# An Incentive-Based System for Information Providers over Peer-to-Peer Mobile Ad-Hoc Networks

Jordi Castellà-Roca, Vanesa Daza, Josep Domingo-Ferrer, Jesús A. Manjón, Francesc Sebé, and Alexandre Viejo

Rovira i Virgili University
UNESCO Chair in Data Privacy
Department of Computer Engineering and Mathematics
Av. Països Catalans, 26, E-43007 Tarragona, Catalonia
{jordi.castella,vanesa.daza,josep.domingo,jesus.manjon}@urv.cat
{francesc.sebe,alexandre.viejo}@urv.cat

**Abstract.** An architecture for a peer-to-peer mobile ad-hoc network offering distributed information provision is presented. Any user can volunteer to become an information server (a server-user). Volunteering implies devoting some of the user's computational resources (storage, bandwidth, processing power) to serving information. An incentive scheme is proposed to encourage end-users to become server-users. The latter are rewarded proportionally to the number of end-user queries served. The proposed architecture is specified as a protocol suite taking security and privacy aspects into account. Details are given on an implementation completed on a WiFi ad-hoc network for the specific case of a distributed tourist information service.

**Keywords:** Peer-to-peer networks, Mobile ad-hoc networks, Distributed information servers, Security and privacy.

## 1 Introduction

The new-generation mobile devices (*e.g.* cell phones, PDAs ...) are enabled with wireless communications technologies like IEEE 802.11b WiFi, which paves the way to a broad range of services based on ad-hoc networks, *i.e.* spontaneous networks without an underlying infrastructure. Clearly, a wireless ad-hoc network spanning a local or a metropolitan area is an extremely flexible and powerful tool, allowing extensive peer-to-peer communication. Access points in these new wireless networks can be viewed as peers whose number and geographical span can be much larger than what is affordable with conventional peer-to-peer wired networks. From the topology standpoint, the ad-hoc model is particularly well suited to set up hop-by-hop communication to end-nodes connected to such wireless peer-to-peer networks.

In this paper, we propose an information system aiming to provide information *just in time* and *just in place* in a specific area. Specifically in our scenario, the system is deployed in a city within which a user, regardless of her location, can request information any time using her WiFi mobile device and its associated software. As justified above, we assume that access to information is made possible by a metropolitan ad-hoc network based on WiFi peers.

A typical application would be tourist information: a person touring a city can query the system to obtain a list of museums near her current location or some information about a given historical building she is currently visiting.

The system we have just sketched needs an architecture with several information servers replicating the knowledge, because a single information server could become a bottleneck due to downloads of large pieces of multimedia content (pictures, videoclips, etc.) or due to temporary connectivity loss caused by dynamic changes in the topology of the ad-hoc network. However, using several servers poses the problem of the cost associated to installing a large number of servers around the city plus the costs resulting from their subsequent administration.

Instead of incurring the expense of installing and maintaining many servers, we propose to trust individual users to offer services to the WiFi ad-hoc network like in classic peer-to-peer communities (*e.g.* BitTorrent, Gnutella, etc). To keep the network running efficiently, we suggest to properly motivate the users to devote some of their computing resources to providing those services.

## 1.1 Previous Work

Providing and exchanging information between users is a research topic heavily addressed in the literature. At the beginning, the information-sharing systems followed a client-server paradigm (*e.g.* Napster) with the bottleneck problems associated to this kind of structure. Afterwards, real peer-to-peer networks arose, and decentralized content-sharing applications appeared, which avoided the client-server related problems. Examples of this approach are BitTorrent or Gnutella.

Peer-to-peer applications are often deployed in traditional wired networks, although most of them could also work on wireless ad-hoc networks. In [6] some applications working on *mobile* ad-hoc networks are presented. Within the scope of mobile peer-to-peer networks, we can find the iClouds Project [5] which investigates several kinds of collaboration among mobile users using the hop-by-hop communication paradigm related to ad-hoc networks.

