First International Conference on Peer-to-Peer Computing (P2P'01), 2001.

When Peer-to-Peer comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad hoc Networks

Gerd Kortuem, Jay Schneider, Dustin Preuitt,
Thaddeus G. C. Thompson, Stephen Fickas, Zary Segall

Wearable Computing Group
Department of Computer and Information Science
University of Oregon
Eugene OR 97403, USA
kortuem@cs.uoregon.edu

Abstract

This paper motivates and describes the notion of ad hoc mobile information systems*. Such a system consists of a decentralized and self-organizing network of autonomous, mobile devices that interact as peers. Connectivity is determined by distance between devices; as hosts change their physical location they establish pair-wise peering relationships based on mutual proximity. We describe application scenarios for ad hoc collaboration with mobile devices and identify technical challenges of mobile peer-to-peer systems. Moreover, we present the goals and architecture of* Proem*, a peer-to-peer system and development platform for mobile ad hoc applications.* Proem *has successfully been used as instructional tool in an advanced Software Engineering course on Peer-to-Peer Computing.*

Keywords

Peer-to-Peer computing, mobile computing, groupware, decentralized architecture, mobile ad hoc networks, personal-area networks

1. Introduction

Recent advances in wireless technology and mobile computing along with demands for greater user mobility have provided a major impetus toward development of mobile ad hoc networks (MANET). Ad hoc networks are self-organizing networks that are comprised of wireless nodes that cooperate in order to dynamically establish communication. Any device with a microprocessor, whether highly mobile or stationary, is a potential node in an ad hoc network.

A special class of ad hoc networks are personal area networks (PAN). They are low power, low range, wireless networks that provide connectivity among devices within or entering a personal operating space. This includes devices that are carried, worn, or located near the body. PANs enable devices to connect easily and with little intervention from their user thereby facilitating "unconscious" communications among personal devices. Examples of personal area networks include Bluetooth [14], Genuity's BodyLAN [16], Zimmermann intra-body network [17] and networks following the emerging IEEE 802.15 standard [15].

The combination of personal mobile devices (PDAs, wearable computers etc.) with wireless ad hoc networks allows the conception of *ad hoc mobile information systems*. Such a system consists of a highly dynamic, decentralized and self-organizing network of autonomous, mobile devices that interact as peers. In such a network, connectivity is determined by distance between devices; as hosts change their physical location they establish pair-wise peering relationships based on proximity. As result, the network is continuously reshaped into multiple clusters of two or more hosts. This model of ad hoc mobility describes an extreme mobile environment in which no fixed infrastructure exists to support communication.

Ad hoc mobile systems provide opportunities for a range of novel and interesting peer-to-peer applications. This includes collaborative systems for ad hoc meetings, mobile patient monitoring, distributed command and control systems and ubiquitous computing. In particular, personal-area networks enable the creation of proximity-aware applications in support of face-to-face collaboration. Mobile devices like cell phones, PDAs and wearable computers have become our constant companions that are available wherever we go. They are small and unobtrusive, and can be used most of the time and in most circumstances. By storing private information about people we know and the things we do, they function as our personal assistants that help us manage our daily life and our relations to other people. Personal-area networks open the opportunity for these devices to take part in our social interactions with people. Their ability to establish communication links among devices during face-to-face encounters can be used to facilitate, augment or even promote human social interactions. Examples of possible uses include applications that alert us to the presence of friends at a crowded public space [6] or identify people we want to meet using a match making algorithm that takes in account our preferences and interests [7, 8], systems that spread rumors [5], facilitate the exchange of personal information [9], or support us in more complex tasks like trading delivery tasks [10].

1.1. The Problem

In this paper we address the following questions:
What are useful and interesting peer-to-peer applications for ad hoc networks? What are the characteristics of these applications?
How are mobile peer-to-peer systems different from conventional, non-mobile systems? How do we need to redesign existing peer-to-peer systems and applications so that they function in highly dynamic mobile ad hoc environments?
What are the requirements for mobile ad-hoc middleware? Can we define a generic peer-to-peer software architecture for mobile ad-hoc computing?

The result of our research is *Proem*, a mobile peer-to-peer platform for collaborative applications in ad hoc networks. The key benefits of *Proem* are:

- High-level development support
- Platform independence
- Interoperability
- Extensibility
- Support for intermittent connections
- Built-in functions for naming, discovery, communication, data sharing, event logging, security, and privacy

Proem is currently used as instructional tool in an advanced Software Engineering course on Peer-to-Peer Computing [11] at the University of Oregon. An earlier version of Proem was described in [9].

1.2. Outline

This paper is organized as follows: In Section 2, we describe peer-to-peer applications for proximity-aware mobile collaboration. Next, we identify challenges of mobile peer-to-peer systems. In Section 3, we present the goals and overall architecture of our peer-to-peer platform *Proem.* In Section 4, we compare *Proem* to several other peer-to-peer systems. We conclude with an outlook on future research directions.

2. Mobile Peer-to-Peer Applications for Augmenting Face-to-Face Interactions

Personal mobile devices are becoming an ever-larger part of our social lives. Devices like cell phones, PDAs and wearable computers have become our constant companions that are available wherever we go. They are small and unobtrusive, and can be used most of the time and in most circumstances. By storing private information about people we know and the things we do, they function as our personal assistants that help us manage our daily life and our relations to other people. Most of these devices however, with the exception of cell-phones and pagers, are designed as mere productivity tools for single users. Despite the fact that these devices are often with us when we meet and interact with other people, they do not afford inter-personal exchanges. The limited infrared capabilities of some devices merely emphasize this point.

2.1. The Importance of Social Encounters and Face-to-Face Interactions

Social encounters of people play an important part in our social life; they are also vital for collaboration at the office and for coordination of work activities [13]. An encounter with another person, whether we know the person or not, is a chance for striking up a conversation and for exchanging information. Sometimes we use encounters to cooperate with other people and sometimes to pursue and advance our own goals.

It has been argued [4] that in today's world individuals are suffering from a lack of authentic psychological encounters or "human moments." For example, [39] recognized that Internet use of as little as four hours per week can result in higher level of depression and loneliness. Face-to-face interactions, on the other hand, have an immediate effect on hormones levels and have thus the potential of reducing feelings of stress and fear, and increasing feelings of trust, bonding and well being.

2.2. Mobile Technology for Augmenting Face-to-Face Interactions

With the advent of personal area network technologies our mobile and wearable device can participate with us in face-to-face interactions. PANs establish communication links among physically close mobile devices.. We can interpret this as if a PAN creates a *digital sphere* (*aura*) around every person. The extension of this aura depends on the transmission range of the wireless transceivers and can range from a few inches to several feet.

When two individuals come in close physical proximity or meet face-to-face their respective auras overlap enabling their personal mobile devices to interact. At that point, the devices can exchange information and access each other's services. On the other hand, when these individuals part and their auras no longer intersect, connections between their devices are broken and inter-device communication is no longer possible. This ability to establish communication links among devices during a face-to-face encounter can be used to facilitate, augment or even promote human face-to-face interactions. Examples of possible uses include applications that alert us to the presence of friends at a crowded public space [6], identify people we want to meet, using a match making algorithm that takes in account our preferences and interests [7, 8], spread rumors [5], facilitate the exchange of personal information [9], or support us in more complex tasks like trading delivery tasks [10].

