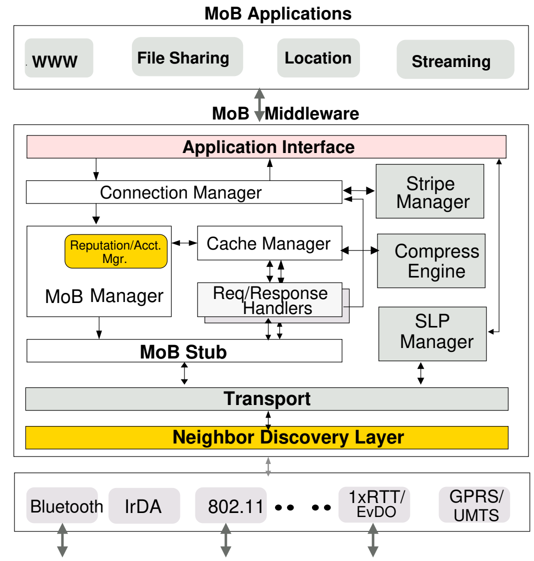


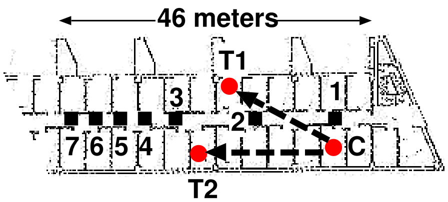
Figure 4: Overview of middleware implementation in a MoB device.

3. IMPLEMENTATION

We have implemented the MoB system over a Linux based platform (we have also ported part of the MoB system in Windows XP using C#). Our implementation of different MoB clients include single as well as multiple wireless interfaces with local (e.g. Bluetooth) and global wireless (e.g. 3G) connectivity. The entire implementation is available as a 'middleware' that is installed in each MoB-enabled device (MoB device, for short). Figure 4 shows the overview of this middleware implementation in each MoB device.

The *connection manager* accepts connections from MoB applications and passes them to the *MoB manager* (which includes the reputation management functionalities and coordinates all activities in the middleware). The MoB manager checks the local *cache manager* for the object being sought by the application. In case of a cache miss, the cache manager invokes a request/response handler leading to a connection setup with a neighboring MoB device. Unique resource locators are assigned to each such request, and are used to map them to corresponding response handlers. The response handler also interacts with the cache manager to update cache state as necessary. On arrival of the response, the data object is then made available to the pending connection.

At the receiving MoB device, a response handler examines the request and takes appropriate action — searching its local cache manager, and if necessary initiates a request to other neighboring MoB devices. If this MoB device has a wide-area access interface, e.g., a 3G-1X/EvDO PC card, depending on the query-type, it may



Location	Latency (ms)	Loss(%)
1	43.4	0
2	49.5	0
3	56.4	0
4	67.6	1
5	78.0	2
6	81.1	2
7	-	-

Figure 5: Floor-plan of building for MoB experiment scenarios. C is customer, T_1, T_2 are traders (positions changed in different experiments as described in this section). The table indicates the latency and loss characteristics obtained in this environment using a source at C and destinations varying from $1 - 7$ using Bluetooth.

initiate a data retrieval process from the Internet through this interface. Note that the MoB architecture makes these application-level data-retrieval hops completely transparent to the initiating device.

The stripe manager regulates block-based application-level data striping of large data objects from neighboring MoB devices. Block-based data retrieval enhances data download performance in environments characterized by high degree of user churn. This module intelligently partitions data objects into multiple small blocks and downloads each of these independent data blocks from in-range MoB devices. To adapt to the variable degrees of user churn, the stripe manager dynamically changes the block size, the number of parallel TCP connections opened, and connection types (e.g. persistent/non-persistent connections) during a download. It also efficiently load balances data traffic across multiple neighboring devices as necessary.

MoB implements a content processing engine, which performs various optional functions including data compression, traffic filtering, etc. Images are downgraded and fixed fidelity data (text) is compressed if needed. The compressing functionality is used to reduce volume of data transferred over wireless links thereby speeding up content distribution in MoB environments. It also offers devices to adapt to low-bandwidth links, e.g., by reducing fidelity of downloaded images. The content filtering functionality lets a MoB device define rules by which it may eliminate (unsolicited and malicious) traffic passing through it.

Neighbor Discovery Layer: Two neighboring MoB devices find each other through periodic scanning using link-specific mechanisms as provided by different wireless interfaces. For example, for 802.11 interfaces we set aside one specific channel for neighbor discovery (and service announcements as well). In its quiescent state, each MoB device goes into a promiscuous mode, monitoring all traffic on this channel. MoB service announcements (and responses) are transmitted on this channel with the SSID set to *MoB* and mode set to *ad-hoc*. As part of this initial discovery, the two participating devices also decide to switch to a specific other 802.11 channel for the actual service interaction.

Using a Bluetooth interface, the device initiates a *scan* procedure to detect other MoB devices in-range. This results in an active set of devices that a MoB device can query for services. The MoB device then connects and dials up to its neighbor device using DUN (dial-up networking) service if needed. Using DUN the MoB device establishes an IP connection using (Point-to-Point) PPP connection over a serial RFCOMM channel. (RFCOMM protocol provides emulation of serial ports over Bluetooth's link layer protocol (called L2CAP)).

Applications running on MoB devices use this middleware with a well-defined interface (as shown in the figure) for all interactions with other MoB devices. Each trader device can implement a set of application-layer services using this middleware and advertises them to interested customers.

4. EVALUATING MOB APPLICATIONS

We have implemented a prototype MoB system along with multiple collaborative applications. In this section we report on our experiences (both based on the implementation as well as simulations) with a subset of these applications: (1) file-transfer applications (including data transfer from a specific location and data location and retrieval operation), (2) web browsing, (3) bandwidth aggregation based media streaming, and (4) location determination. Although file-transfer applications and web browsing applications both use TCP-based transfer, the applications primarily differ in the way the transferred data is organized and located across multiple servers.

Note that this evaluation work provides a snapshot of the range of applications we have implemented using MoB.