As stated in [16], the motivation of users to participate in the community is a crucial factor for the success of a peer-to-peer system. The authors of that paper describe various methods to motivate different kinds of users and describe the design of a peer-to-peer system called Comutella, which has been developed for supporting file and service sharing. [4] presents I-Help, a system devoted to sharing help between students where participation can be rewarded either in real dollars (for paying teaching assistants) or in marks improvement (for students). This system also uses hired knowledgeable persons, teaching assistants or lecturers, to be constantly on-line and to immediately answer any question. This kind of users can be considered knowledge providers in this system, and they are very similar to the information providers presented in our proposal.

Providing motivation to nodes, who offer services to end-users, implies the need of a secure way for collecting the rewards from the served users. Secure electronic payment is a profusely studied research topic. From electronic money to e-coupons [1], there are several electronic payment methods suitable for mobile devices. However, for the specific case of secure incentive-based schemes, the literature is rather scarce. [13] and [17] propose incentive-based schemes where the network nodes have an account and the content provider gives them credit depending on the information they have uploaded. The network nodes can use their credit to increase their download rate or change it for money. Nonetheless, these proposals are not designed for a mobile ad-hoc network, and the security is only focused toward the protection of the copyrighted content. Thus, the credit of the network node can be tampered with. In [10], the authors describe and analyze three different schemes aimed at giving proofs of service to providers in a hybrid environment where a server collaborates with a peer-to-peer network intended to mitigate server congestion. The authors compare the three schemes in terms of scalability, effectiveness and cost. Recently, [11] have proposed a lightweight and hash-chain-based micropayment scheme for ad-hoc networks. They compensate collaborative peers that sacrifice their resources to relay packets for others. They also use an offline trusted third party (the Private Key Generator), but following an ID-based approach instead of our PKI-based one.

In our proposal, we provide proofs of service to information providers using a scheme similar to the scenario of fair exchange with an offline trusted third party (TTP) described in [10]. However, our scenario is pretty different since we deal with a pure peer-to-peer community working through a mobile ad-hoc network.

## 1.2 Contribution and Plan of This Paper

In this paper we focus on an environment where some end-user nodes build a WiFi ad-hoc network and act as information providers. These nodes devote some of their computational resources (storage, bandwidth, processing power) to storing and serving information. In this way, when another user in the ad-hoc network requests some information, those nodes storing the requested information can supply it. Volunteering to become an information provider is rewarded depending to the amount of served information requests. Information is shared according to the peer-to-peer paradigm.

*Note.* We concentrate on the WiFi technology because it currently is the most common choice in mobile ad-hoc networks. However, the system described here is open to similar wireless technologies.

To perform information searches through the peer-to-peer network, we have implemented a catalog-based search engine. Our system makes use of an item catalog shared between information providers. Since updating a unique catalog distributed among all peers is a resource- and time-consuming task, this approach is best suited to a system where the items change rarely. We have taken as a case study a tourist information system: the information on accommodation, monuments, etc. is pretty stable. For items that change more dinamically, *e.g.*

films shown in theaters in a specific moment, a different search engine should be used. Anyway, the communication protocol in our system has been designed to be independent of the search engine used, which can be replaced if necessary.

Every time a server provides information to some user, it obtains a receipt allowing it to prove that it has performed the service. Periodically, the server contacts the main content provider (*i.e.* the main source of content, in our case study the city tourist office) to get paid according to the number of requests it has served. This is a way to encourage users to devote more resources to providing information. The more information a server stores, the more requests it will be able to serve, and thus, the more money it will receive for its service. A server located at a certain place will probably contain information that may interest nearby users.

Section 2 describes the architecture of the proposed system. Section 3 is a security and privacy analysis. Implementation details are given in Section 4. Section 5 is a conclusion.

## 2 System Architecture

In this section we present the system components: entities, messages exchanged between entities and protocols between entities. We note that the system architecture is similar to a multi-agent system.

### 2.1 Entities

The proposed system consists of the following entities:

– **Content Source (CS):** This is the entity offering the information service. In the aforementioned example about tourist information, this entity may be the tourist office of a city holding information of particular interest for residents or visitors. Some examples could be:
  - Information on historical landmarks, including short multimedia videos, audio streams and digital documents on them.
  - Schedule of cultural and leisure activities, like cinema or theater, including trailer viewing options.
  - Information about restaurants: opening hours, menus, prices.
  - Location of services: police stations, hospitals, pharmacies.