2.3. Impromptu Collaboration

The focus of our research is in building mobile systems that assist us in our every day social interaction with other people, not only at the work place, but also during informal activities like shopping, when we pursue our hobbies or hang out with friends. In particular, we are interested in how this technology can promote social relationships among co-located persons during chance encounters.

We refer to these proximity-based ad hoc interactions using mobile devices as *impromptu collaboration*. Impromptu collaboration can happen in the absence of any enabling hardware other than what the collaborators commonly carry with them and is thus possible at any time and in any environment.

Impromptu collaboration is

- *Opportunistic*: it allows people to make take advantage of and make use of an opportunity that presents itself
- *Spontaneous*: it requires no prior planning or preparation on behalf of the human.
- *Proximity-based*: collaboration is made possible by physical proximity of two or more individuals.
- *Transient*: interactions are short-lived, seldom lasting more than a few minutes or even seconds.

In the following we will illustrate the concept of impromptu collaboration with several usage scenarios. We have successfully used these scenarios as requirements definition in an advanced software engineering course at the Department of Computer Science at University of Oregon [11]. As part of this course, which focused on peer-to-peer computing, eight second-year Master's students developed collaborative applications using the *Proem* peer-to-peer platform (see Chapter 4).

2.4. Impromptu MP3 File Sharing (Version 1)

Despite legal problems MP3 file sharing remains one of the most successful peer-to-peer applications. With the increasing use of personal mobile devices it becomes logical to think about a mobile version of Napster. We anticipate that next generation MP3 players will be integrated wearable devices (e.g., with phone, PDA, etc.) supporting wireless exchange of MP3 files. Indeed, companies like Parthus Technologies [12] have started to think about Bluetooth-enabled MP3 players. This will allow the following scenario:

"One evening Erik and Jennie decide to go out for dinner at a fancy downtown restaurant. On the way out they grab their ultra-portable Personal Mobile Assistants and strap them on. As they arrive at the restaurant they notice some friends among a large group of people and after some introductions they decide to join them for dinner.

While the appetizers are being served, Erik starts a conversation with his neighbor and soon they discover that they are both ardent fans of Cuban Jazz music. Since both carry their personal MP3 music collection on their Personal Mobile Assistants they immediately decide to swap digital music clips. After turning on the collaboration mode of their Personal Mobile Assistants they engage in a heated discussion about various musicians. Meanwhile, their mobile assistants create personal recommendation lists and start exchanging files."

MP3 File sharing using mobile computers is different from sharing files over the Internet for a number of reasons:

- **Social context**: In a mobile scenario involving PANs exchanges are only possible over short distances, that is when people come face-to-face or at least are within close physical proximity. Consequently, trading partners will be aware of whom they are trading with and be able to observe important social cues including sex, clothing and gestures. In addition, they might even be able to talk to each other. This is in stark contrast to anonymous MP3 file sharing on the Internet provided by Napster and Gnutella. It is clear that these differences in social context can lead to a difference in behavior regarding the general willingness to interact with strangers and the way people interact. For example, politeness and trust are two aspects of human interactions that strongly differ whether people interact face-to-face or terminal-to-terminal across the Internet.
- **Usage context:** The context in which mobile devices are used is different in two important respects from using a stationary computer at home or in the office. When mobile devices are involved, user attention is a scarce resource. Instead of sitting in front of a computer where we can pay full attention to the computer and its operation, handheld and mobile computers are often used in situations where our attention is occupied by

demanding real-world task like driving, operating a machine, or simply conversing with other people.

Furthermore, with mobile computers time becomes a critical resource as well. When surfing the Internet at home or at the office we are less likely to care if a download takes just a few seconds or several minutes; we will always find something else to do in the meantime. This is no longer true if we exchange information from one mobile device to another, because this requires our physical presence. If a transfer takes too long we are less likely to complete it, because we might run out of patience or must hurry to the next appointment.

- **Technical context**: Mobile devices and wireless networks are severely limited in the following aspects: processing power, storage capacity, and bandwidth. This suggests that rather than exchanging mp3 files of several megabytes, we might want to exchange URLs that point to the files on a server. The actual download of the music file might occur at a later time.

  Another severe handicap of mobile devices relates to the user interface. Some mobile devices are missing simple and efficient ways to enter text or have a display that is too small for effective browsing of large amounts of data. Consequently, the traditional way of finding music on the Internet, that is via keyword search, is inappropriate for mobile settings. Imagine entering the name of an artist or music song on a PDA-like device in the dim light of a bar while conversing with someone you just met.

These observations make clear that it does not suffice to simply re-implement a Napster or Gnutella client for mobile devices. Instead, it becomes necessary to rethink the whole system design and modify it in accordance with the social, usage, and technical context.

2.5. Impromptu MP3 File Sharing (Version 2)

A possible approach is to develop *personal agents* that act on the behalf and in the interest of their users [10]. The task of such agents would be to find and interact with other peoples' personal agents for a variety of well-defined purposes, either automatically as side effect of a chance encounter or when explicitly instructed to do so.

A personal agent for MP3 file trading could use a number of criteria to determine which music to download: most likely these criteria would contain a user's wish list and music preferences in terms of artists, music genre etc. Likewise, it would be possible to employ matching algorithms as described in Yenta [7] to determine people with similar music taste and extract recommendations from their music library. Moreover, a personal agent could base its decisions on information about human relationships and trust. For example, the agent could know that we are inclined to download and listen to a song if it is recommended by a friend who's taste in music we trust.

This approach is highlighted in the next scenario:

"Kim is a 20 year old architecture student and an ardent music lover. She has an extensive CD collection and has digitized all of them as MP3s. During most of her waking hours she can be seen sporting an MP3 player and headsets, especially when she is working in her studio at school. In order to stay up-to-date on what is going on in the music scene and to hear about latest releases of her favorite artists she subscribed to several mailing lists and follows the discussion threads on several music web sites.

When the latest generation of Bluetooth-enabled wearable MP3 players came out she could not resists to buy one despite the steep price: she knew that most of her friends at school would do the same. Her new MP3 player has the ability to detect other MP3 players in the vicinity and to trade play lists when she passes another person in the hallway or when she sits at the same table in the cafeteria. Kim set up her player to recognize the MP3 players of all of her music loving friends and to automatically trade information about the songs she listens to. Furthermore, she enabled the privacy mode of her MP3 player. As a result, her player emits an audible signal every time it trades song lists with another device. However, more important is that fact that other people's players are prevented from accessing her private information.

After each day at school, Kim downloads the information she collected from her friends players to her laptop computer that is connected to the Internet. She then downloads freely available samples of the songs her friends listen to and copies some of them to her MP3 player. On a good day she identifies two or three new CDs she plans to buy as soon as she has enough money."

This scenario stresses two important points:

First, impromptu collaboration with personal mobile devices does not necessarily involve direct human interaction. People might not be aware of the fact that their respective devices interact. The general willingness to share information and the opportunities presented by physical proximity suffice for successful collaboration.

Second, privacy is very important for impromptu collaboration. The very fact that exchanges can occur without Kim's knowledge makes it tantamount to provide measures to protect her private data from unauthorized access.

2.6. Impromptu MP3 File Sharing (Version 3)

Yet another version of the MP3 file-sharing scenario further emphasizes the need for security:

"The Music Trading Organization, an industry business group, has recognized both the need for copyright protection and a business opportunity in collecting royalties from digitally exchanged music. They

developed technology that makes it possible that a small royalty gets automatically paid to the copyright owner when two individuals exchange copyrighted music. Further, both individuals involved in a trade will get a small payment every time they generate royalty for the copyright owner.

