2005 International Conference on Distributed Computing and Internet Technology, Bhubaneswar, India, December 22-24 2005, LNCS, Springer.

# Off-line Micro-payment System for Content Sharing in P2P Networks

Xiaoling Dai[1] and John Grundy[2, 3]

Department of Mathematics and Computing Science
The University of the South Pacific, Laucala Campus, Suva, Fiji[1]
dai_s@usp.ac.fj

Department of Electrical and Computer Engineering[2] and Department of Computer Science[3]
University of Auckland, Private Bag 92019, Auckland, New Zealand
john-g@cs.auckland.ac.nz

**Abstract.** Micro-payment systems have the potential to provide non-intrusive, high-volume and low-cost pay-as-you-use services for a wide variety of web-based applications. We propose an extension, P2P-NetPay, a micro-payment protocol characterized by off-line processing, suitable for peer-to-peer network services sharing. Our approach provides high performance and security using one-way hashing functions for e-coin encryption. In our P2P-NetPay protocol, each peer's transaction does not involve any broker and double spending is detected during the redeeming transaction. We describe the motivation for P2P-NetPay and describe three transactions of the P2P-NetPay protocol in detail to illustrate the approach. We then discuss future research on this protocol.

## 1 Introduction

A peer-to-peer architecture is a type of network in which each workstation generally has equivalent capabilities and responsibilities. Peer-to-peer networks are often simpler than client-server but they usually do not offer the same performance under heavy loads. A P2P network relies on computing power at the ends of a connection rather than from within the network or dedicated servers.

A Central Indexing Server (CIS) is sometimes used to index all users who are currently online. This server does not host any content itself but provides support for peers to locate content from other peers. Queries on the index server are used to find other connected users with content required and when a match is found the central server will tell clients where to find the requested content. Users can then choose a result from the search query and their peer will attempt to establish a connection with the computer hosting the information requested.

In a P2P CIS system, peers cooperate to search the relevant information in the system. However, in some peer-to-peer systems, peers often cannot find suitable services since many peers choose to decline requests from others for security or other reasons.

This problem characterises the "free rider" problem in P2P – users who search and use content but don't allow others to use their client for services.

A natural approach to control free riding is to introduce a payment protocol into CIS systems, in which each peer has to pay for the services it receives from others, e.g., [11]. However, traditional heavy weight macro-payment protocols are unsuitable in this domain of high-volume, low cost-per-item searches and information downloads. We propose an off-line micro-payment protocol, P2P-NetPay, to address this common *free-rider* problem. Our protocol allows peers to buy "E-coins", worth very small amounts of money, from a broker and spend these E-coins at various peers to pay for large numbers of searches and digital files of small value each. P2P-NetPay shifts the communication traffic bottleneck from a broker and distributes it among the peers by using transferable E-coin Touchstones and Indexes, much in the same way as micro-payment in client-server network applications [2].

In this paper, we briefly describe Ppay protocol and the NetPay micro-payment protocol with the three kinds of e-wallets in the client-server networks. We then proposal an off-line micro-payment protocol called P2P-NetPay to control free riding problem in peer-to-peer networks. We conclude with an outline of our further plans for research and development in this area.

## 2 Motivation

While there is an emergence of new technologies and applications to enable users to exchange content over P2P networks, the success of such systems depend on users' willingness to share computing resources and exchange content. One of the first and most well-known P2P file-sharing systems, Napster [15], has attracted great public attention for the P2P systems as well as at one time having tens of millions of users. Napster was designed to help its users to trade music files, however, P2P applications could exchange any kind of digital document. The file sharing is free by peers in most current P2P systems. Since peers do not benefit from serving files to others, many users decline to provide services to others. In fact, a recent study of the Gnutella network found that more than 70% of its peers have made no contribution to the P2P system [12]. This emerging phenomenon of "selfish" individuals in P2P systems has been widely studied, and is known as the *free-rider* problem. There is a trend towards charging peers for access CIS or charging for every file download in order for peers make direct profit from files they upload [12].

