第 XXII 部

Integrated Distributed Environment with Overlay Network

第 22 部
Integrated Distributed Environment with Overlay Network

第 1 章 Activities of IDEON WG in FY2008

1.1 Introduction

IDEON (Integrated Distributed Environment with Overlay Network) is a working group of researchers trying to approach realization of the integrated distributed environment (IDE) through construction of overlay networks (ON).

IDEON focuses on research, development and operation of overlay networks as an infrastructure to realize free and creative rendezvous, location and routing. IDEON members are encouraged to actually live on the infrastructure to verify their designs.

Research topics of IDEON include, but not limited to, the following:

- Application-layer multicast
- Operable distributed hash tables
- Self-sustained trust management for distributed autonomous systems

For the detailed plans of specific projects, please refer to the web pages of each project.

- `http://member.wide.ad.jp/wg/ideon/?en%2FProjects` (English)
- `http://member.wide.ad.jp/wg/ideon/?ja%2FProjects` (Japanese)

Activities in IDEON can be outlined as follows:

$\{understand \rightarrow build \rightarrow try\}* \rightarrow deploy \mid live$

where '$*$' denotes repetition and '$|$' denotes concurrency.

Currently, all our development projects are in *deploy* phase.

1.2 Summary of Activities

In this year, we have organized The Third International Workshop on Dependable and Sustainable Peer-to-Peer Systems (DAS-P2P 2008) in July[1].

We have investigated the design of a *storage-standard currency*, and a design for distributed backup and migration of distributed hash tables.

1.3 Glossary

Table 1.1 shows the glossary for IDEON.

第 2 章 DAS-P2P 2008 International Workshop

2.1 Overview

IDEON has organized the third annual international workshop on dependable and sustainable

Table 1.1. Glossary for IDEON

Free	Having no restriction whatsoever as to with which peers one can communicate.
Creative	Being able to select a set of peers according to one's objectives, requirements, needs and contexts so that communication becomes most valuable for the participants.
Rendezvous	To identify such a peer.
Location	To locate such a peer in the overlay networks by the acquired identifier.
Routing	To deliver a message to such a peer on the acquired location.
Overlay Network	An application-specific virtual network of peers over the IP network to realize rendezvous, location and routing over an appropriate abstraction of entities.

1 We will also organize DAS-P2P 2009 international workshop in January 2009, which will be reported in our report next year.

peer-to-peer systems, this time in conjunction with The 2008 International Symposium on Applications and the Internet (SAINT 2008), which took place in Turku, Finland, in July–August, 2008.

For the detail of the workshop, please refer to the following URL:

- `http://das-p2p.wide.ad.jp/2008/`

The Third International Workshop on Dependable and Sustainable Peer-to-Peer Systems (DAS-P2P 2008) is a workshop which focuses on dependability and sustainability of peer-to-peer (P2P) systems, with respect to their designs, operations, applications and social impacts.

The goal of this workshop was to share experiences, insights and new ideas, and set forth research agendas and suggestive future directions by collaborations among researchers with different disciplines and with similar interests toward dependability and sustainability.

The following table shows the status of the workshop.

Number of submitted papers	13
Number of accepted papers	8
Number of presentations*	9
Number of participants	30∼35

* 8 high quality papers and one invited talk

2.2 Topics

We had three sessions in the workshop in which we had discussions on the following topics:

- Session 1: Identity, Reputation and Trust
  - Accountable anonymity/pseudonymity, reputation, recommendation
- Session 2: Reliability and Self-Organization
  - Reliability in volunteer computing, bootstrapping, self-organizing maintenance under constant churn, distributed backup and migration
- Session 3: Fairness and Social Responsibility
  - Fair storage sharing
  - Talking Circle: Towards P2P Social Responsibility
    - How do we make P2P systems trustable?
    - How can P2P system contribute to our society?

2.3 Future Workshops

We have already planned the next workshop to be held in January, 2009.

- DAS-P2P 2009 (`http://das-p2p.wide.ad.jp/2009/`) (January 13, 2009)
  - in Las Vegas, Nevada, USA
  - in conjunction with the 6th Annual IEEE Consumer Communications & Networking Conference (CCNC 2009)

At the closing of SAINT 2008, we have suggested the following topics for next SAINT conferences, which we would also like to discuss in our subsequent DAS-P2P workshops.

- Social Responsibility of Internet Applications
  - Can use of sophisticated networks lead to less total energy consumption in our society?
  - How do we support local production, local consumption economy?
  - Will ICT contribute to make our civilization sustainable?

第 3 章 Local Production, Local Consumption Storage Economics for Peer-to-Peer Systems

Abstract

Autonomy of peer-to-peer (P2P) systems requires some form of economies. Forwardable *storage claims* of Samsara[36] can define a common value in computer networks, and possibly be a *commodity money* in P2P systems. However, without managing how far the claims can be forwarded, they would not form an efficient, dependable and sustainable economic chain.

We propose to use *i*-WAT[154] tickets to represent storage claims to form *drafts in real terms* so that a claim can be dynamically replaced by its equivalent in the vicinity, allowing the accesses to

the storage to be fast and robust.

An incentive analysis and a simulation are made to discover the extent to which the proposed methodology is effective.

3.1 Introduction

3.1.1 Needs for peer-to-peer economies

Designs of P2P systems are characterized by their usage of overlay networks such that participants can potentially take symmetrical roles. This implies distribution of authorities, not only preventing introduction of single points of failure, but also possibly assuring the level of autonomy for self-organization, where any subsystem can spontaneously start, maintain itself, or recover from its failures.

To make use of under-utilized computing resources in such an environment, since the resources are distributed over autonomous entities, exchanging needs to be performed in an incentive-compatible way: the coordination must be accomplished by collection of selfish behaviors.