Eric and Jennie, both owners of copyright-enhanced wearable MP3 players, meet at a party. After listening to a song they both like, they begin discussing music and find they have a similar interest in Latin Calypso music. At Eric and Jennie's request their wearable computers start comparing music profiles and identify songs that Eric and Jennie might want to exchange. As result, their wearable computers wirelessly send selected songs to each other and record the transaction in a tamper-resistant database module.

When returning home from a long day, both Eric and Jennie synchronize the database on their wearable personal assistants with their desktop computer. As side effect, MP3 transactions that happened throughout the day are automatically uploaded to the server of the central Music Trading Organization which in turns arranges payments to and from Eric and Jennie's bank accounts."

In this scenario, Eric and Jennie, the copyright owner, the producer of the wearable computer hardware, multiple developers of software components, the e-commerce service providers and perhaps many others, become part of the wireless wearable e-business community and are interested not only in the proliferation of the community but also in the way the requirements for the system are evolving.

This scenario stresses two additional aspects of impromptu collaboration:

First, it points to the necessity of security in mobile ad hoc networks. Users must be authenticated and payment transactions must be processed and recorded.

Second, this scenario also indicates that impromptu collaboration can span mobile ad hoc networks and wired infrastructure networks like the Internet.

2.7. Task Trading

Our final scenario derives from our earlier work on task trading in "Wearable Communities" as realized by the WALID system [10]. WALID implements a digitized version of the timeworn tradition of borrowing butter from your neighbor. You do a favor for others because you know that one day they will do it for you.

With WALID two individuals use their mobile devices to negotiate about and to exchange real-world tasks: dropping off someone's dry cleaning, buying a book of stamps at the post office, or returning a book to the local library.

WALID employs personal agent software to find close-by community members and to negotiate the exchange of tasks. The agents maintain a user's task list, becoming fully aware of the locations and activities involved. When an encounter occurs, the agents produce a negotiation. If both users approve, a deal is struck.

The role of the agent in a negotiation is to evaluate the value of favors and to keep scores. Having to run across town just to drop off someone's mail compares unfavorably with buying milk for someone if the grocery store is just a block away. Agents employ ideas from game theory to ensure that results of negotiations are mutually beneficial. They cooperate only if there is the opportunity to enhance the users' goals.

2.10. Conclusion

We are interested in the use of mobile and wearable computing technology to enhance people's social interactions during unexpected and unplanned encounters in the real world: when people meet on the way to the office, in the elevator, or at the grocery store. Our research is based on the premise that in the near future a large percentage of the population will use handheld or wearable computing devices with PAN capabilities. The ubiquity of this kind of mobile technology will enable new forms of computer-supported human interactions.

The primary aim of our research is to develop new software technologies and to build practical applications that function in the real world. In the next chapter we will discuss the technical challenges for building such systems.

3. Challenges of Ad Hoc Mobile Information Systems

In order to provide the style of mobile collaboration described above, sophisticated mobile systems and applications that can operate in mobile ad hoc networks are required. We use the term *ad hoc mobile information systems* in order to describe such a system. An *ad hoc mobile information systems* consists of a highly dynamic, decentralized and self-organizing network of autonomous, mobile devices that interact as peers. Connectivity among devices is determined by their relative distance; as hosts change their physical location they establish pair-wise peering relationships based on proximity. In this chapter, we will discuss technical challenges of such systems.

3.1. Ad hoc Networks

Wireless ad hoc networks are self-organizing networks that are comprised of wireless nodes that cooperate in order to dynamically establish communication. Any device with a microprocessor, whether highly mobile or stationary, is a potential node in an ad hoc network.

Ad hoc networks have a number of advantages compared to traditional wireless cellular networks. These include:

- **No Infrastructure required**: Ad hoc wireless networks don't rely on wired base-stations and for that reason can be deployed in places without existing infrastructures. They can be created spontaneously and on as needed basis, because they require little configuration to setup.
- **Self-organization**: In a wired network the connection topology of nodes is determined by the physical cabling and thus is fixed. This restriction is not present in ad hoc network: as soon as two nodes are within hearing distance of each other, an instantaneous link between them is automatically formed. As a consequence, the network topology of an ad hoc network reflects the relative distance of its nodes and is continuously reconfigured as nodes come into reach of each other.
- **Fault Tolerance**: The self-organizing nature of ad hoc networks and the fact that they don't rely on dedicated base stations makes ad hoc networks fault-tolerant. In a traditional cellular network, a fault in the base station will impair all nodes in its cell. In ad hoc networks, a malfunction in one node can be easily overcome through network reconfiguration.

Personal area networks (PAN) are a particular class of ad hoc networks optimized for low complexity, low power, low range and low cost. PANs provide ease-of-connectivity of personal wearable or handheld devices thereby facilitating "unconscious" communications among personal devices. They have small coverage, typically about 10m, and allow only a limited number of devices to be connected in the same geographic region. Examples of PANs include Bluetooth [14], networks following the emerging IEEE 802.15 standards [15], Genuity's BodyLAN [16], and Zimmermann's intrabody network [17].

3.2. Mobile Peer-to-Peer Systems

Traditionally, mobile devices have been designed as thin clients as part of a client-server system. Mobile devices such as cell-phones or Personal Digital Assistants (PDAs) use wireless connections to gain access to resources such as data and computation provided by large central servers.

The emergence of wireless ad hoc networks and powerful mobile devices has made it possible to design mobile systems as peer-to-peer systems. Such a mobile peer-to-peer system or network consists of personal mobile devices that interact during brief physical encounters in the real world thereby engaging in short-haul wireless exchanges of data. Mobile peer-to-peer applications take advantage of resources -- storage, cycles, content, human presence – provided by mobile (or stationary) devices in the immediate physical proximity.

Bolcer at al [18] describe peer-to-peer as "any relationship in which multiple, autonomous hosts interact as equals. An autonomous host is useful in it's own right even in the absence of others. The peering relationship implies that additional functions are available to other peers *collectively* as a consequence of their collaborations with other hosts. Known as the *network effect*, the value and extent of these added powers increases dramatically as the number and variety of peers grows".

This definition of peer-to-peer computing closely matches our ideas as outlined in the previous section. Personal mobile devices are autonomous and provide great benefits to their users even when not connected to other devices. However, as soon as two devices come within reach and a communication link is established, they enter into a short-lived, yet mutually beneficial partnership by exchanging data and accessing each other's services.

A mobile peer-to-peer system inherits many of the features of ad hoc networks:

- **Self-organizing**: as side effect of the movement of devices in physical space, the topology of a mobile peer-to-peer system constantly adjusts itself by discovering new communication links.
- **Fully decentralized**: each peer in a mobile peer-to-peer system is equally important and no central node exists.
- **Highly dynamic**: Since communication end-points can move frequently and independently of one another, mobile peer-to-peer systems are highly dynamic.

However, despite this similarity in character there are clear differences between traditional and mobile peer-to-peer system when it comes to their realization. Bolcer at al [18] describe (non-mobile) peer-to-peer computing as the natural and desirable outcome of three profound and pervasive trends:

- The ease of interconnection
- The expansion of bandwidth
- The wealth of cycles

"… in a world where (network access) interconnection is universal, (network) bandwidth is plentiful, and (processor) cycles are inexpensive, peering among physical unequals is both natural and desirable." ([18], p. 3).

