

# Software Agents and Electronic Commerce

**CS486-23**

## *Payment and Exchange #2*

Chris Brooks

Department of Computer Science  
University of San Francisco

## 23-0: Introduction

---

- Current-day payment systems (credit cards, PayPal, etc) are sufficient for some needs, but lacking for others.
- Hard to deal with small payments
- Setup required.
- Can't be easily liquidated
- What other solutions are possible?

## 23-1: Past Failures

---

- The e-commerce landscape is littered with failed payment schemes
- Many of these ideas looked appealing, but failed to catch on
- Some had technical problems
- Others were unable to get a market foothold
- Examining these systems can be instructive

## 23-2: Micropayments

---

- One impediment to electronic commerce is the inability to pay small amounts for an item.
- For example, you might be willing to pay a penny to read an article, or 10 cents to watch a video
- Standard payment mechanisms can't handle this
  - The transaction costs are larger than the amount being exchanged!
- A mechanism is needed to allow people to pay small amounts of money for an item

## 23-3: Markets for Micropayments

---

- Potential markets for micropayments include:
  - Songs
  - Articles
  - Images
  - Comic Strips
  - Stock Quotes
  - Book pages
  - Games

## 23-4: Challenges in Implementing Micropayments

---

- There are a number of difficult technical problems to solve in implementing micropayments, including:
  - small transaction costs
  - Fast turnaround (1 second or less)
  - Customer service
    - A customer service call can cost a firm several dollars or more.

## 23-5: CyberCoin

---

- CyberCoin was an early attempt at micropayment,
- Users load an account with a balance
- Payments are then deducted from that account.
- Banks and credit card companies removed 'from the loop'
- System optimized to limit number of messages and crypto operations
  - Small-value transactions can be protected with weaker encryption

## 23-6: CyberCoin

---

- CyberCoin used *assured delivery* to reduce customer service expenses and implement nonrepudiation.
- Encrypted information is sent to user by merchant.
- User's system sends payment to merchant.
- Merchant forwards information and key to CyberCoin
- CyberCoin processes payment and forwards key to customer
- Both sides have protection from fraud and failure

## 23-7: CyberCoin

---

- Why did CyberCoin fail? (according to co-founder Steve Crocker)
- Too much overhead for customers
  - Multiple authentication screens
  - Need to load an account ahead of time
- No critical mass of merchants
- Pricing alternatives emerged
  - Subscriptions
  - Advertising

## 23-8: CyberCoin

---

- Subscriptions are appealing to sellers - guaranteed income
- Easier to predict demand
- Problem: Subscriptions work best for high-value content.
- Ads have also proved to be very successful (so far)
- The small amount a good is 'worth' is charged to the advertiser.
- But ... advertisers prefer volume
- May not be appropriate for small sellers
- Web Ad revenue has declined for (at least) seven quarters

## 23-9: Millicent

---

- Millicent was another micropayment protocol
- Proposed by DEC
- Millicent uses a cash substitute known as scrip
- Only valid with particular merchants
- Purpose: avoid double-spending without expensive transactions or encryption.

## 23-10: Millicent: How it works

---

- User gets an account with a *scrip broker*.
- User wants to buy something from a vendor
- The scrip broker issues scrip for a particular vendor
- User then spends that scrip

## 23-11: Millicent: How it works

---

- More detail:
- A piece of scrip consists of:
  - Vendor ID
  - Expiration Date
  - Value
  - Consumer information (for taxes, discounts, etc)
  - The above part is referred to as the *Info*
  - Session ID
  - Session key
- The session ID contains a unique session number

## 23-12: Millicent: How it works

---

- The vendor stores a secret  $X$ (session number)
  - $X$  depends only on session number (high-order bits, for example)
- Session key is the MD5 hash of session ID concatenated with  $X()$
- A *Coin* is a string with a value, a serial number, and a key.
- A Coin key is the Hash of Session ID + Info + Coin +  $X(\text{serial})$

## 23-13: Millicent: How it works

---

- A customer gets vendor scrip from a broker
  - This scrip contains SessionID, Coin, Coin key and session key.
- Customer sends SessionID, Coin, Info and Coin key to vendor.
- Vendor can verify that the Coin key is valid by hashing the Coin.
  - Customer doesn't know the secret X function, so can't create scrip.
- If Vendor and broker use different X functions, vendor can prevent the broker from colluding with the customer and changing the scrip value.
- Hashing is a very fast operation.

## 23-14: Millicent: Current status

---

- Early promise, but abandoned by Digital in 1999.
- Trial systems being tested in Japan
- Research projects based on Millicent are ongoing.
- Main challenges
  - Infrastructure
  - Vendor adoption
  - Digital's economic health

## 23-15: Netbill

---

- Developed as an online payment system that would allow:
  - Atomic payment and delivery
  - Anonymity of customers
  - Ability for customers to prove group membership (for discounts, etc)
- Developed at Carnegie Mellon. Later acquired by CyberCash and Visa.

## 23-16: Netbill: How it works

---

- Again, digital goods are sent in encrypted form
- Customer receives decryption key only after payment.
- Netbill server acts as a trusted intermediary, so that identity can be hidden from the merchant.
- Public-key encryption is used (in the standard way) to prove authenticity of goods and nonrepudiation.

## 23-17: Netbill: Status

---

- Still has promise, but there are difficult challenges to widespread adoption.
- Scaling group membership databases to large number of vendors in a consistent way is difficult.
- Standardization across issuers and vendors is a challenge.
  - How to get everyone to agree on what a “student” is, for example.
- Transactions can require many messages to be sent.
- Netbill is essentially at the “promising research application” stage; large-scale industry backing will be needed to allow it to be adopted of a wide scale.

## 23-18: More recent systems

---

- A common recent idea is *accumulating balance systems*
- You make a number of smaller purchases which are aggregated.
- You're then charged once for the balance.
- Number of transactions are reduced.

## 23-19: PepperCoin

---

- Founded by Ron Rivest (RSA) and Silvio Micali (zero-knowledge proof)
- Yet another cryptographic delivery system
- Uses selective sampling of payments
- A small fraction  $s$  of the micropayments are selected for processing.
- Each of these users are charged  $\frac{1}{s}$  times their actual payment.
- Other payments are thrown away.

## 23-20: PepperCoin

---

- *In expectation*, everyone pays the same amount
- But the number of transactions is greatly reduced.
- User fairness can be added by having consumers number their transactions.
- When a transaction is selected for payment, at most that number of payments are charged.
- Bank assumes some risk here.
- Recently created a company - not yet generally available.

## 23-21: Conclusions

---

- Most people agree that more flexible payment mechanisms are needed
  - Low transaction costs
  - Anonymity
  - “walk-up”, little setup needed.
- Difficult to get a critical mass
- Backing of large industry players needed

## 23-22: Conclusions

---

- Also hard technological problems
  - How to implement authentication on a wide scale
  - How to make transactions both secure and very fast
- Many good ideas have been proposed, but nothing has caught on yet.
- Unworkable, or just not time yet?