One payment model for peer-to-peer systems is a subscription-based method. In this approach the CIS charges a membership fee per time period as a way of recovering the overhead involved in running its services. The subscription charge does have an impact on peers' decisions about whether or not to participate in the P2P network. However, the contribution to the system of such a fee is irrelevant to their efforts to maximize utility when they have made this decision. Most importantly, the fact that subscription fees are unrelated to peers' behavior implies that they still give rise to a free rider problem.

In order to encourage peers to balance what they take from the system with what they contribute to the system we present an on-line micro-payment approach used to charge peers for every download and to reward peers for every upload [11]. For each registered peer the Central Indexing Server tracks the number of files downloaded and the number of files uploaded during the time period. Each time a file is successfully exchanged between two peers, the server increments the download count of the peer who downloaded the file and the upload count of the peer who uploaded it. Observe that in such a model server involves all such transfers and it's an on-line, client-server brokered system.

A point-based mechanism that is similar to the micro-payment mechanisms discussed above is introduced in [11]. In order to make use of an internal currency, peers are allowed to buy points either with money or with contributions to the network, but peers are not allowed to convert points back into money. Since peers cannot "cash out" their points, the mechanism must allow them to maintain a balance from one time period to the next. This system also uses an on-line mechanism. There are a number of micro-payment systems for client-server networks in various stages of development from proposals in the academic literature to systems in commercial use [1], [7], [8], [9], [10]. Micro-payment systems can be used to support payment of vendors from customers in client-server networks. In peer-to-peer applications, there is not any clear distinction between vendors and customers. There are simply peers which can be vendors or customers or both. Ppay is an example off-line micro-payment system in peer-to-peer networks [14].

## 3. Ppay: a peer-to-peer micro-payment protocol

The Ppay micro-payment system was proposed by Yang and Garcia-Molina [14]. The concept of floating and self-managed currency is introduced, so each peer's transaction does not involve any broker. The coins can float from one peer to another peer and the owner of a given coin manages the currency itself, except when it is created or cashed. Fig. 1 shows key Ppay interactions.

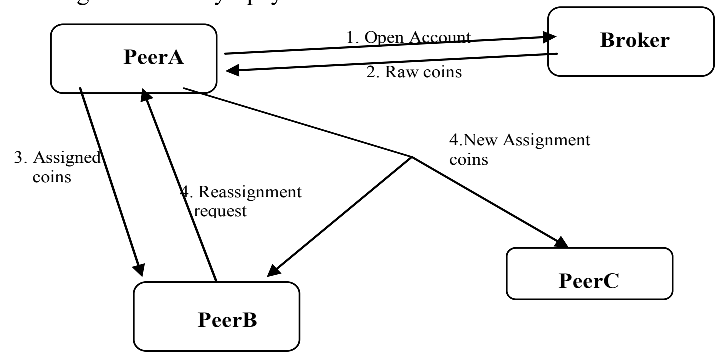

**Fig. 1. Ppay protocol participant interactions [based on 14].**

- *Open an account with a broker:* The PeerA opens an account with the broker scrip at start of the day and the broker returns initial raw coins to the PeerA. Now PeerA is the owner of the coins.

- *Assigned coins*: when PeerA wants to purchase an item or a service from PeerB, PeerA will send the assigned coins to PeerB. Now PeerB is the holder of the coins. PeerB can decide to cash them or re-assign them to another peer (PeerC).
- *Reassignment request*: If PeerB wants to re-assign the coins, PeerB sends the reassignment request to PeerA
- *New assigned coins*: after receiving the request, PeerA PeerA processes and sends the new reassignment to PeerB and PeerC.

The problem with this approach is that PeerA can be down when PeerB wants to reassign his own coins. A peer can be down with almost 97% probability, on average, when a payment must be made, so a *downtime protocol* is presented in Ppay [14]. In the downtime protocol, the Broker generates the newly assigned coins and sends the assigned coins back to PeerA when PeerA comes back online in order to detect frauds committed. Key drawback with downtime protocol includes: the broker must be on-line when the peers wish to re-assign the coins and the broker has to check when peers came back on-line. Due to the high percentage of off-line periods for a peer, the broker' load significantly grows up.