Existing peer-to-peer systems such as Napster [40], Gnutella [35,36], Freenet [41], and Groove [42], which are intended to run on stationary hosts in wired networks, have been designed with this view in mind. However, two of these trends do not apply to mobile peer-to-peer systems: while ad hoc networks provide ease of interconnection, there is much less expansion of bandwidth in wireless networks than there is in wired networks, and much less wealth of cycles in mobile devices than there is in desktop units.

The unique character of mobile peer-to-peer systems represents a significant challenge for the designer. In the following, we will discuss some of these challenges in more detail.

3.3. Challenges

3.3.1. Networking

Wireless data networks present a more constrained communication environment compared to wired networks. Because of fundamental limitations of power, available spectrum, and mobility, wireless data networks tend to have less bandwidth, more latency, less connectivity stability, and less predictable availability. Furthermore, as bandwidth increases, the device's power consumption also increases further draining the already limited battery life of mobile devices. Thus, even as wireless networks improve their ability to deliver higher bandwidth, the power availability still limits the effective throughput.

Communication links in proximity-based ad hoc networks are prone to unexpected interruptions. Consequently, mobile peers must anticipate frequent network failures and handle them gracefully. In addition, peer applications should provide for disconnected operations [19] such that a peer remains operational even without network connection.

Communication between arbitrary peers in a mobile peer-to-peer network requires routing over multiple-hop wireless paths. The main difficulty arises because without a fixed infrastructure these paths consist of wireless links whose end-points are likely to be moving independently of one another. Consequently, node mobility causes the frequent failure and activation of links, leading to increased network congestion while the network routing algorithm reacts to topology changes [19].

The instability of multi-hop paths and the limited lifetime of routes in ad hoc networks have a negative impact on the performance on peer-to-peer routing. Gnutella uses routing as a way to return search results to the peer that initiated a search. Search results travel backwards on the same route the original search query took. If one of the intermediate peers disappears from the network before the search result could be returned, the search result will be lost. In our own experiences using Gnutella this happens in only about 2% of all cases. In an ad hoc network, this number would increase dramatically.

3.3.2. Mobile Device Limitations

Mobile devices present a more constrained computing environment compared to desktop computers. Because of fundamental limitations of battery life and form factor, mobile devices tend to have less powerful CPUs, less memory, less storage, restricted power consumption, smaller displays, missing or restricted input devices. In the previous chapter, we discussed some techniques how an agent-based approach can alleviate user-interface problems. Less powerful CPUs make it harder to implement CPU intensive cryptographic security measures.

3.3.3. Naming

Traditional (non mobile) peer-to-peer systems are characterized by an increasing decentralization and autonomy of hosts. Because accessing these decentralized resources means operating in an environment of unstable connectivity and unpredictable IP addresses, peer-to-peer systems often operate outside the DNS system. The same must be true for mobile peer-to-peer systems. Additional reasons for not relying on the DNS system are:

- In ad hoc networks, access to a central DNS server cannot be assumed
- Not all mobile devices support IP networking and thus do not have IP addresses
- Impromptu collaboration requires the ability to identify not only peers, but also the people who run and use these peers. For example, in our MP3 file-sharing scenario from the last chapter, Kim authorized only her friends to access her play lists. Since Kim does not want to specify each friend's MP3 player, the system must be able to determine that a device belongs to a particular user.

3.3.4. Resource Discovery

One of the things that makes current peer-to-peer system so powerful is that they take advantage of resources -- storage, cycles, content, human presence -- available at the edges of the Internet. In a mobile peer-to-peer system we want to take advantage of resources provided by peers running on mobile (or stationary) devices in the immediate physical proximity. Because of the unpredictable physical mobility of mobile devices, discovering resources becomes a challenge.

The highly dynamic nature of mobile peer-to-peer systems requires similarly dynamic mechanisms for device and resource discovery. In ad hoc networks, device discovery is part of the network; resource discovery, however, is the task of the peer system. We need algorithms through which a computing device can detect the presence of neighboring devices, share configuration and service information with those devices, and notice when devices become unavailable. Resource discovery must be timely (in order to detect moving devices) and efficient (so not to overload the network) [31].

In contrast to peer-to-peer systems that are targeted at fixed networks, decentralization is not a mere option for mobile peer-to-peer networks, but a necessity. Even seemingly decentralized peer-to-peer systems sometimes rely on centralized servers. For example, many Gnutella clients use a central host cache for determining entry points into the Gnutella network. A mobile Gnutella client could not rely on such a host cache, but would have to discover appropriate peers in its surrounding.

3.3.5. Data Sharing and Synchronization

An ad hoc mobile information system is basically a highly dynamic, decentralized distributed system with weakly connected mobile hosts. In order to cooperate to the fullest extent peers need to be able to share and synchronize data. The extreme decentralization and unpredictability of ad hoc mobile system together with the fact that peers always only establish pair-wise connections leads to the following conflicting requirements:

High availability: On the one hand we want peers to be autonomous as much as possible. They should be able to perform computations even in the absence of connections with another peers. Thus we need to employ a replicated object scheme where each peer maintains a local copy of each shared data object.

Consistency: On the other hand, any replicated object scheme introduces the problem that copies of a shared object can be updated independently and thus might become inconsistent.

Timeliness: Finally, any solution to the consistency has to cope with the problem that data might be shared across a group of peers that never meet all at the same time. It is possible that all interactions in an ad hoc system always only occur between two peers. This situation can lead to a slow propagation of updates throughout the whole system.

3.3.6. Security

The security implications of mobile peer-to-peer systems must be taken seriously. Without countermeasures it is potentially possible to track every movement of an individual as well as examine what they are doing, System security cannot be restricted solely to the link layer, but it must encompass every layer of communications, including the applications and the data that these applications manipulate. In ad hoc networks users may not even be aware to which devices they are connected or, more importantly, which devices are connected to theirs. Someone in the next room or on the floor above may connect to someone else's mobile device and gain access to private data such as stored e-mail and meeting schedules. As a consequence, it is not enough to employ encryption to avoid eavesdropping, but robust authentication procedures need to be established for connecting both trusted and non-trusted devices with each other.

A particular security aspect of mobile devices and decentralized systems relates to the question "how do we know we can trust somebody on the network?" In systems with a centralized component, this problem can be handled by globally trusted certification authorities (CA) that issue public key certificates, cryptographically signed by the CA itself. This certificate proves that its holder is trustworthy simply because the issuer, the trusted CA, has signed it and can therefore vouch for the holder's credentials. A chain of CAs, each trusting the next CA in the chain, may sometimes be necessary when a certificate signed by an unknown CA is presented.

For ad hoc systems we need efficient distributed authentication protocols. This, however, is made difficult by the fact that this must occur in a completely decentralized environment with no or intermittent connection to a trusted authority. Possible solutions might include the use of reputations [20].

In order to engineer a fully secure mobile system it becomes necessary that the device is able to authenticate the user. Otherwise digital certificates can easily be stolen by taking the device itself. Commonly, passwords are used for this task. Depending on the required security level it might be necessary to employ biometric identification instead.

3.3.7. Privacy

Privacy is the right of individuals to control collection and use of personal information about themselves. Unlike security, which deals with safeguarding of information from unauthorized users, privacy is concerned with the amount of information known about an individual. Privacy can often be guaranteed through security measures.