3.1.2 Challenges

There are following challenges:

1. We need an economy to avoid the tragedy of the commons[61], in which the shared resources are depleted through over-exploitation because participants are motivated to maximize their own benefits while the costs of exploitation are equally distributed among them; we need an economy that does not tolerate those participants who do not pay appropriate costs in return for their benefits.
2. This economy needs to be autonomous, efficient and tolerant of failures; the economy should not interfere with the benefits of P2P designs.
3. This requires an exchange medium conforming to the principle of local production, local consumption (LPLC)[159] — what consumed locally are to be produced locally, and only when they are unavailable, they are to be conveyed from the nearest producers.

3.1.3 Contributions of this work

Recent studies in P2P economics have shown that a decentralized digital medium of exchange is possible by representing debts: Samsara[36] and *i*-WAT[154] are two examples. However, Samsara alone cannot implement LPLC, and *i*-WAT does not have a credible way to bind debts with resources.

This work is an attempt to design a new medium of exchange for P2P systems, called *Storage-Standard Currency* (*SSC*), based on the concept of *storage claims* (incompressible placeholders) adopted from Samsara, and representing them in the form of digital tickets adopted from *i*-WAT. The medium conforms to the principle of LPLC.

Possible applications of this work include any P2P activities, such as file sharing, data aggregation, data dissemination, bandwidth sharing and distributed computing.

3.1.4 Organization of this report

The rest of this report is organized as follows. Section 3.2 describes some historical media of exchange to help readers understand the core concepts of this work, and gives overviews of Samsara and *i*-WAT on which the proposed economy is based. Section 3.3 describes the *commodity money* realizable by Samsara, and *drafts in real terms* realizable by *i*-WAT with the storage claim as the guarantee of the debt. Section 3.4 gives an incentive analysis and simulation results of the proposed designs. Section 3.5 describes other existing designs of P2P economic systems. Section 3.6 describes some future work. Finally, section 3.7 reviews the outcome of this work in relations to the original challenges, and assesses the success of the designs.

3.2 Background

3.2.1 Historical media of exchange

3.2.1.1 Commodity money

Autonomous economy, in which participants can generate exchange media themselves, is not a new concept.

For example, in old Japan, rice was a common value being used as a medium of exchange (*commodity money*), whose value comes from the commodity itself. Anyone (any farmer) with facilities to produce rice could generate a medium of exchange themselves.

Implication to P2P designs A core concept of this work is that a remote storage space can be seen as a common value in a P2P system, which can become a medium of exchange as a commodity money. A storage claim similar to the one proposed by Samsara is used as a secure representation of an amount of a remote storage space.

3.2.1.2 Drafts in real terms

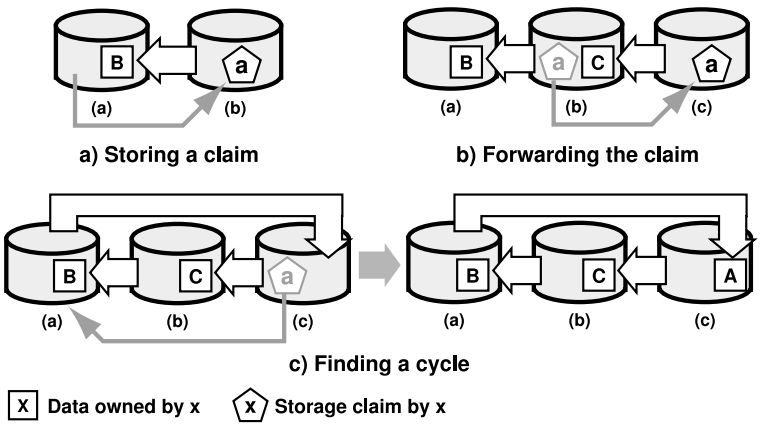
When rice was a common value, promise of a specific amount of rice of a specific grade could become a guarantee believable enough to work as money. This is the concept of *drafts in real terms*, or drafts exchangeable with a common value other than money, which was an exchange medium used in Japan in around 12th∼13th centuries. If we are to receive rice of the same grade in exchange for such a medium, the nearer the producer is, the cheaper the transactional cost is. Therefore this medium implies the concept of LPLC. A draft represented a common value as a medium of exchange, so that it contained the concept of commodity money as its subset.

Implication to P2P designs Another core concept of this work is that an interchangeable promise of providing a remote storage space can be seen as a draft in real terms in a P2P system. An *i*-WAT ticket is used as a representation of a draft whose promise is expressed in the form of a storage claim.

3.2.2 Samsara

Samsara is a fair P2P storage infrastructure proposed by Cox and Noble in 2003.

Design of Samsara (Fig. 3.1) is based on an observation that symmetry is rarely found in P2P exchanges, and that symmetry can be manufactured. In Samsara, each peer that requests storage of another must agree to hold a *storage claim*,



a) Data B is stored in (a) for (b). Peers (a) and (b) create symmetry in their transactional relation by placing a storage claim a on (b).
b) Instead of doubling the required storage, the claim a is forwarded to (c) in return for storing its data C in (b).
c) The claim a disappears when forwarded to the original claimer (a).

Fig. 3.1. Overview of Samsara

or an incompressible placeholder, in proportion to their consumption. Claims can be forwarded along the chain of nodes that requests storage of another, eliminating themselves when cycles are found.

Queries are occasionally made by claimers based on the *claim seed* that was used for generating the claim. To respond to the queries, a node needs to be physically storing the claim, thus guaranteeing that the space is reserved.