Experimental setup

For each of the applications mentioned above, the experiments were conducted using (marginally) different setups as were necessary. All experiments were conducted indoors. The floor-plan of our experimental environment is shown in Figure 5. Communication between customers and traders used different wireless technologies — Bluetooth, 802.11a, and 802.11b. In some experiments traders had 3G wide-area cellular interfaces. We used two different 3G technologies: (1) 3G EvDO data service with a maximum ideal downlink data-rate of 2.4 Mbps and an uplink data-rate of 153 Kbps, and (2) 3G 1xRTT data service offers a maximum ideal downlink data-rate of 144 Kbps and an uplink data-rate of 64 Kbps.

4.1 File-transfer Applications

We consider two classes of file-transfer applications — data transfers from a specific location and data location and retrieval.

Data transfer from an Internet location

We first consider the scenario where a user performs a *ftp*-like transfer of a (large) file between his device and a specific location in the Internet. In this application, the customer finds an in-range bandwidth trader and initiates the file download by requesting a sequence of moderate sized blocks. As each block transfer is about to finish, the customer makes a request for the next block. If a new bandwidth trader is available prior to the entire download process terminating, the customer will simultaneously use the new trader to download the file data.

We now present a scenario purposefully constructed to illustrate how the MoB implementation adapts to user mobility.

Scenario 1 (Figure 6): There is a single customer C, using a Bluetooth wireless interface, and two mobile traders, T_1 (with a CDMA 1xRTT interface) and T_2 (with a CDMA EvDO interface). At the initial instant, only T_1 is in range of C. Hence, C requests a file

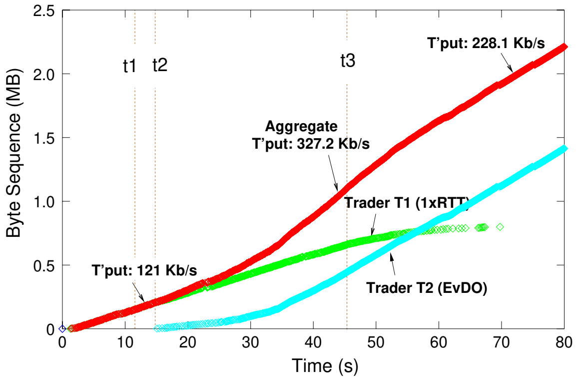


Figure 6: A file download for a MoB device using two traders with different wide-area interfaces (Scenario 1). The two traders are used simultaneously for the download over the period of time when they both are in-range of the customer.

download through T_1 at time 0. A second trader T_2 moves into range of C such that there is a period of time when both the traders are simultaneously available to the customer for data downloads. As shown in the corresponding time sequence plot (Figure 6) trader T_2 moves in-range of C around $t_1 = 12$ seconds and is detected by C around $t_2 = 14$ seconds, while trader T_1 starts to move out-of-range of C around $t_3 = 45$ seconds but continues to stay connected until time 70 seconds. Between t_2 and t_3 the C uses both traders to simultaneously stripe individual blocks of the file to achieve a high aggregate throughput of 327.2 Kbps. Once T_1 goes completely out-of-range, C continues to download the remaining file using trader T_2 alone. The total transfer in this scenario takes about 79 seconds (i.e., a throughput of 227.8 Kbps).

Data location and retrieval

We next examine the performance of Gnutella and Kazaa-style peer-to-peer data search and retrieval applications. As in the file download from a specific location application, we show the impact of user mobility (churn) on performance through three different scenarios. In this application we study three different levels of user churn: (1) High churn, when each potential trader stay in-range of a customer for 10 to 20 seconds; (2) Medium churn, when each such trader stays in-range of a customer for 40 to 60 seconds; (3) low (or minimal) churn, when each potential trader stays in-range for 60 to 120 seconds; and (4) no churn, where there is no trader mobility and serves as the base case. We show results for the high churn rate case only for Scenario 5 (where high data rates available made it feasible). Note that in typical coffee-shop scenarios, we expect user behavior to follow the low churn rates or no churn (see Figure 2).

Scenario 2 (Figure 7): A customer locates a trader device in its vicinity that has the queried data object. On locating such a trader, the customer starts block-based data download of the object. If due to any reason the trader goes out-of-range, the customer attempts to find another trader and resumes the same download. In this scenario we assume that a customer at any given time uses only one trader for a given object retrieval (say, its a user-imposed policy limit on the device). Only when the current trader device moves out-of-range does the customer search for an alternate trader. In this scenario, the interaction between the customer and trader occurs using their Bluetooth interface.

In Figure 7 we show the impact of such user churn on download performance for a 5 MB (audio MP3) file as the block size parameter is varied. In the no churn case, a block sizes in the range of 10-50 KB is necessary to keep the pipeline of the Bluetooth channel properly utilized (download takes ~ 212 seconds). Beyond 50 KB block sizes performance does not increase substantially. However, as churn in the system increases, there is an optimal block size for such data download operation. This is because as the block size increases beyond this optimal, the trader often moves out-of-range prior to the successful transfer of the block and the effort spent in the partial download of this block is wasted. We can see this in the low and medium churn scenarios. For example, in the medium churn scenario, the optimal block size is between 20-50 KB and transfer attempts with block size in excess of 200 KB did not finish.

Why not download the entire file instead of using a sequence of blocks? The block size limit for file transfers in MoB is important for two different reasons. First, it allows a customer to bound the amount of outstanding data request in MoB transactions. Note that in a MoB transaction, the customer makes a payment prior to the service (due to reasons explained in Section 2). Consider the case where the customer requests the entire data all-at-once. In this case the customer makes the entire payment prior to receiving the data. If for some reason the data transfer is incomplete (say, the trader moved out-of-range) then the liability of the customer is high. Breaking the data request into multiple smaller units therefore helps in bounding this liability. Second, it opens up the possibility of efficient data download for the customer by requesting independent blocks from multiple traders. The availability and number of such traders may not be known in advance, and using smaller block size allows for greater parallel downloads. Additionally, in such scenarios it helps in managing the data range downloads better.