One of the main privacy concerns is protecting a user's anonymity. Monitoring network traffic or gaining access to confidential personal data can compromise a user's anonymity. Not only must a system prevent spying and monitoring, but users must also given control what information is disclosed, to whom, and when. In particular, it must be possible for an individual to stay anonymous if so desired.

3.4. Conclusion

Ad hoc mobile systems, characterized by decentralization and peer-to-peer interactions, must be engineered for a number of often-conflicting requirements. For example, we need to define solutions for particular problems like data synchronization or security. Answers will most likely take the form of algorithms and protocols.

Yet another important aspect of ad hoc mobile systems is related to software engineering and the question of how we can support the application developer. In order to simplify the task of the application developer we need high-level development support in the form of tools and platforms. Among other things, such a platform must include support for naming, communication, discovery, security and privacy. It must define an application environment that facilities the development and deployment of ad hoc applications. In the following chapter we will discuss such a platform.

4. *Proem*: A Mobile Peer-to-Peer Platform

4.1. The Vision and Goals of *Proem*

Proem is an open computing platform targeted at ad hoc mobile information systems. It provides a complete solution for developing and deploying collaborative peer-to-peer applications for mobile ad hoc networks and personal area networks.

The motivation for developing *Proem* arose from our experiences in implementing a series of mobile applications for face-to-face collaboration [9,10]. Over time we identified enough commonalities among these applications to merit the development of a generic software platform. The objectives for *Proem* include:

- **Adaptability**: *Proem* is designed to facilitate rapid and timely response to changes in the operating environment, for example regarding connectivity and resource availability.
- **Universality**: *Proem* is an infrastructure for building diverse mobile communities ranging from MP3 file sharing to instant messaging. In contrast to systems like Napster or Gnutella, which are designed for one particular purpose (mp3 file sharing), *Proem* provides the building blocks for a wide range of peer applications..
- **Interoperability**: *Proem* is designed to allow interoperability between heterogeneous hardware and software platforms.
- **Platform independence**: *Proem* is designed to be independent of programming languages, system platforms and networking platforms. We achieved this goal by leveraging open web standards and technologies like http, xml, mime, etc.
- **Extensibility**: Developers should be able to modify the internal working of *Proem's* core components.
- **High-level development support**: The most prominent goal in designing *Proem* was to provide a simple yet powerful development platform that facilitates the implementation of mobile peer-to-peer applications.

4.2. Important Concepts and Terminology

Before we go into details of the *Proem* platform we must first clarify some concepts that are fundamental for its understanding.

4.2.1. Entities

The *Proem* architecture defines the following four entity types where an entity can be seen as a named object:

- **Peer**: A peer is any autonomous, mobile host or device taking part in a peer-to-peer relationship. In *Proem* each peer is uniquely identified by a Uniform Resource Identifier (URI) of the form `proem://<peer-name>`
- **Individual:** An individual is a person who owns and uses one or more peers. We assume mobile devices to be personal devices that are not routinely shared. Any person might use any number of peers, but each peer belongs to only one individual.
- **Data Space:** A data space is a collection of data items that are cooperatively owned and managed by a set of peers. Data spaces are stored in a replicated fashion on all peers that share them.
- **Community:** A community is a set of entities (peers, individuals, data spaces and other communities). Each entity can be a member of several communities (including none) and each community can contain members of different type. Communities can be used to define access rights to data and functionality or simply as a way to group entities. Examples of common communities include:
  - The set of peers owned by a particular user
  - A set of individuals who are friends and who grant each other special access rights
  - The set of data spaces related to a particular project
  - The set of all entities related to a particular project: individuals, peers, data spaces.

  The concept of communities is different from the notion of groups as commonly defined in distributed systems (including the peer-to-peer platform JXTA []). A community is an open set of entities. Membership is not controlled by the owner of the group (which might be one particular member or the collective of all members), but can be passed on by any member to any other entity. As a consequence it is impossible to determine the complete set of members of a community: no single authority controls membership and members can join at any time. Membership is conferred upon an entity by passing along a secret membership token that is unique to the conferring entity.

  Communities represent realms of trust. In order to 'prove' membership in a society, an entity has to produce a minimal number of valid tokens that are also known to the verifying entity. However, it is up to the verifying entity to accept or reject tokens as proof. It is up to each individual entity how much proof it requires before it trusts another entity. Communities only provide a mechanism for trust, policies can be defined by individual peers or applications.

4.2.2. Identities, Names and Profiles

Entities are identified by *names*. Names are expressed by Uniform Resource Identifiers (URI). Each entity can have one or more names. For example, one and the same peer can be known as `proem:peer:0101` or as `proem:peer:2222`. Multiple names provide for pseudo-anonymity: since there is no central name repository it is impossible to determine whether two different names refer to the same or two different identities. Each name, however, is unique and can only refer to one entity.

Proem provides a second way to refer to entities besides using explicit names. Entities can indirectly be referenced by *profiles*. A profile is an XML-based data structure for describing *Proem* entities, i.e. meta-data. A profile for an individual might contain his real name, address and email address while a peer profile could include a list of data spaces it has access to. Profiles function as intentional names and are used to advertise the presence of entities with particular attributes throughout a network.

4.2.3. Protocols and Messages

At the highest abstraction level, *Proem* simply can be viewed as a set of communication protocols that define the syntax and semantics of messages that peers can exchange. The definition and use of peer protocols guarantees interoperability between implementations of the *Proem* system on different hardware and software platforms.

Proem defines four protocols, one low-level transport protocol and three higher-level protocols.

The ***Proem* Transport Protocol** is a connectionless asynchronous communication protocol. Data is passed from peer to peer in one atomic unit. The *Proem* Protocol uses XML for representation of messages and can be implemented on top of a variety of existing protocols such as TCP/IP, UDP or HTTP. When an unreliable transport protocol is used messages may be delivered more than once, may not arrive at all, or may arrive in a different order than sent. The reception of a message is not acknowledged unless explicitly specified by the protocol.

Messages are the basic unit of communication between peers. Messages are addressed to and are sent from one peer to another. Messages are encoded as XML documents. This allows peers to implement the protocol in a manner best suited to its abilities and role. In particular, *Proem* peers can be implemented in any programming language and do not require a specific transport protocol.

A collection of peers that communicates using the *Proem* Transport Protocol form a *Proem network*. Each host that implements the protocol can become part of a *Proem* network. In a *Proem* network, each peer operates independently and asynchronously of any other peer.

The Proem core protocols are:

- The **Presence Protocol** contains messages that allow peers to announce their presence and the availability of entities throughout a network. The primary message type of the presence protocol is profiles.
- The **Data Protocol** contains messages that allows peers to share and synchronize data by means of data spaces.
- The **Community Protocol** contains messages for applying for, granting and verifying community membership.

In addition to these built-in protocols, application developers can define their own application- specific protocols. This way, *Proem* can be extended to support MP3 file sharing or any other peer-to-peer application as an extension to the base *Proem* protocol.

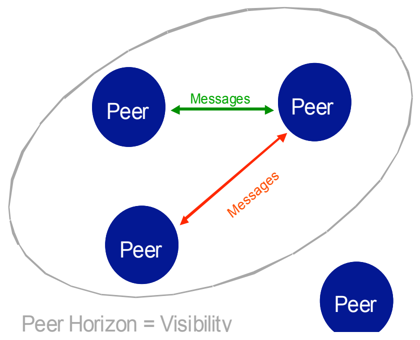

Figure 1. Peer Horizon*Proem* Platform

The Proem platform is a collection of tools, APIs and runtime structures for developing and deploying applications within the Proem framework. It currently exists in form of a Java implementation [34].