– **Users:** Users whose devices form the ad-hoc network. We distinguish two kinds of users:
  - Those that query the system when they need information. Normally, they use a mobile device and request information through the ad-hoc network. We refer to them as *end-users (EU)*.
  - Those that devote part of their computational resources to storing and serving some of the information supplied by the content source. These users not necessarily use a mobile device. They could store information in their desktop PC with an ad-hoc network interface. We refer to these users as *server-users (SU)*.

Our description assumes an underlying information sharing system (following the P2P paradigm) like those described in [3]. The data query launched by $A$ circulates through the ad-hoc network and reaches several server-users who have the requested data. Those users will send a positive answer to $A$. Then, $A$ chooses one of them, denoted by $B$. Next, $A$ requests the information from $B$ who will send it to $A$. Finally $A$ sends a receipt to $B$.

SUs store all the receipts they collect during a given period. At the end of that period, they send all the receipts to the CS in order to get paid according to the number of served requests. Upon getting the receipts, CS checks their validity. If everything is correct, CS pays to the user the money corresponding to the services provided.

### 2.2 Messages

In our proposal, we distinguish two types of communication:

- The first type follows a client-server paradigm and involves the content source CS. We have chosen this approach because the communication between CS and the other two entities (SU, EU) only occurs at very specific moments and is unlikely to cause a bottleneck.
- The second type of communication, the dialog between an EU and a SU, follows a P2P paradigm.

In both cases, the communication uses structured messages coded with XML. Messages consist of two parts. The first part contains the message itself, divided in two or three sections: message type, sender identifier (in the P2P environment) and message body. The second part contains the cryptographic data: the signature on the first part of the message, the algorithm used to calculate the signature and the digital certificate for the sender's public key. This structure allows the receiver to verify the validity of the message.

### 2.3 Protocols

In this section, we detail the different protocols used by the entities participating in the system:

- End-user registration
- Server-user registration
- Information request
- Server-user payment

The following notation is used in the rest of the paper:

- $P_{entity}, S_{entity}$: Asymmetric key pair of *entity*, where $P_{entity}$ is the public key and $S_{entity}$ is the private key.
- $S_{entity}[m]$: Digital signature of message $m$ by *entity*. By digital signature we refer to computing the hash value of message $m$ using a collision-free one-way hash function and encrypting this hash value using the private key of *entity*.

- $E_{entity}(m)$: Encryption of message $m$ under the public key of *entity*.
- $D_{entity}(c)$: Decryption of message $c$ under the private key of *entity*.
- $H(m)$: Hash value of message $m$ using a collision-free one-way hash function.
- $m_1||m_2$: Concatenation of messages $m_1$ and $m_2$.

**End-user registration.** To register as an end-user, a candidate user must contact the CS and install the necessary application software. *E.g.* in our tourist information case study we assume that there exist several places in the city (*e.g.* airport, railway station or tourist office) where a user can register. The end-user registration protocol is as follows:

**Protocol 1**

1. *The user does:*
   (a) *Obtain the following information from the CS:*
      - *Internet address from which to download the application software.*
      - *Validity period,* i.e. *time window during which the user will be allowed to use the system.*
      - *Access code to download and install the software. In our case study implementation the access code consists of 16 alphanumeric characters, for instance A3GZ-BB44-223G-AGDR.*

   (b) *Connect her device to the Internet and download the application software.*
   (c) *Install the application software.*
   (d) *Run Procedure 1 below and obtain the private key $S_{EU}$ in a PKCS#8 file [12], and a Certificate Signing Request (CSR).*
   (e) *Send the CSR to the Content Provider.*
2. *The CS does:*
   (a) *Issue the user's certificate using the CSR.*
   (b) *Add the issued certificate to the CS database.*
   (c) *Send the issued certificate to the user.*
3. *The user stores the following information in a PKCS#12 [12] file:*
   - *User private key $S_{EU}$.*
   - *User certificate.*
   - *CS certiticate.*

**Procedure 1**

1. *Generate a private/public RSA key pair [14].*
2. *Store the private key in a PKCS#8 file.*
3. *Generate a Certificate Signing Request (CSR). The file must use the PKCS# 10 [12] standard.*
4. *Return the PKCS#8 file and the CSR.*