Scenario 3 (Figure 8): This scenario is the same as Scenario 3 except that each customer is allowed to use at most two traders for the download of an object at any given time. If one of the traders move out-of-range, it may be replaced by an alternate trader. This scenario therefore, depicts impact of parallel downloads in a MoB environment. In this scenario, the customer uses its *single* Bluetooth interface to connect to the two traders (both equipped with corresponding Bluetooth interfaces).

We present the results of this scenario in Figure 8. It is interesting to note that in this scenario, the download performance in the no churn case is marginally worse than in Scenario 3 (which uses only one trader at a given time). Even though we are using two traders in Scenario 3, the customer is sharing its single Bluetooth interface between them and hence there is no effective gain in download performance. In fact the performance goes down slightly (download time is 239 seconds) because of switching overheads between the two parallel transfers using the same interface.

However, as churn in the system increases, the improvement in download performance over Scenario 3 is apparent. This occurs because in Scenario 3 there are multiple periods of disconnections and "dead-time" (when one trader goes out-of-range and another one needs to be phased in). In contrast, with the two trader scenario, there is very little "dead-time" which can only happen if both the current traders move out of range within a short time period and there is no replacement.

Scenario 4 (Figure 9): This scenario is the same as Scenario 3, i.e., at most two traders for a single download at any time, except that the customer here is using two different interfaces — one Bluetooth

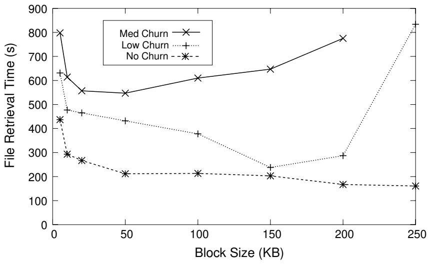


Figure 7: The impact of user churn and block size variations on peer-to-peer object download (Scenario 2: only one trader allowed per customer). Interaction using single Bluetooth interface.

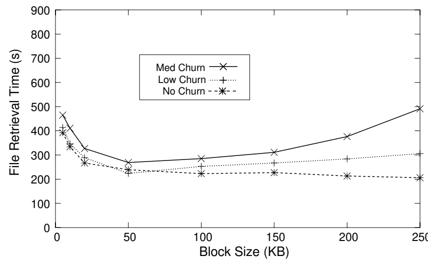


Figure 8: The impact of user churn and block size variations on peer-to-peer object download (Scenario 3: up to two traders allowed per customer). Interaction using single Bluetooth interface.

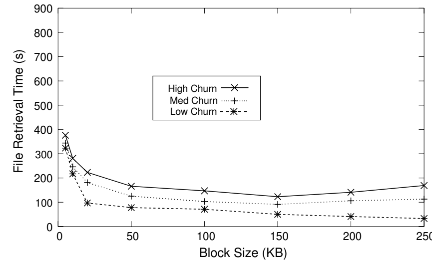


Figure 9: The impact of user churn and block size variations on peer-to-peer object download (Scenario 4: up to two traders allowed per customer). Interaction using Bluetooth and 802.11a.

(2.4 GHz) and one 802.11a (5 GHz). Note that the data rates of 802.11a interfaces are much higher than Bluetooth and hence it was feasible to use high churn rates in this case. It is quite clear that using the two interfaces (operating in non-interfering parts of the spectrum) simultaneously leads to significant performance benefits in download times (even in high churn case download latency with 250 KB block size is 169 seconds). Note that these two interfaces operate in different ranges of the wireless spectrum and hence do not interfere with each other.

4.2 Web browsing Applications

It is possible to implement more efficient web browsing applications in MoB for devices that otherwise have low-bandwidth connectivity to the Internet. The main advantage for the corresponding users in MoB is the ability to use collaborative caching. A set of MoB devices with a cache of web content can respond to the requests made for these objects by customer devices in their physical neighborhood. If substantial amount of web object requests can be served by neighboring MoB devices, it can help significantly lower the response time perceived by the user.

We evaluate the potential benefits of distributed web object caching in a MoB environment. We explain using the following scenario.

Scenario 5 (Figure 10): A MoB customer is in range of two MoB traders and can communicate with them using Bluetooth. The two MoB traders have Internet connectivity through a 3G 1xRTT and a 3G EvDO interface respectively. Each of the MoB traders can also serve as a cache for the customer. The customer recruits both these traders to serve as wide-area wireless interfaces and web caches. We consider the two extreme cases — (1) none of the web objects are cached in either of the two traders (cache miss), and hence the objects need to be downloaded across the wide-area interfaces; (2) all static and cacheable objects of various websites are cached in the two MoB traders (cache hit), and only the dynamic content of websites need to be downloaded across the wide-area wireless interfaces. In the latter case we also consider the impact of high user churn (trader stays connected for between 10 and 20 seconds).

We consider five websites with significant diversity in their content characteristics. They include (a) a news website (washingtonpost.com) with 61 objects and over 250KB data, (2) an e-auction website (ebay.com) with 53 object and over 153 KB data, an open source website (sourceforge.org) with 41 objects and over 122 KB data, a technical news website (slashdot.org) with 26 objects and 128 KB data, and a university website (wisc.edu) with 46 objects for 67 KB data.

The figure shows the complete download time for each of these webpages. For example, for washingtonpost.com, using two traders instead of one leads to an improvement of 33% for cache miss cases. When compared to two trader cache miss case, the cache hit case improves performance by further 45%. In fact, analysis of trace logs in MoB devices indicates that a significant percentage (e.g. about 77% for the wisc.edu website) of the web content was static and cacheable and hence served locally by the MoB trader caches. This means that MoB not only improves user response time during web downloads, but also reduces costs and the traffic on the wide-area interfaces.

4.3 Collaborative location determination Application

Location information can be a key enabler of various mobile applications. Indeed, numerous applications can be well customized to users' needs based on their location context. Navigational services are the prototypical example of location-sensitive applications. Numerous Global Positioning System (GPS) devices are currently available commercial worldwide with positioning accuracy varying between 10 meters to less than 3 meters.

In this section we examine how MoB can help users without such GPS access obtain location information from other in-range users with GPS access.