In order to avoid the above problems, a concept of *layered coins* is used in the Ppay protocol. The layered coins are used to float the coins from one peer to another. Each layer represents a reassignment request and the broker and the owner of the coins can peel off all the layers to obtain all the necessary proofs. The layered coins introduce the delay of the fraud detection and the floating coins growing in size.

## 4. NetPay in Client-server Networks

We developed a protocol called NetPay that provides a secure, cheap, widely available, and debit-based protocol for an off-line micro-payment system [1]. We have developed NetPay-based systems for client-server broker, vendor and customer networks [3], [4]. We have also designed three kinds of "e-wallets" to manage e-coins in our client-server NetPay systems [3], [4], [5]. In one model the E-wallet is hosted by vendor servers and is passed from vendor to vendor as the customer moves from one site to another. The second is a client-side application resident on the client's PC. The third is a hybrid that caches E-coins in a web browser cookie for debiting as the customer spends at a site.

The client-side e-wallet is an application running on the client PC that holds e-coin information. Customers can buy article content using the client-side e-wallet at different sites without the need to log in after the e-wallet application is downloaded to their PC. Their e-coins are resident on their own PC and so access to them is never lost due to network outages to one vendor. The e-coin debiting time is slower for a client-side e-wallet than the server-side e-wallet due to the extra communication between vendor application server and customer PC's e-wallet application. In a client-side e-wallet NetPay system, a Touchstone and an Index (T&I) of a customer's e-wallet are passed from the broker to each vendor. We designed that the broker application server communicates with vendor application servers to get the T&I to verify e-coins. The vendor application servers also communicate with another vendor application server to pass the T&I, without use of the broker. The main problem with this approach is that a vendor system cannot get the T&I if a previous vendor system down.

## 5 P2P-NetPay Protocol in Peer-to-peer Networks

Based on the client-side e-wallet NetPay protocol, we propose an adaption to a P2P-NeyPay protocol that is suitable for P2P-based network environments. Our P2P-NetPay protocol uses touchstones that are signed by the broker and an e-coin index signed by requesting peers. The signed touchstone is used by a supplying peer to verify the electronic currency – paywords, and signed Index is used to prevent double spending from peers and to resolve disputes between peers. In this section, we describe the key transactions in P2P-NetPay protocol in P2P networks.

In this section, the details of a peer-to-peer micro-payment NetPay model are discussed. Consider a trading community consisting of Peers and Broker (B). The CIS system can also act as a Broker in the P2P networks. Assume that the broker is honest and is trusted by the peers. The peers may be or may not be honest. The peers open accounts and deposit funds with the broker. The payment only involves Peers and Broker is responsible for the registration of peers and for crediting the peer's account and debiting the peer's account. In a P2P-NetPay system, there are three transactions which are requesting peer-broker, requesting peer – supplying peer1, and peer - broker transactions. How the NetPay protocol works in each transaction will now be described in more detail. We adopt the following notations:

**IDa** --- pseudonymous identity of any party A in the trade community issued by the broker.

**PK-a** --- A's public key.

**SK-a** --- A's digital signature.

**{x}SK-a** --- x signed by A.

**{x}PK-a** --- x is encrypted by A's public key.

**{x}SAK- a** --- x signed by A using A’s asymmetric key.

There are a number of cryptography and micro-payment terminologies used in the P2P-NetPay micro-payment protocol. The details of these terminologies are given as follows

1. ***One-way Hash Function -*** the one-way hash function MD5 used in the NetPay implementation is an algorithm that has the two properties. It seems impossible to give an example of hash function used in hash chain in a form of normal functions in mathematics. The difficulties include:
   - The value of a mathematical function is a real or complex number (a data value for hash function);
   - It is always possible to compute the set $X = \left\{ x \middle| x = h^{-1}(y) \right\}$ for a given $y$ for a mathematical function $h$ (not satisfying the two properties of the hash function).

2. ***Payword Chain*** – A “payword chain” is generated by using a one way hash function. Suppose we want to generate a payword chain which contains ten “paywords”. We need randomly pick a payword seed $W_{11}$ and then compute a payword chain by repeatedly hashing

   $W_{10} = h(W_{11})$,     $W_9 = h(W_{10})$,
   **……,**
   $W_1 = h(W_2)$,     $W_0 = h(W_1)$

where h(.) is a hash function such as MD5 and $W_0$ is called the root for the chain. The MD5 (Message Digest) algorithm is one of the series of messages in hash algorithms and involves appending a length field to a message and padding it up to a multiple of 512 bit blocks. This means that every payword $W_i$ is stored as a 32 length string in a database. A payword chain is going to be used to represent a set of E-coins in the P2P-NetPay system.

