Web-based Universal Micropayment System
A Service-oriented Design Using Enterprise Architecture Approach

Xiang Shao
Bachelor of Information Technology (Honours)

School of Information and Communication Technology
Science, Environment, Engineering and Technology
Griffith University

Submitted in fulfilment of the requirements of the degree of
Doctor of Philosophy

December 2009
Statement of Originality

This work has not previously been submitted for a degree or diploma in any university.

To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Signed:

Date:
Abstract

The e-commerce of low-value online content, like music and videos, has generated considerable revenues worldwide over the past few years, and the market for these micropayments is expected to continue to grow substantially. To allow ‘pay-per-use’ for such content, Micropayment Systems (MPSs) are playing an important role.

Research on MPSs began along with the Internet boom a decade ago, when a number of trial systems appeared, but none achieved great successes. The main reason for the failures was not the limitation of the technologies involved, but user resistance: people had simply been used to getting online content for free. In addition, the transaction costs were rather too high.

The second wave of MPS developments occurred during recent years, when online content available for a small charge became quite popular on the Internet. Many systems dedicated to processing micropayment transactions on the Web emerged, such as PepperCoin, BitPass, ClickandBuy, etc. However, no single system has yet gained wide acceptance among online merchants and consumers, due to the fact that most systems are running locally with limited user base. The current situation has resulted in many problems because various MPSs are concurrently used and competing with each other on the Internet market. Accordingly, recent research efforts in this area have tended to shift towards social and human, rather than technical, issues.
One of the main problems encountered in current micropayment practice on the Internet is that both merchants and consumers are forced to use multiple systems, manage multiple accounts and trust different system operators. In this study, a possible approach to overcome this problem is proposed. It involves a Universal Micropayment System (UMS), which would incorporate the various MPSs, allowing users (both merchants and customers) to use the systems of their choice without the need for multiple accounts or having to change their habits. The main objective of the research is to design and develop the UMS architecture for universal payments, in terms of functionality and payment protocol, allowing the existing MPSs to comply with it without changing their original functionalities.

In order to achieve this objective, an Enterprise Architecture (EA) approach was adopted as the design principle to guide the process of dealing with system requirements, conceptual framework, implementation and measurements. The EA approach was further segmented into three levels – enterprise viewpoint, business viewpoint and solution viewpoint. In the process, three major questions emerged:

1. What system is required?
2. What should the system look like?
3. How to develop the system and measure its performance?

To address these three questions, further detailed approaches – such as a strategic approach for requirement identification, a service-oriented approach for the system design, and a case study approach for system development – were adopted and developed.
The outcome of this research may contribute to the development of system integrations and future design of MPSs. The core element of the design of the proposed UMS is a generic and systematic interconnection approach to enable cross-system interactions among the existing MPSs. It determines an optimised method of integrating the payment services of these systems into a universal level by standardising multiple payment interactions. The service-oriented design of the UMS architecture and protocol ensures high scalability and system compatibility, and may make it acceptable to a wide range of users.

The proposed system also enables protected data exchange at universal transactions and minimises security and related threats for both users and system brokers without overhead computational burden and significant time lag for cross-system payments. The lightweight design of universal payment service allows the participating MPSs to comply with this service without changing their local functionalities and users’ payment habits. Furthermore, the implementation of the proposed UMS and its protocol demonstrates the achievability of a universal micropayment system. Consequently, the conceptual design of the UMS and its services may promote the development of Internet commerce involving micropayments by means of simple, low cost, secure and efficient universal payment portal and protocol.
# Table of Contents

Abstract i
Table of Contents iii
Table of Figures viii
List of Tables ix
Acronyms x
Acknowledgements xi

Chapter 1 Introduction 1
  1.1 Background of the research 1
    1.1.1 Online paid content: subscription vs micropayment 3
    1.1.2 Current and potential markets for low-value content with micropayment 4
    1.1.3 Micropayment system (MPS) in general 5
  1.2 Current MPS issues and desired solutions 6
  1.3 Objectives and research questions 8
  1.4 Research design 10
  1.5 Outline of the thesis 13
  1.6 Chapter summary 15