To simulate the efficacy of trading location information in MoB we constructed a simulation scenario as follows. We considered a specific urban area — 4 km by 3.2 km of mid-town Manhattan, NY, USA, where numerous automobiles ply continuously on the streets. (This zone corresponds to roughly a 50 block width across across mid-town Manhattan.) A vast majority of thoroughfares in this region of Manhattan is organized in a relatively regular grid structure. We varied the average number of vehicles in this zone between 1000 and 5000 for different simulation experiments (while we could not gather exact statistics on vehicular density, we are fairly confident that these numbers are on the lower end of the vehicular density of Manhattan and typical urban areas of the world).

Each vehicle in this simulation traveled along the city thoroughfares using a "Manhattan random waypoint" model (inspired by the random waypoint model [13]). We define this as follows: Each vehicle chooses a destination location uniformly at random and travels towards it at a constant speed along city streets (which were mostly north-south and east-west). The speed of motion was also chosen at random from a range with a minimum and maximum speed. Speed choices were consistently biased towards higher speeds to avoid the mobility problem identified in [37].

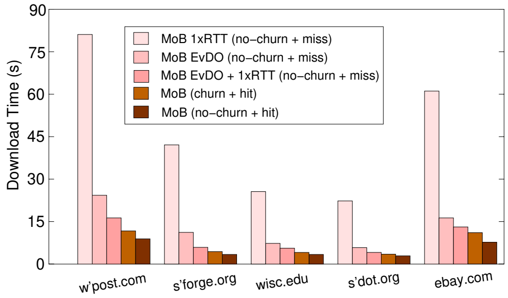


Figure 10: Web downloads using collaborative caching (Scenario 5).

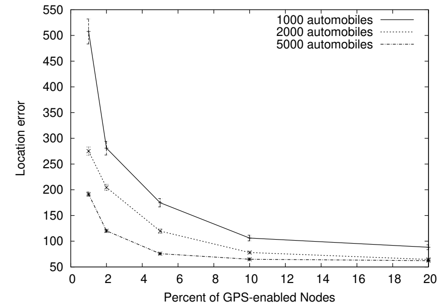


Figure 11: Location error (in m) for the Manhattan scenario Three different automobile densities are considered within the same 4 km by 3.2 km region.

We assume that a fraction of the vehicles are equipped with GPS devices. Each such GPS-equipped vehicle is willing to sell its location information to another in-range vehicle, which lacks GPS. In these experiments we assume that two vehicles are in range if and only if they are in communication range of their omni-directional 802.11b radios with a maximum transmission power of 30 mW. This translates to roughly a 80 m communication range in outdoor environments based on our experiments.We assume that each GPS-equipped vehicle continuously has perfect knowledge of its own location. In each location information trade, a GPS-unaware vehicle, A, obtains this information from another vehicle, B, that is aware of its own location. If B is equipped with a GPS device, that A calculates its own location as the location available from B. This information, therefore, is inherently error-prone; the error being exactly equal to the distance between A and B at the time of the trade. If B is not equipped with a GPS device, it is still possible that B's location information (obtained through a prior trade) is still fairly accurate. In such a case, we assume that A performs the location trade if and only if B's location information was last updated in the recent past within a configurable threshold period. In this scenario A calculates its location as the average of its last location update and the new location information being obtained from B. Note that A updates its location only when it encounters another vehicle that is either GPS-equipped or has some fresh information about its own location.

Therefore there are three sources of location error in this simulation environment for a non-GPS vehicle. First, the information obtained from a GPS-equipped vehicle has an inherent error equal to the distance between the two vehicles at the time of the trade. Second, it does not use any intelligent techniques to update its own location between successive acquisitions of this information. Third, location information obtained from other non-GPS vehicles is also inaccurate.

Inspite of these drawbacks we find that the location accuracy of this system of location dissemination is surprisingly good. In Figure 11 we plot the average location error experienced by different non-GPS vehicles as the fraction of GPS-equipped vehicles increase for different vehicle density. Clearly the location accuracy increases with increase in the fraction of GPS-equipped vehicles. Additionally, as we can expect, the location accuracy increases with increasing automobile density. Hence It is interesting to note that even when the fraction of GPS-equipped vehicles is only 5%, the location accuracy of our simplistic location dissemination service in MoB leads to an average accuracy of 72 meters (or less than a street block) for the 5000 automobile scenario.

Techniques to improve location accuracy: Our simple location dissemination approach enabled by MoB can achieve an significant accuracy (72 meters) for a fairly conservative density of 5000 automobiles in a 12.8 square kilometers area of the city. Clearly the performance of the system is expected to be even better for more realistic automobile densities. Additionally, it is possible to employ multiple other techniques to further improve the quality of location information. For example, it is possible to effectively exploit information like the speed of a vehicle's motion and its possible direction to continually update the location information. Additionally, there are typically a large number of wireless Access Points distributed in the city, e.g., coffee-shops, public hotspots, that can also serve as location "anchors" and enhance the location information available at non-GPS vehicles. It is possible to design more intelligent algorithms based on these anchors, e.g., triangulation-based approaches (Radar [2], Horus [38]) to significantly enhance the quality of location for non-GPS nodes.

4.4 An evaluation of Vito

The reputation and trust management system plays a central role for all users requiring its support to induce trust. While it is possible to use different reputation systems in the MoB architecture, in this section we focus on Vito and examine how it can be used in MoB. Our evaluation is done with detailed simulations involving a pool of that are constantly looking for services and are themselves capable of providing services. Not every service is available at every user's device. Users perform transactions amongst each other using the sequence of operations described in Section 2. As described in that section, positive and negative scores on various trades are uploaded to Vito when the corresponding users gain Internet access. At such a time, Vito recalculates and issues a new reputation certificate for the user.