## 5.1 Transaction 1: Requesting Peer1 – Broker

Before a Requesting Peer1 (RP1) asks for service from the Supplying Peer2 (SP2), she has to register and send an integer n (M1), the number of paywords in a payword chain the RP1 applied for, to the broker (Fig. 2). The broker completes two actions:

- Debits money from the account of RP1 and creates a payword chain $W_0$, $W_1$, $W_2,\ldots,W_n$, $W_{n+1}$ which satisfy $W_i = h(W_{i+1})$, where i = n, …, 0. (here h(.) is a one way hash function). Root $W_0$ is used to verify the validity of the paywords $W_1$, $W_2$, …, $W_n$ by peers and the broker. Seed $W_{n+1}$ is kept by the broker to be used to prevent the peer1 from overspending and forging paywords in that chain. The peer1 only receives $ID_e$ (e-coin ID) and paywords $W_1$, $W_2,\ldots,W_n$ that are encrypted by RP1’s public key from the broker (M2) as shown in Fig. 2.

  $$M2 = \{ ID_e, W_1, W_2, \ldots ,W_n \}_{PK\text{-}RP1}$$

  The broker computes the touchstone for the payword chain:

  $$M3 = T = \{IDe, W_0\}_{SK\text{-}broker}$$

  and sends it to RP1.

- Save IDe, $W_{0,}$ $W_{n+1}$, and amount to the broker database.

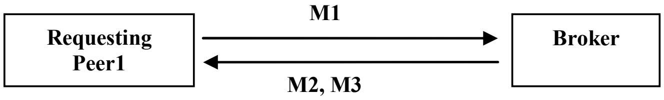

**Fig. 2. Requesting Peer buys e-coins transaction**

For example, the requesting peer sends n=50 to the broker who generates the IDe=1 and payword chain $\{W_{0,}$ $W_1$, $W_2$, … $,W_{50}$, $W_{51}\}$. The RP1’s e-wallet is thus

{IDe, $W_1$, $W_2$, … ,$W_{50}$} and T. The broker saves IDe, $W_{0,}$ $W_{n+1,}$ and 50 to its database.

The requesting peer - broker transaction guarantees no overspending and forging. The broker selects the seed $W_{n+1}$ to create the payword chain which satisfy $W_n = h(W_{n+1})$, $W_{n-1} = h(W_n)$, …, $W_1 = h(W_2)$, $W_0 = h(W_1)$ and keep the seed $W_{n+1}$ secretly. It is impossible to forge the paywords in that chain by peers and attackers, since they do not have the seed $W_{n+1}$, i.e. it is impossible to generate other paywords in a chain by knowing some of them in the chain since h() is a truly one-way hash function [16].

## 5.2 Transaction 2: Requesting Peer1 – Supplying Peer2

The following sequence of messages describes a transaction between a requesting peer and a supplying peer1 in the course of a download of information from Peer1 to Peer2. The requesting peer1 (RP1) and supplying peer2 (SP2) needs to agree on the amount that RP1 pays.

When a RP1 find a desired file that belongs to SP2, the RP1's e-wallet sends message M4 and T to the SP2.

$$M4 = \{ IDe, paywords\}$$

where paywords = {$W_1$, $W_2$, …, $W_m$}. For example, to make a 2cs (m=2) payment, the peer1 sends the paywords $W_1$, $W_2$: Paywords = {$W_1$, $W_2$} to the SP2. The RP1 also signs the following transmission message:

$$Index = \{ID_e, i\}_{SAK\text{-}RP1}$$

and transmits them to SP2, where i is the index of the last payword SP2 received. The Index is used to prevent double spending from RP1 and may be used for disputes between the peers. The touchstone authorises SP2 to verify the paywords using root $W_0$ and redeems the paywords with the broker as shown in Fig. 3.