3.2.2.1 Storage claims

In Samsara, the claimer can compute the claim from the following information: a secret passphrase P, a private, symmetric key K, and a location L in the claimer's storage space (for every claim to be unique). The triple $\langle P, K, L\rangle$ is called a *claim seed* henceforth.

A claim is generated from consecutive 20-byte SHA1 hash values of the passphrase P concatenated with an offset calculated from location L, encrypted by the symmetric key K.

For example, let $h_i = SHA1(P, i)$ and a claim is 512-byte long. Then the ith claim C_i is computed by concatenating the first 25 hash values from the offset from L and the first 12 bytes of the 26th hash value as follows:

$$C_i = \{h_j, h_{j+1}, \ldots, h_{j+24}, \\ h_{j+25}[0], \ldots, h_{j+25}[11]\}K$$

where $j = i \times 26$.

The current holder of a claim cannot compute it without knowing the claim seed, and can present it only when it is physically stored in their storage.

The claimer infrequently queries for the claim to verify that the space is reserved for them. The response to the query can be a single SHA1 hash value by the query algorithm described in [36].

3.2.2.2 Characteristics

Some characteristics of Samsara are as follows.

Autonomy Any peer can spontaneously create claims to be stored by others, or participate in a claim forwarding chain.

Compatibility Storage is a common computing resource so that this scheme works in any part of the world with any computers with sufficient computing capabilities.

Security Upon forwarding a claim, a peer may want to strategically withhold a copy of the claim, because the peer is responsible for answering queries, and the node to which it has forwarded the claim may fail at any time. This means that consumption of storage is duplicated as the claim forwarding chain is lengthened.

3.2.3 *i*-WAT

i-WAT is a P2P currency system proposed by the authors in 2003.

Design of *i*-WAT (Fig. 3.2) is seen as electronization of the WAT System[188], a real-life barter currency using a physical sheet of paper resembling a bill of exchange (*WAT ticket*).

In *i*-WAT, messages signed in OpenPGP are used for implementing transfers of an electronic version of WAT ticket (*i-WAT ticket*). An *i*-WAT ticket contains a unique number, amount of debt and public key user IDs of the drawer, users and recipients. Endorsements are realized by nesting PGP signatures over canonical XML expressions.

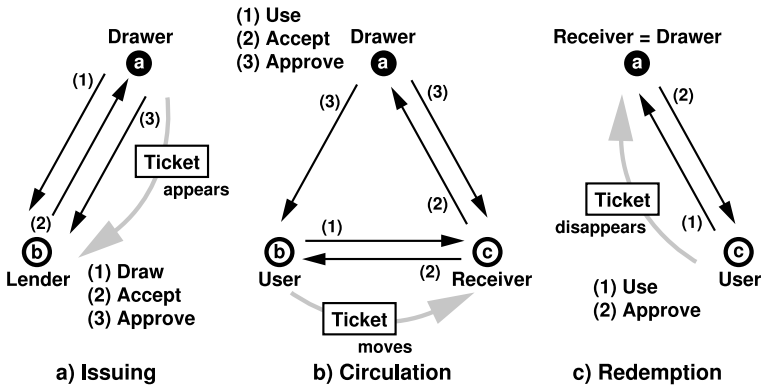
Upon electronization, the following changes have been made from the state machine of a WAT ticket:

1. Trades need to be asynchronously performed. Intermediate states, such as waiting for acceptance or approval, are introduced, and
2. Double-spending needs to be prohibited. The drawer is made responsible for guaranteeing that the circulating ticket is not a fraud. This means that every trade has to be approved by the drawer of the involved ticket.

The semantics of this design and the trust model of *i*-WAT are discussed in detail in [155].

A life cycle of an *i*-WAT ticket involves three stages of trading:

Issuing A *drawer* issues an *i*-WAT ticket by clarifying the public key user ID of the provider



a) Drawer (a) issues a ticket in return for a service from (b).
b) When the ticket is transferred to a third party (c), the transaction needs to be approved by the issuer (a) to detect double-spending.
c) The ticket disappears when it returns to the original issuer (a).

Fig. 3.2. Overview of *i*-WAT

(*lender*) of the goods or service, the amount of debt, the present date, and the drawer's signature. The drawer sends the ticket to the lender, and in return obtains some goods or service.

Circulation The peer to whom the *i*-WAT ticket was sent can become a *user*, and use it for another trading. To do so, the user clarifies the public key user ID of the recipient. The recipient will become a new user, repeating which the *i*-WAT ticket circulates among people.

Redemption The *i*-WAT ticket is invalidated when it returns to the drawer.

3.2.3.1 Tickets

The value of an *i*-WAT ticket is expressed as a tetrad $\langle V_0, V_m, V_x, f\rangle$ presented by the drawer, where V_0 is the face value (initial value) of the ticket, V_m is the minimum value, V_x is the maximum value, and $f(t)$ is the differentiation (derivative) of a function of time $F(t)$. V_m/V_x are set to be $\bot/\top$ respectively if those values are not applicable.

The effective value V_t of a ticket at time t is given by the following equation:

$$V_t = \min\left(\max\left(\int_0^t f(t)\,dt + V_0, V_m\right), V_x\right)$$

This allows the value of a ticket to vary over time, limited by some minimum/maximum values. Typically, it holds that either $f(t) = 0$ for all t (*regular* ticket), $f(t) < 0$ for all t (*reduction* ticket) or $f(t) > 0$ for all t (*multiplication* ticket).

The incentive mechanism for *reduction* and *multiplication* tickets have been discussed in [158] and [156], respectively.

3.2.3.2 Characteristics

Some characteristics of *i*-WAT are as follows.

Autonomy Anyone can spontaneously become a member of *i*-WAT if they follow the above protocol.