We consider three different Internet access models as follows: (1) *Continuous connectivity (and uploads and downloads of certificates):* models the scenario where a wide area Internet connection is available continuously. Therefore, the service token for each user can be validated by Vito immediately after the payment is made to the trader. Similarly, the reputation score for each user can also be immediately updated by Vito. (2) *Uniformly at random:* In many scenarios, users may not have continuous connectivity. As a first approximation, we model users connecting to the Internet uniformly at random at different times, say 5 different hours in a 24-hour period. (3) *End-of-day connectivity:* A more realistic model of Internet connectivity maybe that a majority of users have guaranteed Internet access only at the end of the day. We use a 80-20 model where each user has a 80% chance of connecting to

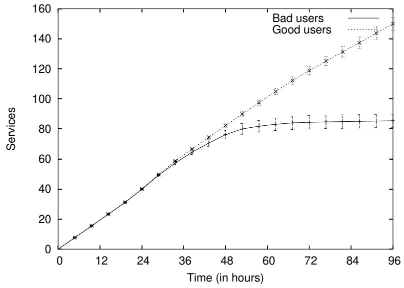


Figure 12: A time evolution of services obtained by good and bad users in MoB based on Vito's design policies.

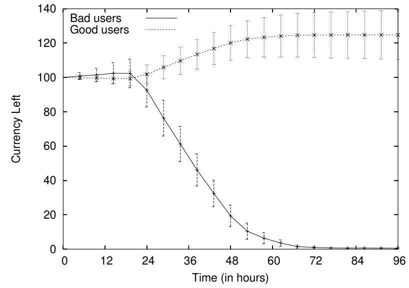


Figure 13: A time evolution of currency left at good and bad users in MoB based on Vito's design policies.

Malicious	Internet access model		
percentage	Continuous	Unif-at-Rand.	End-of-day
1	22.6	42.2	51.8
5	24.0	38.4	43.7
10	25.4	40.3	48.5
20	24.5	39.4	47.0
40	23.0	42.2	48.5

Figure 14: Number of hours at which currency left at malicious users decreases below a 'low' threshold.

the Internet in the last 20% of the day (roughly between 7pm and midnight). Each user requests up to 50 services within a 24 hour period.

Each user typically has multiple other in-range users who responds to a service query. In this evaluation we assume that the user chooses only one of these respondents for its service. This raises the issue of user selection policies — given that there are multiple potential traders in-range, who should a customer pick to obtain services from. We consider five different selection policies in our evaluation as follows:

- *Reputation:* Choose the user with the best reputation, irrespective of advertised price of the service. Such a choice is biased in favor of transaction reliability instead of its price.
- *Price:* Choose the user who is offering the lowest price regardless of his reputation.
- *Price with reputation threshold:* Choose the user with the least price as long as the reputation of the user is greater than a customer-configured threshold. Such a choice implies that the customer prefers a user who offers a low price but is not willing to compromise too much on reputation.
- *Ratio of reputation to price:* Choose the user with the highest value of this ratio. Such a choice integrates most of the the positive aspects from the previous policies since it implies that the user is willing to pay a higher price to a high-reputation trader and a lower price to a lower-reputation trader.

Since each customer is free to practice any of these selection policies, each trader in these simulation experiments need to be smart enough to make himself attractive to the choices of customers. Therefore we introduce two parameters, a price reduction factor, γ, and a price increment factor, λ. When a trader advertises his service in response to a service request and does not get chosen, he lowers the price, p, by the factor γp for the next service offering. Similarly, once a user's service gets chosen by a customer he increases the price by the factor λ in the next service offering.

In general, individual traders can choose these parameters on how they perceive their own reputation vis a vis their offered price. In fact, since MoB does not dictate price choices such for any one, in a real deployment each trader can be more aggressive or conservative in their price alterations as they choose to be. In our simulations, however, we use the values $\lambda = \gamma = 0.01$.

Results

In our simulations, we have experimented with different choices of user policies, Internet access models, sizes of user population, and fraction of malicious users. Due to space constraints we highlight some of the interesting aspects of these results performed with 500 users. In the simulations we choose a fraction of users (traders) to be *malicious*. A malicious trader is one that receives payment for a service from a customer but does not provide the corresponding service. Such behavior will lead to accumulation of negative feedback for the malicious trader.

We initialize all users with a starting currency amount of 100 units. Users use this currency to perform transactions with other users. As a consequence of each transaction, the trader gains currency from the customer. The total amount of the currency in the system is conserved. Each service was initially priced uniformly at random between 1 and 5 currency units and were adjusted by the traders based on their trading history using the γ and λ parameters.

Selection policies: In the first set of experiments we consider different user selection policies keeping the fraction of malicious users fixed at 10%, while using the end-of-day Internet access model.

Since MoB is an open-architecture system, customers are free to choose traders based on their own selection policies. We first compare the performance of the set of selection policies in a set of experiments where 10% of the users are malicious (Table 1). Each column in the table corresponds to a scenario where all users chose a trader using the corresponding selection policy. The table shows the total number of services obtained and the total currency left at the good and malicious users at the end of a 20-day period (on each day a user attempted up to 50 transactions). The maximum number of transactions by a user can be 1000.

All schemes that used reputation as a determining factor resulted in exhaustion of currency at the malicious users, e.g., in the Reputation scheme the currency left at a malicious user was 0.01. As the reputation of malicious users decreased, customers stopped using them for transactions. Hence malicious users continued to lose their currency in acquiring services (all users have to pay first to obtain services) and were not used as traders and hence, never gained additional currency. Because of this reason, the total number of successful services gained by malicious users were also limited. The Price policy was the clear exception among all the policies. Since this policy did not distinguish between good and malicious users, the latter were never penalized for their behavior. Finally it is easy to observe that the Ratio of reputation to price policy led to the best overall performance in our simulated scenarios. Therefore in the subsequent experiments we use this trader selection policy only.

Good users and Malicious users: To study the efficacy of reputation management in Vito using the Ratio of reputation to price

User policy	Reputation	Price	Price with reputation threshold threshold = 10	threshold = 20	Reputation/price
Services obtained by					
Good users	527.66	787.7	362.0	323.3	858.9
Malicious users	72.9	789.0	151.6	111.7	81.7
Currency left at					
Good users	124.7	97.9	124.7	124.7	124.4
Malicious users	$\sim$ 0.0	106.6	0.7	0.6	0.7

Table 1: Impact of different user policies in choosing MoB traders for services.