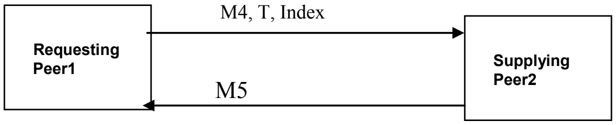

**Fig. 3. Requesting peer buys digital file transaction**

The paywords are verified by taking the hash of the paywords in the order $W_1$ first, then $W_2$, and so on. The paywords $W_1$ and $W_2$ are valid if the hash matches the root of the chain ($W_0$) in the touchstone ($h(W_1)=W_0$, $h(h(W_2))=W_0$). This works because the hash function with the property $W_{i-1}= h(W_i)$ (i = 1, 2, …, n) and SP2 gets $W_0$ from the broker.

On the other hand, it is hard for SP2 to create $W_1$ even though he knows $W_0$ since the generation of a value that would hash to $W_0$ is computationally infeasible due to the nature of the one-way hash function [16]. For the same reason, it is also hard for an attacker to generate valid paywords in the chain even if he knows $W_0$ or some paywords except for the seed $W_{n+1}$ [16], [17].

If the paywords are valid, they will be stored for a later offline transaction with the broker. The RP1 downloads the file from SP2 (M5). Multiple payments can be charged against the length of the payword chain, until the payword chain is fully spent or the RP1 no longer requires files with other peers [16].

When the RP1 wishes to purchase files with supplying peer2, RP1 repeats the transaction2 with M4, M5, and M6.

For example, the RP1 requests to buy a file which costs 3cs. The RP1 sends M4 = {IDe, $W_1W_2W_3$}, T and signed Index to the SP2. The current state of the RP1 e-wallet database is shown in Fig. 4.

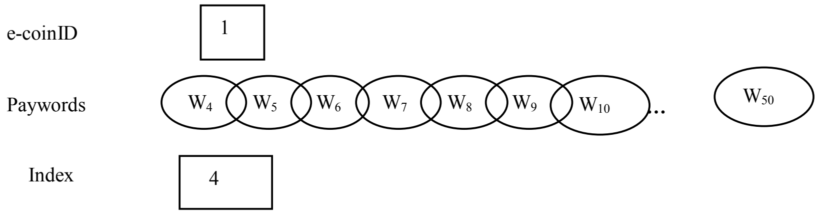

**Fig. 4. Example of RP1's e-wallet database after first transaction**

The SP2 gets T from the RP1 and then verifies $W_1$, $W_2$, $W_3$ by using $W_0$ such as $h(W_1)=W_0$, $h(h(W_2))=W_{0,}$ $h(h(h(W_3)))=W_{0.}$ If the paywords are valid, the RP1 download the file from SP2 (M6) and saves IDe=1, index=4, price=3, $W_{0,}$ paywords= $W_1W_2W_3$ in a redeem database as shown in Fig. 5.

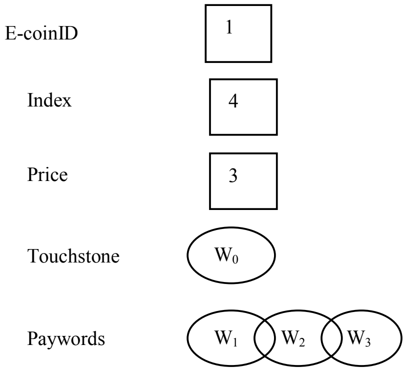

**Fig. 5. Example of redeem database after first transaction**

The RP1 continues to buy another file which costs 2cs, the RP1 sends M4 = {IDe, $W_4W_5$}, T and Index=6 to the SP2. The current state of the RP1's e-wallet database is shown in Fig. 6.