Chapter 2 Literature review 16
  Part 1: Overview of electronic payment systems
    2.1 Terminology 17
      2.1.1 Electronic money 17
      2.1.2 Electronic payments 18
      2.1.3 Electronic payment systems 18
      2.1.4 Electronic payment service users 19
      2.1.5 Electronic payment methods 20
    2.2 Evolution of electronic payment systems 21
    2.3 Characteristics of electronic payment systems
      2.3.1 Business characteristics 23
      2.3.2 Functional characteristics 25
    2.4 Classification of electronic payment systems 26
      2.4.1 Credit card systems 29
      2.4.2 E-cash payment systems 33
      2.4.3 Micropayment systems 35
    2.5 Security and trust issue of electronic payment systems 39
      2.5.1 Security issues 39
      2.5.2 Trust issues 40
  Part 2: Micropayment systems and related issues 45
    2.6 Evolution of micropayment systems 46
    2.7 Micropayment techniques and cryptography
      2.7.1 Early adopted micropayment methods and techniques 47
      2.7.2 Currently adopted micropayment methods and techniques 50
    2.8 State-of-the-art micropayment systems 51
      2.8.1 BitPass 52
2.8.2 PepperCoin 56
2.8.3 ClickandBuy 60
2.8.4 Wallie 64
2.8.5 PayPal 68
2.9 Summary and analysis 71
  2.9.1 Summary of business roles and functional characteristics 72
  2.9.2 Analysis of functional characteristics 73
2.10 Chapter summary 78

Chapter 3 Research methodology 81
  3.1 Enterprise Architecture (EA) approach 81
    3.1.1 EA framework 82
    3.1.2 How EA approach works in this research 84
  3.2 Current issues and desired micropayment environment 85
    3.2.1 Current situation of MPS 86
    3.2.2 Problem statement 87
    3.2.3 Desired micropayment environment 88
  3.3 An analysis of system design approaches 89
    3.3.1 Market-driven approach 89
    3.3.2 Service-oriented approach 91
    3.3.3 Outcomes of the determined approach 92
  3.4 Approaches to system development 94
    3.4.1 Approaches to system cooperation and integration 94
    3.4.2 Approaches to system implementation and measurement 95
  3.5 Design methodology 96
  3.6 Chapter summary 99

Chapter 4 Requirements for Universal Micropayment System (UMS) 100
  4.1 User requirements for UMS 101
    4.1.1 A uniform payment platform 103
    4.1.2 A user-friendly payment interface 104
    4.1.3 A secure system 105
    4.1.4 A trustworthy system 107
    4.1.5 A system with anonymity and privacy 108
  4.2 MSB requirements for UMS 110
    4.2.1 Original system functionalities retained in place 110
    4.2.2 Minimal modifications applied to existing systems 111
    4.2.3 Comparability, reliability and scalability 111
    4.2.4 Trust 112
  4.3 Functional requirements for UMS 113
    4.3.1 Access authentication 113
    4.3.2 Payment initiation 114
    4.3.3 Payment confirmation 114
    4.3.4 Payment method 115
  4.4 Chapter summary 115
Chapter 5  UMS architecture  
5.1 Universal micropayment interaction approaches  
5.1.1 Interaction between UMS and its users  
5.1.2 Interaction between UMS and multiple MPSs  
5.2 UMS architecture  
5.3 UMS architecture - Universal Access Layer  
5.3.1 Payment initiation DEI  
5.3.2 Payment confirmation DEI  
5.4 UMS architecture - Universal Transaction Layer  
5.4.1 Universal transaction initiation DEI  
5.4.2 Universal transaction confirmation DEI  
5.5 Chapter summary  