Compatibility An *i*-WAT ticket is compatible with any other *i*-WAT tickets in the world, so that the currency system is operable globally, as long as the drawer can be credited.

Extensibility The protocol illustrated in Fig. 3.2 defines the *i*-WAT version of the WAT Core, the essence of the WAT System. An extended part can be defined for a new currency based on the WAT/*i*-WAT, stating, for example, the region, group and duration in which the tickets are usable, as well as the unit in which the debt is quantified.

Security In case the drawer fails to meet their promise on the ticket, the lender assumes the responsibility for the debt. If the lender fails, the next user takes over. The responsibility follows the chain of endorsements (*security rule*). The longer the chain is, the more firmly backed up the ticket is.

3.3 Design

3.3.1 Preconditions

The following preconditions are assumed:

1. Every storage claim expires at a certain time, defined upon its creation. The reserved storage space can store data until the specified time.

   This is to balance the value of storage with other values being exchanged in a P2P system, such as bandwidth, which can be provided only temporarily.
2. The system is built upon a proximity-based overlay network with embedded rendezvous and reputation systems.

   A peer can determine what to purchase or exchange, by searching nearby entities satisfying required specifications (*rendezvous*). For such searches to work, grading entities in a distributed manner and guaranteeing its correctness are required (*reputation*).

3.3.2 Storage-Commodity Money (SCM)

3.3.2.1 Overview

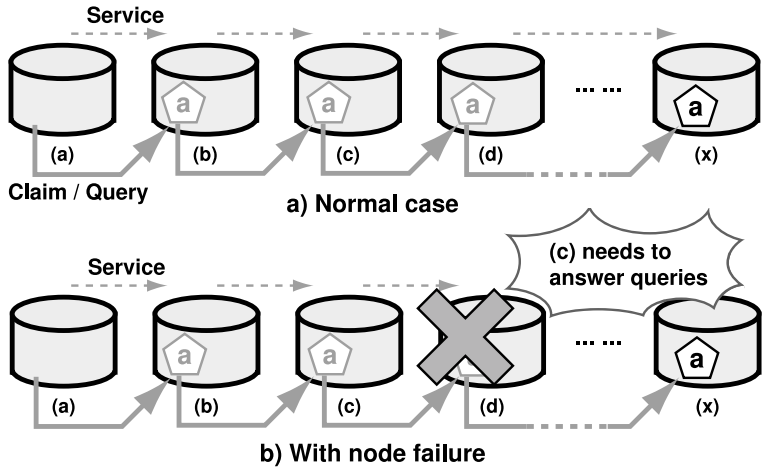
Samsara's storage claims can be forwarded in exchange with services other than storage, making them commodity money in the context of P2P (Fig. 3.3).

This commodity money has a unique property of service and the medium moving in the same direction. This has an interesting consequence; double-spending is naturally avoided because the medium of exchange is held by the producer of a service instead of a consumer when exchanging is taking place. The consumer cannot *double-receive* the medium to receive the service without paying the cost.

3.3.2.2 Drawbacks

Despite the benefit, this commodity money does not satisfy the principle of LPLC, as illustrated in case a) of Fig. 3.3. The available remote storage is likely to be departed from the vicinity of the original claimer as the claim keeps getting forwarded.

Efficiency of the storage space is also a problem, as illustrated in case b) of Fig. 3.3. Because the



a) The remote storage is likely to be departed from the vicinity of the original claimer (a) as the claim a keeps getting forwarded.
b) The strategy for the peers (b)∼(x) is to keep a copy of the claim a in their storage in case some node in the chain fails.

Fig. 3.3. Storage-Commodity Money (SCM)

node to which the claim has been forwarded may disappear from the system at any time, a peer needs to strategically withhold a copy of the claim it has forwarded; unused space is replicated as the claim forwarding chain is lengthened, unless a cycle is formed.

Another problem is how we induce formation of a cycle. A cycle is formed when a claim holder provides some service to the original claimer (typically the storage space it has been reserving), but the claimer should only want to receive the service if it is beneficial for itself. We have already seen that as the claim forwarding chain is lengthened, the storage space is likely to be further away from the claimer; there are contradicting demands (to be analyzed in section 3.4.1.1).

3.3.3 Storage-Standard Currency (SSC)

3.3.3.1 Overview

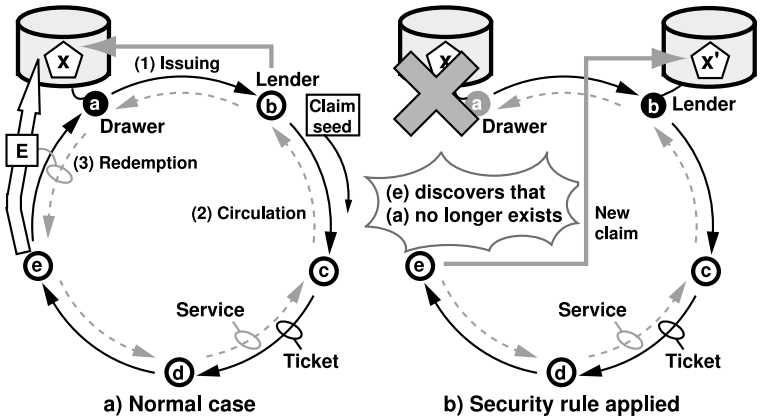
i-WAT can implement drafts in real terms in combination with storage claims, as illustrated in Fig. 3.4.

By issuing an *i*-WAT ticket guaranteed by a storage claim, a peer in a P2P system can purchase a service from another. Those tickets can be forwarded along the chain of nodes in exchange with services, eliminating themselves when they return to their issuers.