Malicious percentage	Services received by Good users	Malicious users
1	868.2 (8.0)	99.2 (28.7)
5	870.0 (8.1)	89.4 (7.0)
10	869.0 (8.4)	94.1 (7.0)
20	858.9 (9.3)	81.7 (4.5)
40	803.7 (12.0)	86.5 (1.8

Table 2: Impact of varying fraction of malicious users (number in parenthesis is the standard deviation).

policy, we take a closer look at the time evolution of services acquired and currency left over time for good and malicious users. We can see that within about 48 hours, the malicious users deplete their currency significantly enough (Figure 13) that they are not able to acquire many further services (Figure 12). In contrast, the total currency at good users stay fairly steady (Figure 13) , and they are able to consistently acquire more services as time progresses (Figure 12).

Effects of varying malicious users: In Table 2 we present the effect of varying the fraction of malicious users in the system. The table shows the total number of services acquired by each user at the end of a 20 day period as the fraction of malicious users increased from 1% to 40%. As expected, the performance of good and malicious users is not sensitive to the fraction of malicious users. In all cases, the reputation of all the malicious users decrease and are equally avoided by any user in conducting transactions.

Internet connection models: Finally we examine the impact of different Internet access models. In all the previous experiments we assumed the end-of-day model and in this experiment we consider the three different models discussed in the previous section. Like in most of the prior experiments we consider a set of 500 users (of which 20% are malicious) each using the ratio of reputation to price selection policy. In Figure 14 we present the time when the currency left at malicious users fall below the 20 unit mark, and never recover. We chose the 20 unit mark, because that was approximately when the plots in Figures 12 and 13 stabilized with no further significant change in performance for good and malicious users. As expected, the continuous connectivity model converges the fastest followed by uniformly-at-random with end-of-day connectivity model taking the most time (51.8 hours for 1% malicious users).

Summary: Among the various user policies explored in this section, we have demonstrated that the Ratio of reputation to price selection policy defines a good way for a customer to choose a trader. This policy is relatively insensitive to the number of malicious users in the system and quickly enables customers to distinguish between good and malicious traders.

However, our evaluation of Vito is by no means exhaustive. Vito only provides a reputation management system and it is the users who decide how they view various reputation scores of various other users. Therefore it is quite possible that some users define adaptive learning techniques that predict the chances of non-malicious behavior in a given transaction, given their prior reputation score.

Additionally, it is possible many users of MoB completely sidetrack the reputation management of Vito and only interact with other users they directly trust, e.g., their friends alone. It is also possible that some users rely on a de-centralized "web of trust model," e.g., as used by PGP (see http://www.pgpi.org), to make similar choices of interactions. We will examine some of these approaches in our future work.

5. DISCUSSION

We believe that our proposed MoB architecture is an important *first step* towards enabling fine-grained competition, diversity, and flexible service composition in wide-area wireless environments. While our work addresses a number of important issues required to realize this architecture, we believe that further theoretical studies and evaluation by deployment needs to be performed in various large-scale scenarios. We discuss some potential challenges that arise in the context of security and legal aspects next.

Security: A continuous concern in any open, collaborative environment is that of security of services. As discussed in Section 2, the proposed MoB architecture provides integrity of reputation certificates that would allow MoB clients to interact in appropriate transactions. However, MoB does not explicitly address data security and integrity issues. We believe that such security issues need to be addressed end-to-end. For example, a MoB customer who is downloading sensitive data from a host in the wired Internet through one or more bandwidth traders should use Secure Sockets Layer (SSL) between the two ends for all data security and integrity needs.

In some other application scenarios, providing data security may not be as straightforward and would require application support. For example, in the distributed location determination application a customer purchases location information from a sequence of traders on its path. A few of these traders can be malicious and provide incorrect location information to the unsuspecting client. Clearly, the location information from such malicious users will be inconsistent with the information from the rest of the traders. The customer's application can detect such mismatch and in turn lead to a negative reputation feedback for the malicious traders.

Finally the amount of security functionality implemented by individual clients reflect their prior experience in the system and the extent of their faith in behavior of others. This is precisely what the reputation system aims to address — manage the history of each user's interactions with other users. A very liberal user may choose to accept data from any neighboring trader, a more discerning user will pick a trader with with "reasonable"' reputation, while a very conservative neighbor can choose to obtain such services from traders they directly trust.

Legal aspects: Many services traded in MoB occur in a peer-to-peer fashion involving only two entities, e.g., the collaborative location determination service. However, there are many other applications in which a client, C, acts as a reseller — it may buy service from X and sell it to Y, in effect acting as a service conduit.

Schemes	Network layer forwarding	Application-layer services: Data/object retrieval	Application-layer services: Web caching	Application-layer services: Other apps.	Extra infrastructure support needed	Software only solution	Service verification	Incentive based
MCN [22] and Aggelou et.al. [1]	✓	×	×	-	Base station support	✓	∞	×
7DS [26]	×	✓	✓	-	-	✓	∞	×
CAPS [19]	×	✓	✓	-	optional cellular network support	✓	∞	×
UCAN [23]	✓	×	×	-	Base station support	✓	Crypto + Base stn. support	✓
ORION [16]	×	✓	×	-	-	✓	∞	×
iCAR [35] and Bejerano [3]	✓	×	×	-	Relay + cellular network support	✓	∞	✓
Sprite [40]	✓	×	×	-	N/A	✓	Cryptographic (on data path)	✓
Watchdog [24] and CONFIDANT [4]	✓	×	×	-	N/A	✓	Reputation	×
Nuggets/Nuglets [5, 6]	✓	×	×	-	N/A	×	Cryptographic (on data path)	✓
Ben Salem et.al. [32]	✓	×	×	-	Base station support	✓	Cryptographic (on data path)	✓
MoB	✓	✓	✓	b/w aggregation cooperative location traffic filtering, etc.	Vito	✓	Reputation	✓

Table 3: Comparison of MoB architecture with prior work. ∞ represents not defined.