**Server-user registration.** A user wishing to register as a server-user contacts the CS from whom she will receive a unique identifier and the software that will enable her to serve information. Afterwards, the user generates a private/public key pair and sends her public key and her identifier to the CS in order to get the corresponding certificate. Finally, the user indicates the desired information items and downloads them to her hard disk. More formally, the server-user registration protocol is as follows:

**Protocol 2**

1. *The user does:*
   (a) *Sign a contract with the CS specifying the user's rights and duties.*
   (b) *Send the user's bank data for future payments to CS. For confidentiality, these data are sent encrypted under the public key of CS.*
2. *CS does:*
   (a) *Generate a unique identifier $Id$.*
   (b) *Send to the user the unique identifier $Id$ and the software that will enable the user to serve information. For confidentiality, $Id$ is sent encrypted under the public key $P_{EU}$ (the candidate server-user is assumed to be already an end-user with a private/public key pair $(S_{EU}, P_{EU})$).*
3. *The user does:*
   (a) *Run Procedure 1 to obtain the private key $S_U$ in a PKCS#8 file, and a Certificate Signing Request (CSR).*
   (b) *Send the CSR to CS.*
4. *CS does:*
   (a) *Issue the user's certificate using the CSR.*
   (b) *Add the issued certificate to CS's database.*
   (c) *Send the issued certificate to the user.*
   (d) *Send the catalog information.*
5. *The user stores the following information in a PKCS#12 file:*
   – *User private key $S_U$.*
   – *User certificate.*
   – *CS certificate.*

**Information request.** When an end-user requests an information item, the query reaches several server-users. Among these, those holding the requested item return a positive acknowledgment. Then, the end-user downloads the requested information from a particular server-user selected among those which have sent positive acknowledgment. Finally, the end-user sends a receipt to the selected server-user. As we will see later on, the SU will use this receipt in order to claim the corresponding reward from the CS.

Now, we describe this protocol in more detail:

**Protocol 3**

1. *The end-user EU computes a request in order to obtain a specific information, where the request consists of the following data:*

   - *Description of the requested item, $I$.*
   - *Date and time of the request, $T_r$.*
   - *Digital signature of $I$ and $T_r$, $S1 = S_{EU}[I||T_r]$.*

   *This query spreads using any of the methods described in [3].*

2. *Each server-user SU who receives the query does:*
   (a) *Verify the digital signature $S1$ using EU's public key.*
   (b) *Search for the information.*
   (c) *If $I$ is in SU's database, reply to EU. The reply contains the following data:*
      - *User's request, $R_{EU} = I||T_r$.*
      - *Date and time of the answer, $T_a$.*
      - *Digital signature on $R_{EU}$ and $T_a$, that is, $S2 = S_{SU}[R_{EU}||T_a]$.*
3. *EU does:*
   (a) *Collect the replies from the SUs. Without loss of generality, assume that the set of SU replying to EU is $SU_1, SU_2, \cdots, SU_n$. See Section 3.3 below on the value of $n$.*
   (b) *Verify the digital signatures of the SUs, that is, $S2_1, S2_2, S2_3, \cdots, S2_n$ using the public keys of each SU.*
   (c) *Choose one server-user SU' $\in \{SU_1, SU_2, \cdots, SU_n\}$. This choice can be performed in a way to maximize privacy (see Section 3.3 below).*
   (d) *Send a request to SU' with the following data:*
      - *Description of the requested information, $I$.*
      - *Date and time of the request, $T_r$.*
      - *Identifier of the node this request is addressed to, $Id_{SU'}$.*
      - *Digital signature on $I$, $T_r$ and $Id_{SU'}$, that is, $S3 = S_{EU}[I||T_r||Id_{SU'}]$.*
4. *SU' does:*
   (a) *Verify the digital signature $S3$ using the public key $P_{EU}$.*
   (b) *Send the following message:*
      - *Description of the requested information, $I$.*
      - *Requested information, $Info$.*
      - *Date and time of the answer, $T_a$.*
      - *Digital signature of the $I$, $Info$ and $T_a$, that is, $S4 = S_{SU'}[I||Info||T_a]$.*
5. *EU does:*
   (a) *Verify the digital signature $S4$ using $P_{SU}$.*
   (b) *Check whether the received data correspond to the information requested.*
   (c) *If the check is OK, issue a receipt and send it to SU' with the following data:*
      - *Description where EU asserts that she has received the item described as I from SU'.*
      - *Date and time, $T$.*
      - *Identifier of SU', $Id_{SU'}$.*
      - *Digital signature on $I$, $T$, and $Id_{SU}$, that is, $S5 = S_{EU}[I||T||Id_{SU}]$.*
6. *SU' does:*
   - *Receive the receipt.*
   - *Verify $S5$ using $P_{EU}$*
   - *Store the receipt.*