Upon circulation, the claim seed $\langle P, K, L \rangle$ corresponding to the storage claim needs to be forwarded along with the ticket (the seed must not be disclosed to the drawer unless the ticket is going to be redeemed) as illustrated in case a) of Fig. 3.4. Otherwise, there will be two inter-related problems:

1. If the drawer and lender collude, it will be possible for them to create an empty claim and make a ticket out of it.
2. Later participants will not be able to verify that the storage space is reserved by the claim.

The security rule of *i*-WAT allows the storage claim to be backed up, as illustrated in case b) of Fig. 3.4. If the drawer has been found disappeared, the lender must take over the claim. The lender does not need to have a copy of the claim prior to the take over. It must not, because the lender knows the claim seed for the original claim, and it can compute the claim on the fly without physically holding it in its storage. Upon the take over, a new claim must be generated from another claim seed.



a) Upon (a) issuing a ticket, the lender (b) requests the drawer (a) to store a storage claim x. Circulation of the ticket involves circulation of the claim seed for x so that receivers of the ticket can query the drawer (a) for the claim x. Upon redemption, the user (e) can have the drawer (a) store its data E.

b) If the drawer (a) fails, the current owner (e) of the ticket requests the lender (b) to store a new storage claim x' compatible with x in place of the drawer (a).

Fig. 3.4. Storage-Standard Currency (SSC)

3.3.3.2 Satisfaction of LPLC

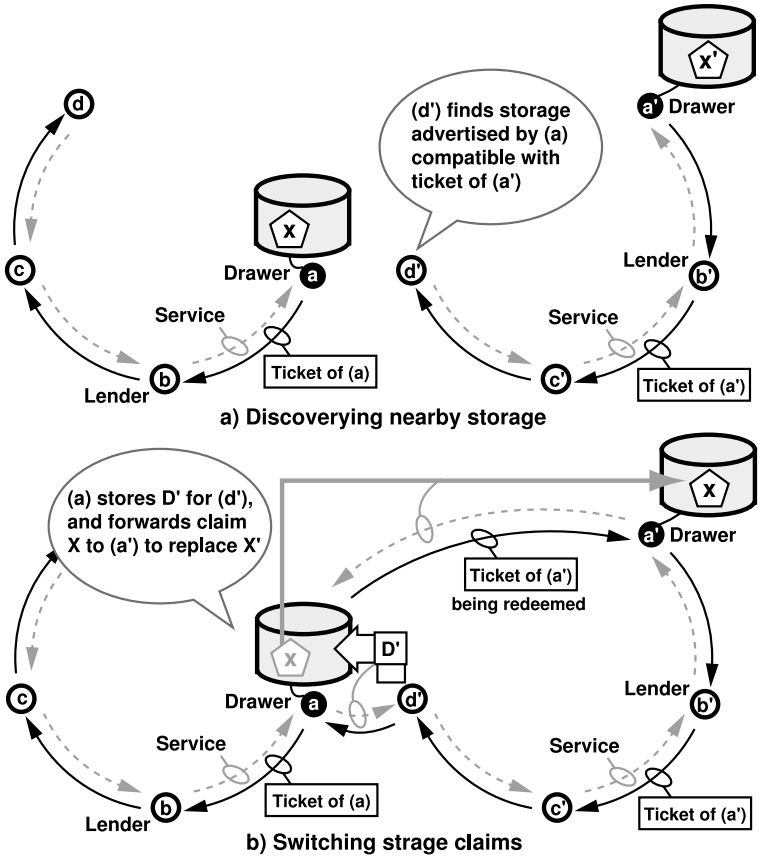
Every drawer (therefore the claim holder) advertises their storage capability that they can lend to others to induce redemption of their debts. The peers in need for a remote storage space listen to the advertisements. When they find one in the vicinity of their location in the logistical network, and if the space is compatible with what is claimable by the ticket they have, they try to rent the space by the ticket.

Fig. 3.5 shows how the principle of LPLC is satisfied with this implementation of drafts in real terms.

In the figure, the ticket is received by the storage provider (a), who transacts with (a′) for redemption of the ticket. Then (a) forwards the claim x it holds to (a′), which replaces the claim x' with x. When this is complete, (a) starts to advertise the forwarded storage claim in (a′).

Because the claim is now backed up by the security rule until the lease expires, (a) may be able to safely discard the copy of the claim, depending on the i-WAT chain length and the participants' reputations.

If, for example, there is a new node (c″) with a ticket of (a″) finding the advertisement of x attractive, and wants to rent the space, (a) would request (a′) to hold data C'' and receive



a) Drawers of tickets, (a) and (a′), advertise their remote storage services to induce redemption of their debts. A holder (d′) of the ticket of (a′) discovers a provider (a) of a remote storage service compatible with the promise on the ticket in its vicinity.

b) Data D' is stored in (a) for (d′) in return for the ticket of (a′). The ticket and the claim x are forwarded to the drawer (a′), and (a) now advertises the storage claim x in (a′). Note that even if this is repeated, the length of the claim forwarding chain is at most one, because the claim x will just be redirected to another peer.

Fig. 3.5. Assuring LPLC

regenerated claim from (d) (the current owner of the ticket of (a)), and forward it to (a″) in return for the ticket (a) has obtained.

Through repeating this kind of operations, the length of the claim forwarding chain remains at most one, avoiding to inherit the problems of the commodity money previously described.

3.3.3.3 Improvement of storage claims

The design of a storage claim as proposed by Samsara has two problems: 1) passing the claim seed to a receiver of a ticket can become a vulnerability as it means that secret information is passed to others (a problem specific to SSC), and 2) the query algorithm can be used for verification of stored claims only and not for verification of the stored data (a problem applied generally to applications of the storage claim).