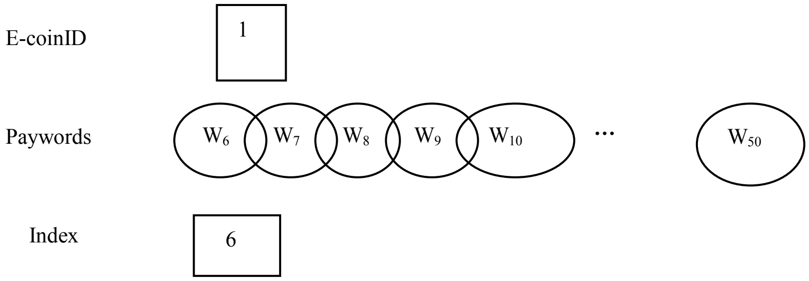

**Fig. 6. Example of the e-wallet database after second transaction**

The SP2 verifies $W_4$, $W_5$ by using $W_0$ obtained before. If the paywords are valid, RP1 downloads the file from SP2 (M6) and saves IDe, index=6, price=2, $W_0$, paywords= $W_4W_5$ to the redeem database as shown in Fig. 7.

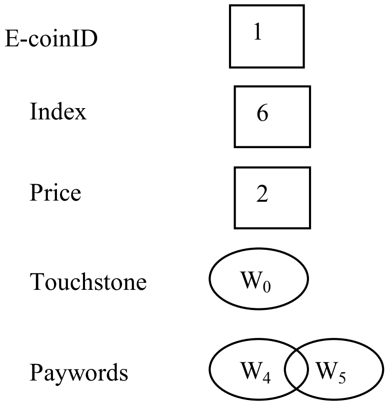

**Fig. 7. Example of redeem database after second transaction**

When PP1 wishes to make a purchase at a different peer RP3, he/she sends M4 , T (where T = $\{IDe, W_0\}_{\text{SK-broker}}$ ) and Index to the SP3. RP1 can download the file if the paywords are valid.

## 5.3 Transaction 3: Peer – Broker Offline Redeem Processing

At the end of each day (or another suitable period), for each payword chain, all supplying peers need to send all paywords that they received from requesting peers to the broker and redeem them for real money. To do this a supplying peer must aggregate the paywords by each e-coinID and send the following message to the broker

$$M6 = \{IDp, IDe, Payments\}$$

The broker needs to verify each payword received from the peer by performing hashes on it and counting the amount of paywords. If all the paywords are valid, the broker deposits the amount to the peer's account, and then sends an acknowledgement

M7 = {Balance Statement of the peer's account}

to the supplying peer as shown in Fig. 8.

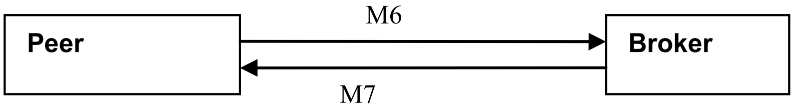

**Fig. 8. Peer-redeem transaction**

The protocol is credit based. There is no protection mechanism to prevent a peer from double spending. Double spending is detected at the time of the redeeming process. The broker checks the peer's paywords whether they are already in the database or not. Once double spending is detected, the malicious peers are penalized by terminating to use P2P-NetPay and access the peer-to-peer networks.

For example, at the end of each day, SP2 aggregates two payments as shown in Fig. 5 and Fig. 7 for IDe=1 and sends $ID_{SP2}$ and IDe along with 6 (index), 5 (price), $W_1W_2$ … $W_5$ (paywords) (M7) shown as Fig. 9 to the broker. The broker verifies the paywords ($W_1W_2$ … $W_5$) by using $W_{0,}$ index (6) and price (5). If they are valid, the broker deposits 5cs to the SP2's account and send the balance to the SP2 (M8).