**Server-user payment.** As previously described, server-users get a receipt every time they serve information. These receipts are stored. Once a large enough batch of receipts has been collected, a server-user contacts CS to get paid for the services provided. Note that sending receipts one at a time to CS would be very inefficient. The reason is that, since the reward for a single service is very low, the processing costs of such a payment would be too significant.

The protocol to redeem a batch of receipts is as follows:

**Protocol 4**

1. *SU sends the receipts to CS.*
2. *CS does:*
   (a) *Verify the digital signature of each receipt*
   (b) *Check for duplicated receipts*
   (c) *Compute the money that must be paid to the information node*
   (d) *Transfer the money to the bank account of SU*

## 3 Security and Privacy Analysis

Our communication protocols use different types of messages to be transmitted in each phase. Every exchanged message contains a plaintext part and a valid signature. The plaintext part contains the information transmitted between nodes and the signature provides *authentication*, *integrity* and *non-repudiation* to such messages.

### 3.1 Confidentiality

In principle, confidentiality is only implemented in the server-user registration protocol, when the user sends her bank data to CS and CS returns a unique identifier (Steps 1 and 2 of Protocol 2). The rest of messages are assumed to be non-confidential, which is plausible for most applications (*e.g.* tourist information). However, if confidentiality is required, it can be achieved by encrypting messages under the public key of the intended receiver.

### 3.2 Collusion Security

Collusion between end-users and server-users to obtain unlawful rewards is conceivable: some end-users perform a huge amount of information requests to certain server-users, and the latter then share with the former the rewards obtained from the CS.

A possible solution is to charge the end-users a small fee for enjoying the information service. This payment can be performed using offline electronic checks as stated in [2] or any micropayment system (*e.g.* PayWord, [15]).

However, one must acknowledge that collecting payment from the end-users can jeopardize the success of many applications, like the tourist information system. Therefore, a preferred countermeasure against user collusion is for the

CS to record and analyze the number of receipts submitted by the SUs and the number of receipts issued by the same EU. Since each receipt contains the exact time and date when it was issued, a limit on the number of requests that an EU can perform within a period of time can easily be enforced. The CS will not honor any receipts beyond those that can be issued by a certain user; furthermore, as soon as CS detects that an end-user has issued more receipts than allowed, CS alerts the SUs to stop serving any further request from that suspect SU. The SUs receive this alert when they synchronize resources with the CS or when they redeem their receipts. In this way, the effects of possible user collusions are tolerably mitigated.

## 3.3 Privacy

In any information service, end-user profiling is a real threat. Indeed, information providers can keep track of the requests submitted by end-users, with a view to investigating their tastes, preferences, locations, etc. This is clearly a potential privacy violation.

In a conventional information service where end-users get information directly from a single information provider, one often assumes that information provider to be trusted or at least not to be interested in violating the privacy of end-users. At any rate, if there ever were any provable violation, the information provider would be liable and could be charged accordingly.

In a peer-to-peer mobile ad-hoc information service, the privacy problem is much more serious. End-users obtain information through server-users who are occasional information relayers and cannot be trusted to the same extent as to privacy preservation.

End-user privacy can be significantly increased by using an alias when registering as an end-user and by properly tuning Protocol 3:

- When the end-user application detects that there are server-users among the $n$ replying to Step 3a who already replied to more than $p$ requests from the same end-user in the past ($p$ is a privacy parameter), the application warns the end-user of a potential privacy problem. The end-user has two choices: either move to a different area where she will find different server-users or to go ahead and jeopardize her privacy.
- In Step 3c, a wise policy is for the end-user application to choose the server-user which has replied to least requests to the end-user in the past.