The *Proem* platform currently consists of two components:

- The ***Proem Runtime System*** is a proof of concept implementation of a peer that 'speaks' the Proem protocols. It consists of an implementation of the *Proem* protocol stack and the **Peerlet Engine.** Peerlets are simple structured peer-to-peer applications similar in purpose to Java servlets that follow an event-based programming model. Peerlets are the locus of computation and function as communication end-points. They are designed as drop-in modules and can be added to and removed from the Peerlet Engine at runtime. The peerlet engine in turn controls the execution of peerlets. Peerlets react to and communicate via events. The peerlet engine fires events to peerlets as reaction to changes in its internal state or as reaction to messages received by remote peers. Peerlets are notified of and handle events asynchronously.
- The ***Peerlet Development Kit (PDK)*** is a set of high-level Java APIs for developing peerlets. The PDK provides an extensive set of high-level APIs for communication, event handling, and management of entities.

shows the relationship between peers, peerlet engine, peerlets, messages and events.

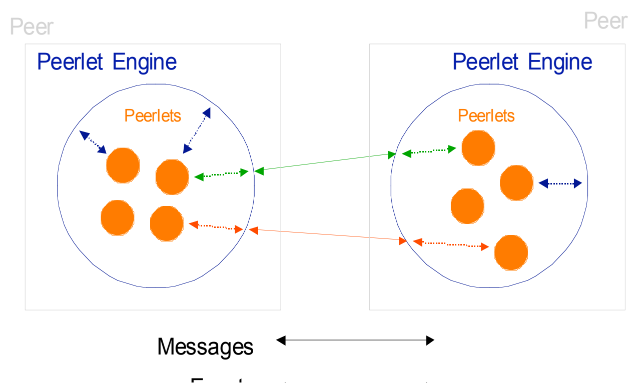

Figure 2. Peer Architecture

4.3.1. *Proem* Runtime System

The *Proem* Runtime System is a proof-of concept implementation of a *Proem* peer. It implements the *Proem* protocol and manages the execution of peerlets.

The overall architecture of a peer is shown in Figure 3. It consists of three components:

- The *Communication and Network Manager* handles basic tasks such as communication and security.
- The *Peerlet Engine (Peerlet Container)* is the run-time environment for peerlets. It controls the execution of peerlets and provides them with access to *Proem Services*.
- The final component is made up of a set of *Proem Services*. Facilities provided by services include mechanisms for naming, discovery, security and event-logging as well as management of user and trust related information.

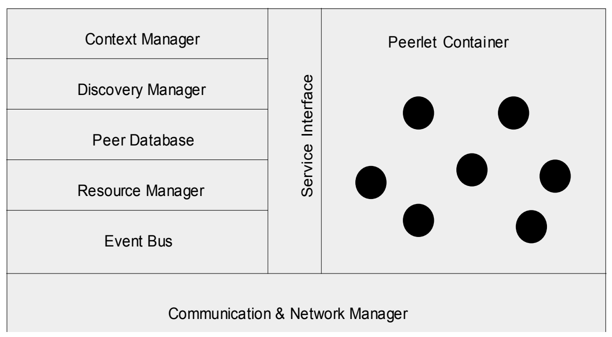

Figure 3. Peer Architecture (add profile manager and protocol manager)

The *Proem* Services are implemented by a number of system components. These are:

- The *discovery manager* is responsible for announcing a peer's presence and for discovering nearby peers. The meaning of "nearby" depends on the current network topology and includes all peers that are reachable either directly or indirectly.
- The *context manager* maintains information on peers that are visible at each moment.
- The *peer database* maintains a persistent log on encounters with other peers and allows peerlets to store custom meta-information on peers. This enables peerlets to determine when and how often a particular peers has been encountered in the past.
- The *resource manager* allows peerlets to define resources that they want to share with other peers. The resource manager maintains a version history for each resource and generates update notifications. It also performs access control.
- The *event bus* enables event-based communication among systems componets including peerlets. It provides a *publish and subscribe* model that allows anonymous exchange of data. System components and peerlets can announce the availability of data item and express interest in data by subscribing to update and event notifications. Events are used by the discovery manager to inform peerlets about encounters by reporting the appearing or disappearing of peers.
- The *profile manager* maintains information about the user and his/her relations to other users. This information is used, for example, for establishing trust relationships.
- The *protocol manager* maintains specifications of the application-specific protocol supported by this peer.

All of these components are themselves implemented as peerlets. This enables independent developers to change the implementation of these system components by replacing a system peerlet with their own version. This is possible since peerlets are plug-in modules that can easily be removed or installed.

4.3.2. Peerlet Development Kit (PDK)

The *Peerlet Development Kit (PDK)* is a collection of Java interfaces and classes for rapid development of peerlets. It enables independent programmers to develop peerlets that can be installed and executed by the *the* Peerlet Engine.

Important classes and interfaces include:

- Peer
- Peerlet
- Encounter
- *Proem*Event
- *Proem*Protocol
- *Proem*Message
- *Proem*Service

The *Proem* Development Kit is available online at http://wearables.cs.uoregon.edu/proem.

5. Related Work

5.1. *Proem* and Ad hoc Networks

Proem is a peer-to-peer platform for mobile ad hoc networks. An ad hoc network is a wireless network formed by nodes that cooperate with each other to forward packets in the network. Examples include Bluetooth [14], networks following the emerging IEEE 802.15 standards for PANs [15], Genuity's BodyLAN [16], and Zimmermann's intrabody network [17]. Most research on ad hoc networks focuses on the lower layer of the protocol stack including the link layer, network layer and transport layer. The application layer is an area that still needs much attention and we have yet to see many compelling applications that exploit this network technology. We view our work on impromptu collaboration and the *Proem* peer-to-peer platform as an important step in this direction.

The most important issue in the development of mobile peer-to-peer systems such as *Proem* is the integration of functions and services provided by underlying ad hoc networks. Ad hoc networks provide low-level functions that span the following four areas:

- Device Discovery
- Routing
- Multicasting
- Information Dissemination
- Security

5.1.1. Discovery

In general, existing ad hoc networks only support discovery of devices, not of services provided by these devices as required for peer-to-peer systems. An interesting discovery algorithm for ad hoc networks that combines device and service discovery is DEAPspace [31,32]. DEAPspace can detect the presence of neighboring devices, share configuration and service information with those devices, and notice when devices become unavailable. Targeted for wireless ad hoc, single-hop networks, this algorithm reduces the number of transmissions required from individual devices.

5.1.2. Routing

Communication between arbitrary peers in a mobile peer-to-peer network requires routing over multiple-hop wireless paths. The main difficulty arises because without a fixed infrastructure these paths consist of wireless links whose end-points are likely to be moving independently of one another. Consequently, node mobility causes the frequent failure and activation of links, leading to increased network congestion while the network routing algorithm reacts to topology changes [19]. Routing has been the single most active area ad hoc networking research. These include [5], [22] [23], [24], [25], [26], [27].