Chapter 6  UMS payment protocol  
6.1 Introduction to UMS protocol  
6.2 UMS protocol entities  
6.2.1 Debtor  
6.2.2 Creditor  
6.2.3 Property  
6.2.4 Token  
6.2.5 Reference  
6.3 Protocol functions and messages  
6.3.1 Authenticating buyer  
6.3.2 Identifying vendor  
6.3.3 Allocating accounts  
6.3.4 Verifying money  
6.3.5 Creating unique content identifier  
6.3.6 Transferring payment information  
6.3.7 Storing payment information  
6.3.8 Summary of protocol functions  
6.4 Chapter summary  

Chapter 7  Implementation and measurement  
7.1 Case studies  
7.1.1 Case study selection  
7.1.2 Implementation scenario  
7.1.3 Universal payment interaction scenario  
7.2 Demonstration of universal micropayment via UMS  
7.3 Implementation of universal micropayment protocol  
7.3.1 Authenticating buyer and identifying vendor  
7.3.2 Allocating accounts  
7.3.3 Verifying money  
7.3.4 Initiating universal payment service  
7.3.5 Transferring payment - from MPS-buyer to UMS  
7.3.6 Transferring payment - from UMS to MPS-vendor  
7.3.7 Storing payment information  
7.3.8 Summary  
7.4 Implementation of system mapping  
7.4.1 Mapping Webcoin  
7.4.2 Mapping NetPay  

vi
7.4.3 Summary
7.5 Measurement of UMS
  7.5.1 Unifying universal payment platform
  7.5.2 Providing a user-friendly system
  7.5.3 Providing a secure system
  7.5.4 Providing a trustworthy system
  7.5.5 Protecting privacy and user’s anonymity
  7.5.6 Keeping maximal functions unchanged and minimal system modifications
  7.5.7 Enhancing comparability, scalability and efficiency
  7.5.8 Summary
7.6 Conclusion of the case studies
  7.6.1 Case studies’ demonstrations
  7.6.2 Limitation and possible solution

Chapter 8 Conclusions
  8.1 Overview of the study
  8.2 Conclusion of the study
    8.2.1 Extensive system review and investigation
    8.2.2 Methodology refinement and approach determination
    8.2.3 Requirement detection
    8.2.4 UMS architecture and protocol design
    8.2.5 System implementation and measurement
  8.3 Contributions to knowledge
  8.4 Limitations of the study
  8.5 Directions for future research
  8.6 Chapter summary

References
# Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>Revenues on micropayments for Internet and mobile digital content</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>Research design</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>Roles of users in e-payment systems</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Characterisation model for e-payment systems</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Pre-paid and post-paid systems</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>A classification schema of e-payment systems</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>Micropayment environment</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Business model for micropayment systems</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>BitPass time sequence diagram</td>
</tr>
<tr>
<td>Figure 2.8</td>
<td>PepperCoin time sequence diagram</td>
</tr>
<tr>
<td>Figure 2.9</td>
<td>Wallie time sequence diagram</td>
</tr>
<tr>
<td>Figure 2.10</td>
<td>PayPal time sequence diagram</td>
</tr>
<tr>
<td>Figure 2.11</td>
<td>Payment initiations</td>
</tr>
<tr>
<td>Figure 2.12</td>
<td>Payment confirmations</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>The life cycle of EA approach adopted in this study</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>Enterprise Architecture levels</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Current situation of micropayment</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>Desired micropayment environment from users’ perspectives</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>The new approach for incorporating existing MPSs</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>Health circle for expansion of user bases</td>
</tr>
<tr>
<td>Figure 3.7</td>
<td>Logical design process circle</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>The requirements for UMS</td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>Interactions between UMS and incorporated MPSs</td>
</tr>
<tr>
<td>Figure 5.2</td>
<td>Architecture of UMS with interior levels</td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>Universal Access Level</td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>Universal Transaction Level</td>
</tr>
<tr>
<td>Figure 6.1</td>
<td>Payers and Payees in different MPSs</td>
</tr>
<tr>
<td>Figure 6.2</td>
<td>Interaction flow of universal micropayment</td>
</tr>
<tr>
<td>Figure 6.3</td>
<td>Universal micropayment interactions with associated parameters</td>
</tr>
<tr>
<td>Figure 6.4</td>
<td>Protocol function sequence</td>
</tr>
<tr>
<td>Figure 6.5</td>
<td>Swim Lane diagram for UMS protocol functions</td>
</tr>
<tr>
<td>Figure 7.1</td>
<td>Demonstration of a universal micropayment</td>
</tr>
<tr>
<td>Figure 7.2</td>
<td>Interaction flow between MPSs and UMS</td>
</tr>
<tr>
<td>Figure 7.3</td>
<td>Pre-payment messages exchanged between Webcoin and UMP</td>
</tr>
<tr>
<td>Figure 7.4</td>
<td>Universal micropayment interactions from Webcoin to NetPay via UMP</td>
</tr>
<tr>
<td>Figure 7.5</td>
<td>Function sequence of universal micropayment protocol</td>
</tr>
<tr>
<td>Figure 7.6</td>
<td>Parameters mapped in Webcoin</td>
</tr>
<tr>
<td>Figure 7.7</td>
<td>Parameters mapped in NetPay</td>
</tr>
</tbody>
</table>
List of Tables