Of course, we are assuming that the server-user application has not been tampered with, so that: i) it replies when the server-user hears a request for an information item it holds; ii) it forgets about requests for information items the server-user does not hold.

In the presence of malicious server-users, a combination of the following two strategies can be useful:

- Use short validity periods for end-users, which will force end-users to frequently re-register under a new alias.

– Avoid issuing many information request from the same place, which should be easy for a roaming end-user (*e.g.* tourist visiting a city). Moving to another area is a way to get rid from the current server-users, both the legitimate and the malicious ones.

## 4 Implementation

We have implemented a functional prototype that demonstrates the main capabilities of the system previously described. Although not all funtionalities are currently implemented as of this writing, the system core is fully operational already.

We have chosen Java [7] as a language for this implementation, due to its portability to different architectures that need to interoperate, an essential feature in our scenario. We have used JXTA [8] to establish the peer-to-peer communications, although there are other options like Gnutella or Bamboo. Finally, to implement the client-server paradigm (for communication with the CS), we use the remote method invocation protocol (RMI) associated to the Java technology.

Specifically, the content source and the server-users have been implemented with Java SE version 5.0 and they use databases containing touristic resources that run on MySQL version 5.0. The CS component executes a secure web server, which acts as an access point for new server-users willing to join the system. In the future we will allow the end-users to enter the system through the secure web server too. SU applications make use of JXTASE libraries, version 2.4, to communicate with the other entities through the peer-to-peer ad-hoc network.

The end-user application is implemented in a WiFi enabled HP IPAQ series hx2700. We have installed the J9 Java Virtual Machine from IBM [9] that works with Connected Device Configuration 1.1 and is PersonalJava 1.2 compliant. This PersonalJava version follows the Java 1.1.8 version specifications. The JXTA version that works on the PDA is the JXME 2.1.3.

### 4.1 Server-User Registration

Once a user runs the SU application for the first time (Protocol 2), she must introduce an alias and a password of her choice, and enter the CS URL as well. After this process, she generates a key pair RSA (1024 bits) and a Certificate Signing Request (CSR) to be sent to the CS. This entity sends back the CSR properly certified. Finally, the user generates the PKCS#12 file and a configuration file, which will be used in the following executions and can be modified any moment from the SU graphical interface. This configuration file contains the user's alias, the URL of the CS and the absolute path to the PKCS#12 file which contains the user's private key, the user's certificate and the certificate of the CS.

After this initial phase, the SU application displays a first window with a menu showing some options to be chosen by the user. Then, she can pick the items from a catalog provided by the CS. The selected items will be stored in the user's computer and will be used to provide information to the end-users. After serving some information requests from end-users, the SU can check the number of receipts stored and the money earned. She can decide to redeem the stored receipts.

### 4.2 End-User Registration

We assume that, in Step 1 of Protocol 1, the candidate end-user receives her registration information (Internet address from which to download, validity period, access code) offline, *e.g.* in a tourist office. After this (remaining steps of Protocol 1), the user goes on-line and, much like a server-user, the end-user submits her Certificate Signing Request to CS, who will return it properly certified. At the end of this transaction, the end-user will obtain a PKCS#12 file and a configuration file will be created. Upon completion of this installation process, the end-user gets her keys and is ready to use the information service.

In the tourist information service example, once the user reaches the vicinity of a monument, she may be interested in obtaining some information about it. For that, she must chose the corresponding item in the catalog, using the EU application in PDA and send the request through the ad-hoc peer-to-peer network to SU applications present in the area. In a few seconds, a list of SU nodes offering the requested item will appear on the end-user's screen. The user then chooses one of the possible providers and the selected node sends the desired information.

## 5 Conclusions and Future Work

An architecture for a peer-to-peer mobile ad-hoc network offering distributed information provision has been presented. The novelty of our proposal lies on the incentive scheme to encourage users to become distributed information servers. Our implementation demonstrates the feasibility and the usefulness of our approach.