These problems can be solved by making a more lightweight protocol for generating and verifying claims, so that a new claim can be generated every time the claimer changes, and making the protocol generally applicable for verification of remote data.

We have devised the following protocol:

1. A storage claim c is a random sequence of data.
2. The claimer calculates secure hash values of n number of randomly chosen blocks of size s in c. The claimer keeps the triplets $\langle h_i, L_i, s\rangle$ where $0 \leq i < n$, h_i is the ith calculated hash value and L_i is the corresponding location in c, and discards c after sending its copy to the claim holder.
3. The claimer occasionally queries the claim holder for the hash value h_x for randomly chosen x where $0 \leq x < n$ by providing L_x and s. The claim holder can answer the query only if it physically stores c.

This protocol is also applicable for verifying any remotely stored data, if we replace c with the data.

3.4 Analysis

3.4.1 Safety and risks

3.4.1.1 Safety and risks of SCM

We assess the risk with respect to participating in the creation and circulation of commodity money by storage claims. The risk takes the form of failing to forward the storage claim after providing a service. This happens when the peer to which one has forwarded a claim disappears after a successful transaction.

This risk for a claim is expressed as an expected cost C_n for the nth receiver of the claim, where the probability of the ith receiver disappearing is p_i, the cost of regaining trust after disappearance is CT_i, and the value of the claim in concern is V:

$$C_n = (V \times (1 - p_n) + CT_n \times p_n) \times p_{n+1}$$

Assuming every peer has roughly the same tendency to disappear, this risk can be seen as a constant value, unless $(n + 1)$th receiver is the claimer, in which case the claim disappears, and there will not be a risk.

Therefore, the risk is unrelated with the length of the claim forwarding chain $n + 1$ if $n > 0$, but the utility of the claim tend to decrease for the claimer as the chain grows, because the available storage is likely to be further away. This contradiction is a source of problems described in section 3.3.2.2.

To minimize the risk, n needs to remain 0, meaning that claim forwarding should be avoided whenever possible. This is the same conclusion drawn in [36] as a way for optimization.

3.4.1.2 Safety and risks of SSC

We assess the risk with respect to participating in the creation and circulation of drafts in real terms by storage claims.

The safety property of WAT/i-WAT is that the debt does not disappear without redemption.

There are apparent risks for a participant that the drawers of the acquired tickets disappear, and

by the security rule of the system, the debts are transferred down to the participant in question. This risk for a ticket is expressed as an expected cost C_n for the nth receiver of the ticket, where the probability of the drawer disappearing is p_0, that for the ith receiver is p_i, the cost of regaining trust after disappearance is CT_i, and the value of the ticket in concern is V:

$$C_n = (V \times (1 - p_n) + CT_n \times p_n) \times \prod_{i=0}^{n-1} p_i$$

The risk is dependent on the length of the chain of endorsements, which becomes zero when the ticket is eliminated (redeemed).

To minimize this cost, nodes are expected to take the following evasive actions as described in [154]:

1. They always try to use a ticket the partner has drawn if there is one, thus making sure that a ticket is eliminated whenever there is a chance.
2. They always try to receive a ticket whose chain of endorsements is longer than those of others, thus making sure that n is reasonably large.
3. They prefer selecting a partner from the drawers of acquired tickets, thus increasing the chance of eliminating a ticket.

3.4.2 Simulation

A simulation is made under the following premises:

1. A proximity-based overlay network is formed.
2. The underlying network has alternative routes, and a node is unreachable only when it is offline.
3. There is a way to authenticate the identity of a node

The last premise is introduced to provide a relaxed trust condition so that SCM nodes do not need to duplicate storage claims or forward storing data along the chain of nodes, and the simulation results can be comparable with those of SSC, making a pure comparison of claim-forwarding vs. currency approaches.

3.4.2.1 Conditions

The conditions of the simulation are as follows:

1. The world consists of population P of one of the node types (SCM, Simple SSC or LPLC SSC nodes) described below, where time elapses from 1 to r rounds.
2. Each node needs to rent maximal k storage units from others. A rent of just one unit can start in one round, and the rent can last for n rounds.
3. Each node has k storage units to provide to others because the exchange relation needs to be fair.
4. Each node is offline in a round by probability p (*failure rate*). Since there assumed to be an alternative route, availability of the remote storage is simply $1 - p$.

The behaviors of nodes are defined as follows:

SCM: A node tries to rent the storage unit for which it has a storage claim. If it does not have a claim, it tries to rent a unit from a node in vicinity, and stores the claim specified by the remote node.

The remote node tries to avoid forwarding claims, and creates a new claim whenever there is a space in local storage whose size equals to k (as analyzed in section 3.4.1.1).

A shortcut path from the claimer to the new claim holder is made whenever a claim is forwarded so that storage spaces for claims do not need to be duplicated. As a consequence, though, a claim cannot be forwarded if the claimer is offline.

Simple SSC: A node tries to rent the storage unit for which it has a ticket. If it does not have a ticket, it tries to rent a unit from a node in vicinity, and issues a new ticket (and therefore stores a new storage claim) for the remote node.

At the level of this abstract simulation, the behavior of a simple SSC node equals to that of a SCM node without claim-forwarding.

LPLC SSC: A node tries to rent the nearest remote storage unit that is equivalent to the

promise (always 1 storage unit in this simulation) on a ticket it has, and transfers the ticket in return to the remote node. The remote node can lend the unit in which it holds a storage claim; in which case the claim is forwarded to the drawer of the received ticket as a form of redemption.

Among the candidates of nearest providers of remote storage units, the node tries to select one being the drawer of an acquired ticket (as described in section 3.4.1.2).