Table 2.1 Examples of token and notation-based systems
Table 2.2 Examples of macro, mini and micro payment systems
Table 2.3 Examples of state-of-the-art e-payment systems
Table 2.4 Examples of credit card transaction fees
Table 2.5 Transaction value ranges of different payment categories
Table 2.6 Parameters of BitPass interactions
Table 2.7 Parameters of PepperCoin interactions
Table 2.8 Parameters of Wallie interactions
Table 2.9 Parameters of PayPal interactions
Table 2.10 Business and Functional characteristics
Table 2.11 Fee comparison of different systems
Table 4.1 Functional requirement list for UMS
Table 5.1 Traceability matrix to functional requirements
Table 5.2 Uni-Pay-Req parameters at UAL
Table 5.3 Denotation of the uniqueness of buyer and vendor’s ID within UMS
Table 5.4 Uni-Pay-Resp parameters at UAL
Table 5.5 Uni-Trans-Req parameters at UTL
Table 5.6 Uni-Trans-Resp parameters at UTL
Table 6.1 Debtor entity related parameters at UAL and UTL
Table 6.2 Creditor entity related parameters at UAL and UTL
Table 6.3 Property related parameters at UAL and UTL
Table 6.4 Token entity related parameters at UAL and UTL
Table 6.5 Reference entity related parameters at UAL and UTL
Table 6.6 Parameters stored in UMS database for each universal transaction
Table 6.7 Shape denotations for Swim Lane diagram
Table 6.8 List of protocol functions with section number
Table 6.9 Traceability matrix to functional requirements
Table 7.1 MPS table in UMS database
Table 7.2 Parameters from Webcoin to uni-Webcoin
Table 7.3 Parameters from uni-Webcoin to UMS
Table 7.4 Parameters from UMS to uni-NetPay
Table 7.5 Parameters from uni-NetPay to UMS
Table 7.6 The processed payment data recorded by UMS database
Table 7.7 Parameters for payment initiation of Webcoin
Table 7.8 Parameters for payment confirmation of Webcoin
Table 7.9 Parameters for payment initiation of NetPay
Table 7.10 Parameters for payment confirmation of NetPay
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATM</td>
<td>Automated Teller Machine</td>
</tr>
<tr>
<td>B2C</td>
<td>Business to Consumer</td>
</tr>
<tr>
<td>DES</td>
<td>Data Encryption Standard</td>
</tr>
<tr>
<td>DEI</td>
<td>Data Exchange Initiation</td>
</tr>
<tr>
<td>DEP</td>
<td>Data Exchange Point</td>
</tr>
<tr>
<td>EA</td>
<td>Enterprise Architecture</td>
</tr>
<tr>
<td>EFTPOS</td>
<td>Electronic Funds Transfer at Point of Sale</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>Secure Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>ID</td>
<td>Identifier</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>iKP</td>
<td>Internet Keyed Payment Protocol</td>
</tr>
<tr>
<td>MD5</td>
<td>Message Digest 5</td>
</tr>
<tr>
<td>MPS</td>
<td>Micropayment System</td>
</tr>
<tr>
<td>MSB</td>
<td>Micropayment System Broker</td>
</tr>
<tr>
<td>P2P</td>
<td>Person-to-Person</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure</td>
</tr>
<tr>
<td>RSA</td>
<td>Rivest-Shamir-Adelman, a public key encryption algorithm</td>
</tr>
<tr>
<td>SET</td>
<td>Secure Electronic Transaction</td>
</tr>
<tr>
<td>SHA</td>
<td>Secure Hash Algorithm</td>
</tr>
<tr>
<td>SI</td>
<td>Service Initiation</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Socket Layer</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TTP</td>
<td>Trusted Third Party</td>
</tr>
<tr>
<td>UAL</td>
<td>Universal Access Layer</td>
</tr>
<tr>
<td>UMP</td>
<td>Universal Micropayment Portal</td>
</tr>
<tr>
<td>UMS</td>
<td>Universal Micropayment System</td>
</tr>
<tr>
<td>Uni-Pay-Req</td>
<td>Universal Payment Request</td>
</tr>
<tr>
<td>Uni-Pay-Resp</td>
<td>Universal Payment Response</td>
</tr>
<tr>
<td>Uni-Trans-Req</td>
<td>Universal Transaction Request</td>
</tr>
<tr>
<td>Uni-Trans-Resp</td>
<td>Universal Transaction Response</td>
</tr>
<tr>
<td>Uni-Verif-Req</td>
<td>Universal Verification Request</td>
</tr>
<tr>
<td>Uni-Verif-Resp</td>
<td>Universal Verification Response</td>
</tr>
<tr>
<td>URL</td>
<td>Universal Resource Locator</td>
</tr>
<tr>
<td>UTL</td>
<td>Universal Transaction Layer</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>3W</td>
<td>World Wide Web</td>
</tr>
</tbody>
</table>
Acknowledgements