5.1.3. Multicasting

Multicasting is important for efficient dissemination of data throughout a network. For example, service discovery often relies on disseminating service advertisements to interested partners. In wired IP networks multicasting uses public multicast groups. Any host can join or leave a multicast group at will. In ad hoc networks we are interested in multicasting data to hosts based on specific host properties. For example, [28] describes a multicast routing protocol based on neighbor relationships; [29] use roles to form multicast groups; and [30] use content as criteria.

5.1.4. Information Dissemination

SPIN [21] is one of the first works towards building adaptive protocols for information dissemination in ad hoc networks. Each node advertises to its set of neighbors whenever it has some interesting information. SPIN optimize on the power consumption in the nodes.

5.1.5. Security

Not many ad hoc networks provide built-in security. Bluetooth [14] is the rare exception. It provides a number of built-in security measures including authentication and encryption at the link layer. Four different entities are used: a public address which is unique for each user, two secret keys, and a random number which is different for each new transaction. Unfortunately, Bluetooth provides security only for connection-oriented communication, not for connectionless packets. More importantly, because Bluetooth does not use a Public Key Infrastructure (PKI), secure relationships between Bluetooth devices must be setup *a priori* and cannot be established dynamically.

5.2. *Proem*, Gnutella and the Transient Web

Gnutella [35,36] is the most prominent example of a decentralized peer-to-peer system. It was developed to provide capabilities similar to the file-sharing network Napster and is mostly used for trading music and image files. Unlike Napster, Gnutella does not use a central server to keep track of all user files. Instead, messages are transmitted in a decentralized manner: a peer sends a search request to its 'neighboring' peers, who in turn pass that request along to their 'neighbors' and so on.

Proem and Gnutella are similar in some respects:

- Both *Proem* and Gnutella are protocol-based, i.e. they define an application level communication protocol.
- Both systems have a decentralized architecture
- In both systems, connections among peers are transient

However, there are significant differences:

- *Proem* is a general-purpose platform for building arbitrary mobile peer-to-peer applications. The Gnutella network, on the other hand, has been designed as distributed file storage and for sharing files. One could implement a system similar to Gnutella, but designed for mobile devices and ad hoc networks using *Proem*. This is in fact what our Software Engineering students did [].
  It is true, however, that Gnutella can be used for purposes other than file-sharing, because it's protocol specification only defines the structure of messages, but not how messages are to be handled by peers. A peer is free to interpret a search string any way it wants. For example the Gnutella network has successfully been used as chat application. Nevertheless, unlike *Proem*, Gnutalla does not provide any support for application developers.
- *Proem* is designed for highly dynamic mobile environments in which the network topology and availability of resource is changing constantly. The physical mobility of peers causes links to be established and broken on a regular basis and with a high frequency. A link will only exist as long as two peers are within ear's shot. Although we do not yet have experimental data, we expect the average duration of an encounter to be quite small and not exceed several minutes or even seconds.
  In Gnutella connections are transient as well. However, the average time a Gnutella peer stays connected to the network is much longer and ranges from several minutes to several days. The reason that Gnutella peers, which are mainly computers at home, go offline is simply that they are shut down.
- Because Gnutella runs over the Internet, individuals can connect directly to someone who is geographically far away just as easily as with their immediate neighbor. Thus, the number of peers a host can reach is potentially very large. To limit traffic on the Gnutella network each message carries a Time-to-Life (TTL) indicator. Most Gnutella peers will reject any messages which have TTL's that are excessively high.
- In *Proem*, however, connections are only possible between geographically close peers. This is important because it guarantees that the number of reachable hosts is relatively small which in turn makes it feasible to use multicasting or broadcasting mechanisms provided by the underlying network.
- Gnutella's architecture is completely decentralized – with one important exception. When a Gnutella peer wants to join the network it needs to connect to a handful of other peers. These peers will become its neighbors and will be the ones forwarding messages on its behalf. The question is: how does a peer know which other peers are around and which of those should it chose as entry points into the network? Obviously, since peers can come online or go offline at any time, there is no way to know *a priori* which peers are available at any time. The solution employed by most (all?) Gnutella applications (like BearShare and LimeWire) is to use a central host cache that maintains a list of available hosts. This host cache is kept up-to-date by peers already connected to the network. In other words, peers must contact a central server before they can join the Gnutella network. This solution is valid and viable on the Internet, but not in ad hoc networks. *Proem* truly is decentralized and employs a discovery mechanism that continuously checks for newly arriving or disappearing peers.
- Finally, *Proem* features built-in security. Gnutella in contrast provides no security, but only pseudo-anonymity for its users: the only data that could identify users, namely IP addresses, are not propagated throughout the network.

5.3. *Proem* and JXTA

Sun Microsystems's JXTA [37,38] is an open-source peer-to-peer platform that aims at making building distributed processes easier. It provides a platform to perform the most basic functionality required by any peer-to-peer application: peer discovery and peer communication.

JXTA's approach is similar to *Proem*, but its focus is different and its scope is more narrow (the first version of *Proem* was released before JXTA was made public).

First, let's look at some of the similarities:

- Both JXTA and *Proem* are protocol-based, i.e. they define an application-level communication protocol.
- Both provide built-in security measures, although JXTA's is more elaborate at this point.
- Finally, both JXTA and *Proem* peer-to-peer platforms as opposed to specific applications.

The differences between JXTA and *Proem* lie in the following areas:

- *Proem* is designed for highly dynamic environments of ad hoc networks where connectivity and resource availabilities change constantly. JXTA strength is in peer-to-peer applications for semi-stable environments like the Internet.
- *Proem* supports the development of adaptable applications that are aware of and can react to changes in the environment. This is achieved by providing feedback to applications about the QOS of resources.

The major goal of *Proem* is to provide effective and high-level support for developers of adaptable peer-to-peer applications. *Proem*'s Peerlet Development Kit makes it especially simple to create new peer-to-peer and significantly reduces the time it takes to develop a peer application.

First International Conference on Peer-to-Peer Computing (P2P'01), 2001.

6. Conclusion and Future Research Directions

The combination of emerging mobile ad hoc networks and personal mobile devices enables the creation of mobile peer-to-peer applications for proximity-based collaboration.

Despite that fact that ad hoc networking and peer-to-peer computing deal with similar issues, namely discovery, routing and information dissemination, there is not much overlap in the research. Most research on ad hoc networks focuses on the lower layer of the protocol stack including the link layer, network layer and transport layer. On the other hand, current peer-to-peer systems are designed for an Internet-like network infrastructure in which stationary hosts are connected by high bandwidth links. The assumptions on which these peer-to-peer systems are built are no longer valid in dynamic ad hoc networking environments. The unique characteristics of such networks require highly adaptable peer-to-peer systems that can react to changes in connectivity and resource availability in a timely and ongoing manner. We view the *Proem* mobile peer-to-peer platform as a step towards an integration of both research areas.

The main goal of the *Proem* platform is to provide high-level support for mobile peer-to-peer application developers. The early experiences of using a prototypical implementation of *Proem* in an advanced Software Engineering course at the University of Oregon are encouraging. Preliminary results suggest that students were able to realize complex collaborative peer-to-peer applications with great success and within a short time frame. Among other applications, students developed several impromptu MP3 file-sharing systems based on the scenarios from Chapter 2.

Our future research will focus on the following areas:

First, we are working on a tighter integration of *Proem* with services provided by underlying ad hoc networks.

Second, we are in the process of specifying and implementing a security architecture for *Proem.* One of the focal points will be the development of a fully decentralized trust mechanisms using a public key infrastructure and the use of reputations. Work in this area has already begun [20].