A transaction involving a ticket is possible only when the drawer of the ticket is online.

We have made a simulation with $P = 250$, $r = 500$, $k = 5$, $n = 5$, and p varying over 0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.48 and 0.64. We have found that larger P or r does not affect the results much, but affects the simulation time very much. By making n larger, we can simulate the situation of more depleted resources, but the effect should be similar to that of increased p.

3.4.2.2 Evaluation metrics

We have measured the following metrics:

Storage Utility: The number of storage units provided to others (excluding those for claims) is counted for every round, divided by the whole capacity.

Distance: The distance to the remote storage (excluding those for claims) is counted for every round.

The distance is measured in overlay hops, but it is expected to be proportional to the distance in the underlying network as the overlay network is assumed to be proximity-based.

Cost: One unit of cost is counted for every occasion of forwarding a storage claim.

Cost of messaging (multiples of tens of bytes) should be negligible compared to that of claim-forwarding (multiples of megabytes).

3.4.2.3 Results

Fig. 3.6, Fig. 3.7 and Fig. 3.8 shows the results of the simulation.

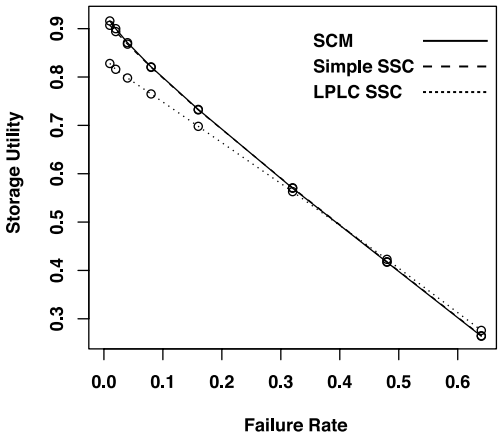


Fig. 3.6. Simulation result — storage utility

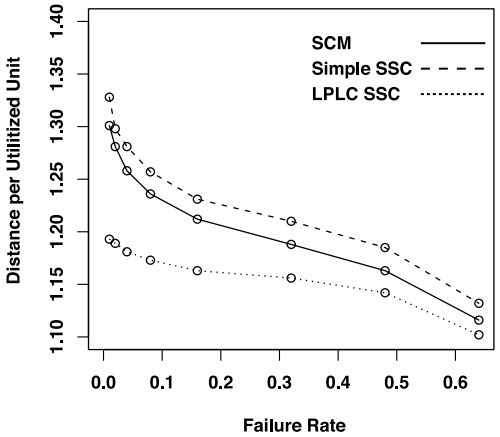


Fig. 3.7. Simulation result — distance per utilized unit

Storage Utility As Fig. 3.6 shows, SCM and Simple SSC nodes can maintain above 90% of storage utility when the failure rate is negligibly low. LPLC SSC nodes provide slightly less storage utility, but are less affected by the rate of failures, and in fact provide slightly higher storage utility than others when the failure rate is above 0.4 (the condition is not unrealistic considering the dynamism of P2P systems).

This must be due to the fact that there are more factors that can disable LPLC SSC transactions than those can disable other types (such as that drawers need to be online), but the difference is reduced as the failure rate grows.

Distance per utilized unit Fig. 3.7 shows that LPLC SSC nodes always provide storage with

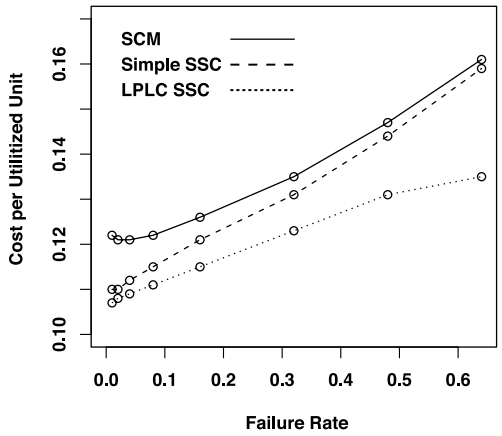


Fig. 3.8. Simulation result — cost per utilized unit

shorter network distance.

The distance decreases steeply for SCM as the failure rate grows. This must be because forming claim-forward chains gets difficult in the first place, resulting in that successful transactions tend to happen in the vicinity of claimers.

Cost per utilized unit Fig. 3.8 shows that LPLC SSC nodes always provide storage with smaller transfer cost, regardless of the failure rates.

Since the improved storage claim protocol described in section 3.3.3.3 is assumed in the simulation, this difference does not come from the effect of exchanging drafts instead of storage claims themselves; claims are transferred every time a ticket is passed to a third party. Instead, the difference must be due to the frequency of redemption, because a claim is not forwarded upon redemption, and redemption is induced by the design of LPLC SSC.

Effects of claim-forwarding The differences between SCM and simple SSC isolate the effects of claim-forwarding. It seems that claim-forwarding does not affect storage utility at all, positively affects network distance (looks paradoxical, but it must slightly increase the chance of exchanging storage units with neighbor nodes than not forwarding claims at all), and negatively affects the transfer cost.

3.5 Related work

3.5.1 Centralized currencies

MojoNation[13] was[2] a system of file sharing, which is perhaps the most well-known in the context of P2P currencies. Every transaction in the system required payment of *Mojo*, which was controlled by a single token server. MojoNation's economy was a form of of MCS[160] (Mutual Credit System).

Karma[181] is a distributed payment system based on *bank-sets*. It is designed to work on DHTs (Distributed Hash Tables) such as Pastry[153]. Its trust model is based on the secure routing discussed in [24] which relies on a set of trusted certificate authorities. Since the whole concern is consistent management of an accounting system in a distributed environment, Karma can be also seen as a form of MCS.