I would like to thank all those who helped me during my study.

First of all, my deepest gratitude goes to my Supervisor, Dr. Ann Nguyen, whose constant guidance and encouragement have helped make the past four years a smooth and rewarding journey.

I am also indebted to the Griffith School of ICT for giving me the opportunity and APA Scholarship to commence the study in the first instance, and additional funding for attendance at various conferences and career development workshops.

Thanks are also due to my Associate Supervisors, Dr. Vallipuram Muthukumarasamy and Head of School, A/Prof. Michael Blumenstein, for their advice, research assistance and support throughout my whole journey of PhD study.

I cannot end without thanking all my family and friends, wherever they are. To my friends and fellow PhD students, I treasured all the precious moments we shared. To my family, I thank you for your love, support and sacrifice, upon which I relied throughout the study in Australia.
Chapter 1 Introduction

This chapter presents the general introduction to this research. Section 1.1 introduces the research background with regard to low-value online paid content, the current market of micropayments and systems to handle those payments. Section 1.2 describes the current issues of micropayment systems and desired solutions. Section 1.3 defines the research objectives and formulates research questions. Section 1.4 provides the research design where the flow of research is diagrammed. Sections 1.5 and 1.6 outlines the Thesis and summarises this chapter respectively.

1.1 Background of the research

This research has been conducted during the Internet commerce boom period. An Internet Activity Index created by the Online Publishers Association indicates that the public accessed and used the Internet primarily for four major activities: content, communications, commerce and search (OPA 2005a). The report also shows that obtaining content is the main reason for the public to access Internet, accounting for 72% whilst other three activities are 13%, 8%, and 7% respectively. This means there are still a majority of Internet users who access websites for information, news and entertainment related content that the Internet provides.