In some cases there may be legal issues that prevent such re-sale of services. For example, many wireless ISPs, cellular data networks, and other infrastructure providers today may require their customers to never re-sell bandwidth services to other parties, primarily because the infrastructure providers have no incentive to carry such third-party traffic when they are not making any revenue. Therefore it may be necessary to provide incentive-sharing techniques between participants in MoB.

In our above example, consider that C is a cell-phone device connected to X, which is a 3G cellular data network provider. C acts as a relay to provide bandwidth services to a laptop, Y. Then X may require a share in the profit that C makes from Y. In such scenarios the terms and agreements between C and X should need to be appropriately updated. How such agreements are formed and are in the realm of user agreement and policies and hence we leave them outside the scope of this paper.

6. RELATED WORK

A number of interesting prior projects have examined various forms of collaborations between mobile devices, in the context of infrastructure-based wireless networks as well as mobile ad-hoc networks. The key difference of MoB from all such prior work is that we propose an architecture to implement a wide-range of application-layer services that are facilitated by a third-party Internet service for managing peer-to-peer incentives. In this section we summarize some of the prior work and illustrate the differences from MoB (see Table 3).

In 7DS [26], Papadopouli et.al. present a peer-to-peer data sharing system for exchange of information among peers that are not necessarily connected to the Internet. It is an application layer protocol and thus can be deployed without any changes to the underlying architecture. Unlike MoB, 7DS assumes cooperation between mobile peers. ORION [16] is another such peer-to-peer file sharing system that combines application-layer query processing with the network layer process of route discovery to reduce control overhead. MAR [31] and Handheld Routers [34] define a wireless router that exploits available diversity in wireless environments and provides bandwidth aggregation functionalities.

Lee et.al. propose a 'virtual cache' for enabling data sharing among mobile hosts in CAPS [19]. CAPS require a subset of nodes to keep track of location of objects. The main emphasis of CAPS is on throughput enhancement in cellular networks whereas MoB focusses on an application-layer service infrastructure where cooperation is facilitated through incentives. A number of proposals have defined enhancements to cellular networks that improve throughput performance by enabling multi-hop ad-hoc network-style data forwarding. They include UCAN [23], MCN [22] and work by Aggelou et.al. [1]. Similarly work by Bejerano [3] and by Wu et.al. (iCAR) [35] have proposed deployment and traffic forwarding through relays to mitigate some of the congestion problems in various cellular network scenarios. Incentive-based multi-hop data forwarding in uncooperative environments have been examined in Nuggets/Nuglets [5, 6], in Sprite [40], by Ben Salem et.al. [32] and by Crowcroft et.al. [7]. Patel and Crowcroft in [27] present a ticketing scheme for access control to services in decentralised mobile environments such as MoB. Related work on a reputation model for supporting trust in virtual communities is described in [29], while reputation management based mitigation of routing misbehavior has been examined by Marti et.al. [24], in CONFIDANT protocol [4], and in peer-to-peer networks [20]. These approaches however focus primarily at network layer data forwarding mechanisms for improved performance.

Traffic/service pricing has also received significant attention in recent literature. Some examples of network pricing approaches can be found in work by La and Anantharam [18], Gibbens and Kelly [11], Key and McAuley [15], Semret et.al. [33], and Yaiche et.al. [36]. A relatively recent work by Lin et.al. [21] presents a game-theoretic framework for integrated admission and rate control of users involving multiple competitive cellular providers. In MoB we sidestep the service pricing question since in this open market environment such choices will be made independently by user and infrastructure providers. In fact, all prior work in this context can be very well leveraged to define appropriate price management in these environments.

7. CONCLUSIONS

We have presented MoB, an open market architecture for collaborative wide-area wireless services. Due to the laissez faire approach, devices in MoB can independently collaborate with other devices to improve application performance. By using financial-based incentives and reputation management, MoB allows users to drive their own rules of interaction. Trusting users can choose to conduct large number of MoB transactions with other users based on their reputation certificates alone. Other, more apprehensive, users can choose to conduct such interactions only with 'pre-trusted' users when they have no direct access to the Internet-based accounting (and token verification) service, and with other unknown users only when they have an alternate (even low-bandwidth) Internet access mechanism to connect to the accounting service to verify transaction payments. We believe that an architecture like MoB can promote fine-grained competition in wide-area wireless markets and ultimately prove beneficial for the users.

Acknowledgments

We thank Julian Chesterfield, Jon Crowcroft, and Pablo Rodriguez for helpful discussions on an earlier version of this work. We also thank the anonymous reviewers and our shepherd, Sujata Banerjee, for valuable feedback.

8. REFERENCES

[1] AGGELOU, G., AND TAFAZOLLI, R. On the relaying capability of next-generation GSM cellular networks. *IEEE Personal Communications Magazine 8* (2001).

[2] BAHL, P., AND PADMANABHAN, V. N. RADAR: An in-building rf-based user location and tracking system. In *Proc. of IEEE Infocom* (2000).

[3] BEJERANO, Y. Efficient integration of multi-hop wireles and wired networks with qos constrains. In *Proc. of ACM Mobicom* (Sept. 2002).

[4] BUCHEGGER, S., AND BOUDEC, J.-Y. L. Performance analysis of the confidant protocol. In *Proc. of ACM MobiHoc* (June 2002).

[5] BUTTYAN, L., AND HUBAUX, J.-P. Enforcing Service Availability in Mobile Ad-Hoc WANs. In *Proc. of ACM MobiHoc* (2000).

[6] BUTTYAN, L., AND HUBAUX, J.-P. Stimulating Cooperation in Self-Organizing Mobile Ad-hoc Networks. *ACM Journal on Mobile Networks* (2003).

[7] CROWCROFT, J. ET. AL. Providing incentives in providerless networks. *Journal of Ad Hoc Networks 2*, 3 (July 2004).

[8] DELLAROCAS, C. Analyzing the economic efficiency of ebay-like online reputation reporting mechanisms. In *Proc. of 3rd ACM Conference on Electronic Commerce* (2001).

[9] FREEMAN, W., AND MILLER, E. An experimental analysis of cryptographic overhead in performance-critical systems. In *Proc. of International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)* (Oct. 1999).