7. References

[1] A. Alwan, R. Bagrodia, N. Bambos, M. Gerla, L. Kleinrock, J. Short, J. Villasenor. Adaptive Mobile Multimedia Networks. IEEE Personal Communications Magazine, 3(2), April 1996.
[2] Z.J. Haas. Panel report on Ad hoc Networks. Mobile Computing and Communication Review, 2(1), 1997.
[3] D. C. Dryer, C. Eisbach, and W. S. Ark. At what cost pervasive? A social computing view of mobile computing systems. IBM Systems Journal, Vol 38, No. 4 - Pervasive Computing
[4] E. M. Hallowell, "The Human Moment at Work," Harvard Business Review, 58-66 (January-February 1999).
[5] Bin Yu and Munindar P. Singh, A Social Mechanism of Reputation Management in Electronic Communities, Proceedings of Fourth International Workshop on. Cooperative Information Agents, pages 154-165, 2000
[6] Holmquist, L.E., Falk, J. and Wigström, J. Supporting Group Awareness with Inter-Personal Awareness Devices. In Journal of Personal Technologies, Special Issue on Hand-Held CSCW, Springer Verlag, 1999
[7] Lenny Foner. Yenta: A Multi-Agent, Referral Based Matchmaking System, The First International Conference on Autonomous Agents (Agents '97), Marina del Rey, California, February '97. Yenta
[8] Iwatani, Y. Love: Japanese Style. In Wired News, 11 June 1998. http://www.wired.com/news/culture/story/12899.html
[9] Gerd Kortuem, Zary Segall, Thaddeus G. Cowan Thompson. Close Encounters: Supporting Mobile Cooperation Through Interchange of User Profiles. 1st International Conference on Handheld and Ubiquitous Computing (HUC), September 1999, Karlsruhe, Germany.
[10] Gerd Kortuem, Jay Schneider, Jim Suruda, Steve Fickas, Zary Segall. When Cyborgs Meet: Building Communities of Cooperating Wearable Agents. Proceedings Third International Symposium on Wearable Computers, 18-19 October, 1999, San Francisco, California.
[11] Department of Computer Science, University of Oregon, Advanced Software Engineering on Peer-to-Peer Computing, Spring 2001. Course home page: www.cs.uoregon.edu/classes/cis650
[12] (http://www.parthus.com/)
[13] Belotti, V. and Bly, S. (1996) Walking Away from the Desktop Computer: Distributed Collaboration and Mobility in a Product Design Team. In Proceedings of CSCW '96, ACM Press
[14] Bluetooth Consortium web site, http://www.bluetooth.org
[15] IEEE 802.15 Working Group for PANs, http://grouper.ieee.org/groups/802/15/index.html
[16] Phillip Carvey. BodyLAN. IEEE Circuits and Devices, V4 No 12 July 1996
[17] T. G. Zimmerman. Personal Area Networks: Near-field intrabody communication. IBM Systems Journal Vol. 35, No. 3&4, 1996
[18] Gregory Alan Bolcer, Michael Gorlick, Arthur S. Hitomi, Peter Kammer, Brian Morrow, Peyman Oreizy, Richard N. Taylor Peer-to-Peer Architectures and the Magi™ Open-Source Infrastructure http://www.peer-to-peerwg.org/downloads/collateral/proposals/ETI_peer-to-peer.pdf
[19] J. J. Kistler and M. Satyanarayanan. Disconnected Operation in the Coda File System. ACM Transactions on Computer Systems, 10(1):3, February 1992
[20] Jay Schneider, Gerd Kortuem, Joe Jager, Steve Fickas, Zary Segall. Disseminating Trust Information in Wearable Communities. 2nd International Symposium on Handheld and Ubitquitous Computing (HUC2K), Sept 25-27, 2000, Bristol, England.
[21] Wendi R.Heinzelman, Joanna Kulik and Hari Balakrishnan. Adaptive
protocols for information dissemination in wireless sensor networks. In
Proceedings of MOBICOM'99, Seattle, pp 174-185.

[22] Charles E.Perkins and Elizabeth Royer. Ad hoc on demand distance
vector routing. Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-
aodv-05.txt
[21] Charles E.Perkins and P.Bhagwat. Highly dynamic destination sequenced distance vector routing for mobile computers. In Proceedings of SIGCOMM'94, October 1994.
[23] Vincent D.Park and M.Scott Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. Inproceedings of INFO-COM'97, 1997.
[24] Sinha P., Sivakumar R. and Bhargavan,V. CEDAR: A core extraction distributed ad hoc routing algorithm. In Proceedings of INFOCOM'99.
[25] R.Sivakumar, B.Das and Bhargavan V., An improved spine based infrastructure for routing in ad hoc networks. In Proceedings of IEEE Sym-posium of Computers and Communications, 1998.
[27] Piyush Gupta and P.R.Kumar. A system and traffic dependent adaptive routing algorithm for ad hoc networks. In Proc. IEEE 36 the Conference on Decision and Control, pp 2375-2380, Sandiego 1997.
[28] Seungjoon Lee and Chongkwon Kim, Neighbor Supporting Ad Hoc Multicast Routing Protocol, MobiHOC 2000 Advance Program, August 11, 2000, Boston, Massachusetts, USA
[29] Linda Briesemeister and Gunter Hommel. Role-Based Multicast in Highly Mobile but Sparsely Connected Ad Hoc Networks. MobiHOC 2000 Advance Program, August 11, 2000, Boston, Massachusetts, USA
[30] Hu Zhou and Suresh Singh. Content Based Multicast (CBM) in Ad Hoc Networks. MobiHOC 2000 Advance Program, August 11, 2000, Boston, Massachusetts, USA
[31] Michael Nidd, Timeliness of Service Discovery in DEAPspace. Proceedings of the 2000 International Workshop on Parallel Processing
[32] Reto Hermann, Dirk Husemann, Michael Moser, Michael Nidd, Christian Rohner, Andreas Schade. DEAPspace - Transient ad hoc networking of pervasive Devices. Computer Networks 35 (2001) 411±428
[33] Theodoros Salonidis, Pravin Bhagwat, Leandros Tassiulas, and Richard LaMaire. Distributed Topology Construction of Bluetooth Personal Area Networks. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, April 2001.
[34] Gerd Kortuem. Proem Peer-to-Peer Platform: http://wearables.cs.uoregon.edu/proem
[35] Gnutella Protocol, version 0.4. http://dss.clip2.com/GnutellaProtocol04.pdf
[36] Gene Kan. Gnutella. In Andy Oram (ed.) Peer-to-Peer: Harnessing the Power of Disruptive Technologies. March 2001.
[37] JXTA web site: http://www.jxta.org/
[38] Project JXTA. Technical Specification 1.0. http://www.jxta.org/project/www/white_papers.html
[39] Kraut, R., Lundark, V., Kiesler, S., Mukopadhyay, T., & Scherlis, W. (1998). Internet paradox: A social technology that reduces social involvement and psychological well-being? American Psychologist, 53, p. 1017-1031
[40] Napster. http://www.napster.com/
[41] I. Clarke, O. Sandberg, B. Wiley, and T.W. Hong, Freenet: A Distributed Anonymous Information Storage and Retrieval System in Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, LNCS 2009, ed. by H. Federrath. Springer: New York (2001).
[42] Groove Networks. http://groove.net/