It’s difficult to precisely count the amount of the online content published on the Internet because the quantity of content could be measured by both static web pages
and dynamic ones linked by web-accessible databases. A research performed by the University of California at Berkley (Lyman & Varian 2003) estimated that the volume of digital content published had reached 5.6 million terabytes ($10^{12}$ bytes) in 2003 and the storage of new information had been growing at a rate of over 25% a year. Not only is the quantity of online content growing, but also the diversity (Leong 2003). Internet content is not limited to text information. According to Boumans’s report (2005) and the Jupiter research (Card 2005), multimedia content (e.g. music and movies), online games and entertainment information is becoming the major content types that attract the attention of broadband users.

Since the beginning of 3W, Internet users have always considered that content should be available for free (Boumans 2005). That explains why many online suppliers now still allow their content to be accessed free of charge. Such user’s perception has resulted in advertising subsidy being the main form of revenue for those websites offering free content. However, this situation has changed over the past few years, and more content with low fees are now increasingly uploaded, and more and more chargeable content will be published in the future. The content suppliers realize that customers are willing to accept the valuable content with an access fee, therefore, a rising number of suppliers have been starting to charge users for access to their content. It is expected that online content will become a major source of revenue for content suppliers in the near future. As a result, the paid content will have an enhanced quality.
1.1.1 Online paid content: subscription vs micropayment

The price of online content also varies from a few cents for an image file up to hundreds of dollars for commercial software. Customer spending for low-value online content shows strong growth in recent years, especially in the US market, reaching a revenue of US$2 billion in 2005; a continuous increase of 15% compared with the revenue in 2004 (OPA 2006), which was an increase of 13.7% compared with 2003 (OPA 2005b). Such growth was driven largely by the entertainment (music and video) online sales.

At present, the payment of online content is being collected according to various schemes such as subscription and pay-per-use. Pay-per-use (e.g. pay-per-click, pay-per-view, pay-by-time) pricing model usually comes with digital content at a small cost – a few cents to a couple of dollars, and the associated individual payments are comparably micro, hence they are regarded as micropayments.

The OPA’s reports (2005b) reveal that subscriptions continue to be the dominant online content pricing model, accounting for 88.5% of paid content sold in 2003 and 84.6% in 2004. Within the pay-per-use model, micropayments (under $5) have steadily increased. For instance, in the US market, the share of micropayments increased from 7.4% in 2003 to 17.9% in 2004.

Online content that is sold by micropayment methods will be considered further in this study.
1.1.2 Current and potential markets for low-value content with micropayment

Low-value content with micro-transactions has been generating increased revenues steadily over the past few years. According to the research results published by TowerGroup, the total market for Internet and mobile micropayments, led by demand for digital content, will increase by 23% annually over the next five years to reach $11.5 billion in total (generated from both the Internet and mobile phones) by 2009 (Loftesness 2004). Figure 1.1 charts the micropayments market at just less than $2 billion in 2003.

Figure 1.1 Revenues on micropayments for Internet and mobile digital content
(Source: TowerGroup 2004)

Examples of low-value digital content sold via micropayments on the Internet could be classified into, but not limited to, the following types:

- downloadable music (e.g. mp3 files),
- video on demand or access to live streaming events (e.g. watching streaming movies or football matches),
- adult entertainment (e.g. pictures, videos and chatting),
• journalistic content (e.g. local/national/world news, sports, technology),
• entertainment information (e.g. movie listings),
• online games (e.g. SpinOff),
• financial information (e.g. stock quotes),
• daily life information (e.g. cyber dating, fortune telling),
• voice over IP services (e.g. Internet telephony),
• e-tickets (e.g. lottery or public transportation tickets), and
• donations.

Among the above, the music downloads have gained a significant profit since 2004, and most revenues were collected from micropayments for individual downloads rather than subscriptions. According to news from Apple’s Media (2005), the iTunes online music store has sold more than 300 million songs at March 2005 following its price reduction to 99 cents per song two years earlier, and this quickly reached 1 billion songs in 2008 to become American’s No. 1 music store; the first time that a seller of digital downloads has ever beaten the big CD retailers (Quinn & Chmielewski 2008). Forrester Research (2003) estimated that in 2007 the revenue from music downloads will grow to €1.3 billion in Europe and US$2 billion in US market. The success of music download e-commerce has raised a new wave of micropayments and induced the fast development of Micropayment System (MPS).