Ripple[54] is a distributed payment system. It finds a chain of credit connections between parties to make payments. If A and B are mutually trusted, and B and C are mutually trusted, and if A wants to make a payment to C, then A pays (owes) to B so that B pays (owes) to C. The account information is made private among trusted parties. Ripple is a form of multiple MCS, in which several MCS's mutually have accounts for one another.

All of the payment systems above are distributed, but their models of economies are centralized. They may be subject to the inherent problem of MCS described in [157]; growing the number of free-riders in MCS has a paradoxical effect of increasing *welfare* of the community. Since there is no pressure among participants to stop the growth of the bad users, it is difficult to sustain the soundness of the system without strong interventions from the operators of the system. This problem implies that MCS is

2 MojoNation does not exist anymore, although development has been continued as a free software project called Mnet[121].

insufficient as a solution to either the first or the second challenge described in section 3.1.2.

3.5.2 Decentralized currencies

BitTorrent[31] is a file distribution system on a tit-for-tat basis. Each peer is responsible for attempting to maximize its own download rate. If peers do not cooperate they *choke* their partners, or temporarily refuse to upload. This economy is integrated into the system, and cannot be separated for other applications.

Geek Credit[97] is a currency system that is close to *i*-WAT. It defines *Geek Credit policy*, which is similar to the state machine of *i*-WAT, but the problem of double-spending is handled differently. Geek Credit detects double-spending at redemption, so that each transaction does not need to be consulted with the drawer. While this simplifies the protocol, exploiting this system may be easy. Moreover, since participants do not need to be reachable to the drawer, induction to redemption may not work too well, possibly imposing higher risks.

PPay[194] is another currency system that is close to *i*-WAT. PPay handles the problem of double-spending in almost the same way as *i*-WAT does; it requires approval (process of *reassignment*) by the issuer of the coins when they are transferred to other parties. The difference is that this authority is duplicated in PPay. It assumes that an external banking facility exists, which exchanges the governments' fiat money with digital coins. Such facility may be given authority to reassign coins.

This makes the currency more available, but it also makes the protocol more complicated than that of *i*-WAT. We believe that availability of the issuers can be increased by applying existing fault-tolerance techniques, independently from the currency design. Since it enhances the utility of tickets, it may increase the level of their freedom to create the exchange media, therefore some issuers may find it beneficial to pay the cost of applying such techniques.

The linkage with a fiat money means that PPay cannot form drafts in real terms in its original design.

3.5.3 Incentive techniques

Fair sharing protocol[129] is a distributed query algorithm which can detect lies about participant's debts probabilistically, which we believe is broadly applicable. However, querying the storage claims gives a more accurate view of participant's debts.

Feldman et al.[52] investigates some incentive techniques to tackle the problem of free-riding. They found that when penalty is imposed on all newcomers, the system performance degrades significantly only when the turnover rate among users is high. This observation can be applied to this work in such a way that all participants should start with relatively small number of trusted acquaintances when joining the overlay network; they start with small opportunities for trades as a kind of penalty.

Stamp trading[122] is a generalized protocol for reputation and payment. This is a theoretical framework, and no practical issues such as double-spending have been addressed.

3.6 Future work

A work[110] is ongoing to implement SSC over an existing reference platform of *i*-WAT. We are also implementing a distributed backup system as an application of SSC. Using those software tools, we will further study the proposed methodology.

In order for the system to work in actuality, we need to develop mechanisms for rendezvous and reputation, as well as a proximity-based mechanism for location and routing.

On more theoretical sides, we need to assess the effects of improved storage claim protocol described in section 3.3.3.3 in comparison with the original protocol proposed in Samsara, and compare designs of SCM, SSC and other techniques in larger variety of metrics.

3.7 Conclusions

This work presented two designs of exchange media based on representation of debts to be used in P2P economies. Forwardable storage claims as proposed by Samsara can form SCM, a commodity money in P2P systems. However, it does not satisfy the principle of LPLC. On the other hand, in SSC, *i*-WAT tickets represent storage claims to form drafts in real terms, so that a claim can be dynamically replaced by its equivalent in the vicinity, allowing the accesses to the acquired remote storage to be fast and robust.

A simulation showed that SSC nodes satisfying LPLC can provide remote storage in more proximity (and thus expected to provide faster accesses to the storage), which is less affected by the rate of failures, than SCM nodes do.

A work is ongoing to implement SSC over an existing reference platform of *i*-WAT, using which we will continue our study of the LPLC storage economics of P2P systems. An application of SSC is being implemented in the form of a distributed data backup system, and will be available for general public shortly.

第 4 章 A Design for Distributed Backup and Migration of Distributed Hash Tables

Summary[41]

To realize huge-scale information services, many Distributed Hash Table (DHT) based systems have been proposed. For example, there are some proposals to manage item-level RFID information with DHT. In such services, service reliability is important. A DHT service is believed to be highly reliable because it is distributed. However, even if the entire service does not stop, the risk of data loss is unavoidable because a DHT consists of many physical nodes. Clearly, in order to avoid compromising the scalability of a DHT-based service, a scalable backup and loss-recovery scheme is required. To solve the problem, we propose a framework of distributed-to-distributed data copy. The proposed design utilizes the distributed manner of DHT algorithms, and it enables coping from a distributed system to a DHT system with minimal bottlenecks.

第 5 章 Conclusions

In this report, the following topics we have investigated this fiscal year have been described in detail:

1. DAS-P2P 2008 International Workshop
2. Storage-standard currency
3. Distributed backup and migration of distributed hash tables

Since all our development projects are in *deploy* phase now, we will continue our efforts for refining our methodologies through actual usage of our technologies in our lives.