The most challenging part of our system is security and, especially, privacy. Indeed, allowing any user to behave as a server is not without risks. While authentication, integrity, non-repudiation and even confidentiality can be guaranteed to a large extent, preserving the privacy of end-users is trickier because they can be profiled by server-users. We have proposed and implemented some basic privacy-preserving countermeasures, but we feel that privacy in peer-to-peer ad-hoc networks is a far-reaching problem that definitely deserves further research. In the specific case of our system, a challenging issue is to design privacy countermeasures which are minimally inconvenient for the end-user, *e.g.* that do not require very short validity periods or constant roaming.

## Disclaimer and Acknowledgments

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization.

This work was partly supported by the Spanish Ministry of Education through project SEG2004-04352-C04-01 ”PROPRIETAS” and by the Government of Catalonia under grant 2005 SGR 00446.

## References

1. Blundo, C., Cimato, S., De Bonis, A.: Secure E-Coupons. Electronic Commerce Research Journal 5(1), 117–139 (2005)
2. Chaum, D., Den Boer, B., Van Heyst, E., Mjolsnes, S., Steenbeek, A.: Efficient offline electronic checks (extended abstract). In: Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques on Advances in Cryptology. Springer, New York (November 1990)
3. Ding, G., Bhargava, B.: Peer-to-peer file-sharing over mobile ad hoc networks. In: First International Workshop on Mobile Peer-to-Peer Computing, p. 1000 (2004)
4. Greer, J., McCalla, G., Vassileva, J., Deters, R., Bull, S., Kettel, L.: Lessons learned in deploying a multi-agent learning support system: the I-help experience. In: Proceedings of AI in Education AIED’2001, pp. 410–421 (2001)
5. Heinemann, A., Kangasharju, J., Lyardet, F., Mühlhäuser, M.: iClouds - Peer-to-peer information sharing in mobile environments. In: Kosch, H., Böszörményi, L., Hellwagner, H. (eds.) Euro-Par 2003. LNCS, vol. 2790, pp. 1038–1045. Springer, Heidelberg (2003)
6. Heinemann, A., Mühlhäuser, M.: Spontaneous collaboration in mobile peer-to-peer networks. In: Steinmetz, R., Wehrle, K. (eds.) Peer-to-Peer Systems and Applications. LNCS, vol. 3485, pp. 419–433. Springer, Heidelberg (2005)
7. Sun Microsystems, JAVA Programming language: `http://java.sun.com`
8. Sun Microsystems, Project JXTA: `http://www.jxta.org/`
9. IBM, WebSphere Everyplace Micro Environment: `http://www-128.ibm.com/`
10. Li, J., Kang, X.: Proof of Service in a Hybrid P2P Environment. In: Chen, G., Pan, Y., Guo, M., Lu, J. (eds.) Parallel and Distributed Processing and Applications - ISPA 2005 Workshops. LNCS, vol. 3759, pp. 64–73. Springer, Heidelberg (2005)
11. Pan, J., Cai, L., Shen, X., Mark, J.W.: Identity-based secure collaboration in wireless ad hoc networks. Computer Networks 51(3), 853–865
12. Public-Key Cryptography Standards (PKCS): `http://www.rsasecurity.com/rsalabs/node.asp?id=2124`
13. Rajasekaran, H.: An incentive based distribution system for DRM protected content using peer-to-peer networks. In: 1st International Conference on Automated Production of Cross Media Content for Multi-channel Distribution, pp. 150–156 (2005)
14. Rivest, R.L., Shamir, A., Adleman, L.M.: A method of Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
15. Rivest, R.L., Shamir, A.: PayWord and MicroMint: two simple micropayment schemes. In: Lomas, M. (ed.) Security Protocols. LNCS, vol. 1189, pp. 69–87. Springer, Heidelberg (1997)
16. Vassileva, J.: Motivating Participation in Peer to Peer Communities. In: Petta, P., Tolksdorf, R., Zambonelli, F. (eds.) ESAW 2002. LNCS (LNAI), vol. 2577, pp. 141–155. Springer, Heidelberg (2003)
17. Vishnumurthy, V., Chandrakumar, S., Sirer, E.: Karma: A secure economic framework for p2p resource sharing. In: Workshop on Economics of Peer-to-Peer Systems (2003)