[10] FRIEDMAN, E., AND RESNICK, P. The social cost of cheap pseudonyms. *Journal of Eco. and Mgmt. Strategy* (2000).

[11] GIBBENS, R., AND KELLY, F. Resource pricing and the evolution of congestion control. *Automatica 35* (1999).

[12] GUTTMAN, E., PERKINS, C., VEIZADES, J., AND DAY, M. Service Location Protocol, Version 2. IETF Request for Comments - RFC 2608, June 1999.

[13] JOHNSON, D., AND MALTZ, D. *Dynamic Source Routing in Ad Hoc Wireless Networks*. Kluwer Academic Publishers, 1996.

[14] KAMVAR, S. D., SCHLOSSER, M. T., AND GARCIA-MOLINA, H. The eigentrust algorithm for reputation management in p2p networks. In *World Wide Web Conference* (2003).

[15] KEY, P., AND MCAULEY, D. Differential qos and pricing in networks: where flow control meets game theory. *IEE Proc. on Software 146*, 1 (Feb. 1999).

[16] KLEMM, A., LINDEMANN, C., AND WALDHORST, O. A Special-purpose Peer-to-Peer File Sharing System for Mobile Ad Hoc Networks. In *Proc. of VTC* (Oct. 2003).

[17] KOLLOCH, P. The production of trust in online markets. In *Advances in Group Processes (Vol. 16), edited by E.J. Lawler, M. Macy, S. Thyne, and H.A. Walker, Creenwich, CT, JAI Press* (1999).

[18] LA, R., AND ANANTHARAM, V. On admission control and scheduling of multimedia burst data for cdma systems. *IEEE Conference on Decision and Control 4* (1999).

[19] LEE, K., KO, Y., AND NANDAGOPAL, T. Load Mitigation in Cellular Data Networks by Peer Data Sharing over WLAN Channels. *Computer Networks 47*, 1 (Jan. 2005).

[20] LEE, S., SHERWOOD, R., AND BHATTACHARJEE, B. Cooperative Peer Groups in NICE. In *Proc. of Infocom* (Apr. 2003).

[21] LIN, H., CHATTERJEE, M., DAS, S., AND BASU, K. Arc: An integrated admission and rate control framework for cdma data networks based on non-cooperative games. In *Proc. of ACM Mobicom* (2003).

[22] LIN, Y.-D., AND HSU, Y.-C. Multihop cellular: A new architecture for wireless communications. In *Proc. of IEEE Infocom* (Mar. 2000).

[23] LUO, H., RAMJEE, R., SINHA, P., LI, L., AND LU, S. UCAN: A Unified Cellular and Ad-Hoc Network Architecture. In *Proc. of ACM Mobicom* (Sept. 2003).

[24] MARTI, S., GIULI, T., LAI, K., AND BAKER, M. Mitigating Routing Misbehavior in Mobile Ad-Hoc Networks. In *Proc. of ACM Mobicom* (Sept. 2000).

[25] MILLS, D. L. Network Time Protocol (Version 3). In *IETF Request for Comments (RFC) - RFC 1305* (Mar. 1992).

[26] PAPADOPOULI, M., AND SCHULZRINNE, H. Effects of power conservation, wireless coverage and cooperation on dissemination amoung mobile devices. In *Proc. of the ACM MobiHoc* (2001).

[27] PATEL, B., AND CROWCROFT, J. Ticket Based Service Access for the Mobile User. In *Proc. of ACM Mobicom* (1997).

[28] PERKINS, C., AND BELDING-ROYER, E. Ad hoc on-demand distance vector (AODV) routing. In *IEEE Workshop on Mobile Computing Systems and Applications* (Feb. 1999).

[29] RAHMAN, A. A., AND HAILES, S. Supporting Trust in Virtual Communities. In *Proc. of Hawaii International Conference on System Sciences 33* (2000).

[30] RESNICK, P., ZECKHAUSER, R., FRIEDMAN, E., AND KUWABARA, K. Reputation Systems. *Communications of the ACM 43*, 12 (Dec. 2000).

[31] RODRIGUEZ, P., CHAKRAVORTY, R., CHESTERFIELD, J., PRATT, I., AND BANERJEE, S. Mar: A commuter router infrastructure for the mobile internet. In *Proc. of ACM Mobisys* (June 2004).

[32] SALEM, N., BUTTYAN, L., HUBAUX, J., AND JAKOBSSON, M. A Charging and Rewarding Scheme for Packet Forwarding in Multi-hop Cellular Networks. In *Proc. of MobiHoc* (June 2003).

[33] SEMRET, N., LIAO, R., CAMPBELL, A., AND LAZAR, A. Pricing, provisioning and peering: dynamic markets for differentiated Internet services and implications for network interconnections. *IEEE Journal in Selected Areas of Communications 18*, 12 (Dec. 2000).

[34] SHARMA, P., LEE, S.-J., BRASSIL, J., AND SHIN, K. Handheld Routers: Intelligent Bandwidth Aggregation For Mobile Collaborative Communities. In *Proc. of IEEE BroadNets* (2004).

[35] WU, H., QIAO, C., DE, S., AND TONGUZ, O. Integrated Cellular and Ad Hoc Relaying Systems: iCar. *IEEE Journal on Selected Areas in Communications 19*, 10 (Oct. 2001).

[36] YAICHE, H., MAZUMDAR, R., AND ROSENBERG, C. A game theoretic framework for bandwidth allocation and pricing in broadband networks. *IEEE/ACM Transactions on Networking 8*, 5 (Oct. 2000).

[37] YOON, J., LIU, M., AND NOBLE, B. Random waypoint considered harmful. In *Proc. of IEEE Infocom* (2003).

[38] YOUSSEF, M., AND AGRAWALA, A. The horus wlan location determination system. In *Proc. of ACM Mobisys* (June 2005).

[39] ZHANG, H. ET.AL. Making Eigen vector-based Reputation Systems Robust to Collusion. In *Proc. of 3rd Workshop on Algorithms and Models for the Web Graph* (Oct. 2004).

[40] ZHONG, S., YANG, Y. R., AND CHEN, J. Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks. In *Proc. of IEEE Infocom* (Apr. 2003).