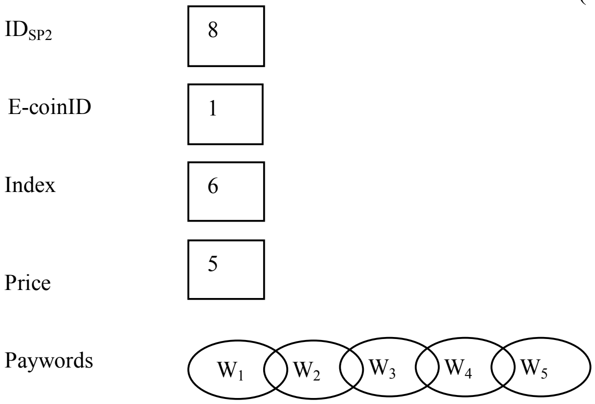

**Fig. 9. SP2 aggregates two payments**

## 6 Discussion

As we discussed in Section 3, existing P2P micro-payment protocols like Ppay have a down time protocol which is almost an on-line micro-payment system. The use of layered coins of Ppay protocol introduces the delay of the fraud detection and the floating coins growing in size. We have presented a real off-line and credit-based protocol suitable for micropayments in peer-to-peer networks. The protocol prevents peers from double spending using after-fact policy and any internal and external adversaries from forging, so it satisfies the requirements of security that a micropayment system should have. The protocol is economical since it does not involve public-key operations per purchase. Netpay can easily handle more transactions between peers. The major thrust of Netpay protocol is that it shifts the communication traffic bottleneck from the broker and distributes it among the peers, thus placing some processing burden on the requesting peer when a requesting peer wishes to purchase from a supplying peer. Work is underway to implement a trading community on the proposal protocol to evaluate its feasibility using our client-server based NetPay e-wallets and e-coin purchase/redemption as a prototype infrastructure.

## References

1. Dai, X. and Lo, B.: NetPay – An Efficient Protocol for Micropayments on the WWW. Fifth Australian World Wide Web Conference, Australia (1999)
2. Dai X. and Grundy J.: Architecture for a Component-based, Plug-in Micro-payment System, In Proceedings of the Fifth Asia Pacific Web Conference, LNCS 2642, Springer, April 2003, pp. 251-262.
3. Dai, X., Grundy, J.: Architecture of a Micro-Payment System for Thin-Client Web Applications. In Proceedings of the 2002 International Conference on Internet Computing, Las Vegas, CSREA Press, June 24-27, 444--450
4. Dai, X. and Grundy J.: Customer Perception of a Thin-client Micro-payment System Issues and Experiences, Journal of End User Computing, 15(4), pp 62-77, (2003).
5. Dai X. and Grundy J., Three Kinds of E-wallets for a NetPay Micro-payment System, The Fifth International Conference on Web Information Systems Engineering, November 22-24, 2004, Brisbane, Australia. Lecture notes in Computer Science 3306, pp. 66 - 77
6. Gabber, E. and Silberschatz, A.: Agora: A Minimal Distributed Protocol for Electronic Commerce, Proceedings of the Second USENIX Workshop on Electronic Commerce, *O*akland, California, November 18-21, 1996, pp. 223-232
7. Gabber, E. and Silberschatz, A.: "Micro Payment Transfer Protocol (MPTP) Version 0.1". *W3C Working Draft*, 1995. http://www.w3.org/pub/WWW/TR/WD-mptp
8. Herzberg, A. and Yochai, H. : Mini-pay: Charging per Click on the Web, 1996 http://www.ibm.net.il/ibm_il/int-lab/mpay
9. Manasse, M.: The Millicent Protocols for Electronic Commerce. First USENIX Workshop on Electronic Commerce. New York (1995)
10. Rivest, R. and Shamir, A.: PayWord and MicroMint: Two Simple Micropayment Schemes. Proceedings of 1996 International Workshop on Security Protocols, Lecture Notes in Computer Science, Vol. 1189. Springer (1997) 69—87

11. Golle, P., Leylton-Brown, K. and Mironov, I.: Incentives for sharing in peer-to-peer networks. In Proc. of Second workshop on Electronic Commerce (WELCOM'01), Heidelberg, Germany, November, 2001.
12. Shneidman, J. and Parkes, D.: Rationality and self-interest in peer-to-peer networks. In Proc. of 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), Berkeley, CA, USA, February 2003.
13. Eytan Adar and Bernardo Huberman. Free riding on Gnutella. First Monday, 5(10), 2000.
14. Yang, B. and Garcia-Molina, H.: Ppay: micropayments for peer-to-peer systems. In prooc. Of the 10th ACM conference on computer and communication security, pages 300-310. ACM press, 2003.
15. The Napster home page, http://www.napster.com/
16. Rivest, R.: "The MD5 Message-Digest Algorithm". RFC 1321, Internet Activities Board, 1992.
17. Menezes, A. J., Oorschot , P. C. and Vanstone, S. A.: Handbook of Applied Cryptography. New York, 1997.