1.1.3 Micropayment System (MPS) in general

The market for online paid content with low-value, like music and videos, will continue to grow substantially in future years. To allow a “pay-per-use” pricing model
rather than subscription for charging such content, MPSs are expected to play an important role. As an efficient alternative of credit card systems, MPSs that support small value (e.g. less than 2 dollars) and sometime even tiny value (e.g. a few cents) transferred online, make such payments economically feasible.

The history of MPS can be traced back to the late 1990s when micropayment operators like PayWord, MilliCent, Beenz, CyberCoin, DigiCash etc., failed to generate enough business (Hines 2004). Most of them had not been put into real operation and some never even started up. The main reasons for failure were not the limitation of technology, but user resistance as people were used to getting online content for free; also the transaction costs were high.

The development of MPS became stagnant since 2000 until recent years. Many new companies, dedicated to processing micropayment transactions on the Web with low costs have introduced commercial services. Most of them are running locally and nationally, such as PepperCoin, BitPass (both of them based in US market), ClickandBuy (Germany), PayStone (Canada), Pico-Pay (Australia), PayAsYouClick (U.K.) and so on.

1.2 Current MPS issues and the desired solutions

The current practice on the Internet shows that there are already many competing MPSs currently operating in the market, all possessing a small portion of local market share. Those MPSs have maintained their own users (both registered merchants and customers), and functioned with their local significance. There is no one yet that has
gained global acceptance as online credit card usage does. Though the credit card transaction is still the dominant online payment model for Internet commerce, and it is also technically possible to justify such a transaction for a micropayment (less than one dollar), it’s either economically unsuitable or beneficially unacceptable by online merchants.

This situation has resulted in customers and merchants being forced to use multiple MPSs to pay and receive small sums of money. To process such a payment, both accounts need to be created and stored within the same system. This means if customers want to pay merchants who use a MPS other than theirs, or merchants want to attract more customers who are signed up with different MPSs, both of them need to use multiple MPSs to serve all their needs. Consequently, customers and merchants have to face problems such as installing multiple software packages or hardware devices, learning the usage of several systems, managing multiple accounts or e-wallets, remembering multiple passwords, trusting different systems and so on.

Learning from the success of credit cards that have been widely acceptable in both the physical and cyber world, it is clear that users desire a uniform or universal platform for processing micropayments, where both customers and merchants could complete a micro-transaction safely and efficiently without the above mentioned issues. As a consequence, the numbers of customers and merchants could expand maximally, and the marketplace for micropayment would develop maturely.
1.3 Objectives and research questions

The outcome of this research is to develop a realistic approach for system integration as to achieve the desired solution and then propose the architecture of such a system, so-called Universal Micropayment System (UMS), which plays the role of the desired micropayment portal, allowing the incorporation of existing MPSs to achieve a cross-system payment. Such type of incorporation refers to a universal micropayment protocol, under which the interactions among multiple MPSs could be performed. The architecture models the UMS in terms of functionality and structure.

Therefore, the objectives of this research are:

- to extensively investigate the electronic payment systems with the main focus on current MPSs, and to compare their characteristics and functionalities;
- to select and develop suitable approaches for system development in terms of system requirements definition, system integration and implementation as well as measurement;
- to develop a universal payment protocol for system mapping, which enables an integrated payment service such that the majority of existing MPSs may comply with this payment integration without changing their local functionalities and user’s payment habits;
- to present an architecture of the UMS, allowing customers and merchants to use the MPS of their choice for paying and receiving small sums of money to all merchants and from all customers, respectively;
• to describe the UMS architecture not only suitable for most micropayment environments but also feasible, scalable and secure for most business micropayment models;

• to enable the implementation of the UMS portal and platform that is commercially efficient, in terms of both development, and in on-going cost and time; and

• to make the design of the UMS as a guideline for the design of future MPSs, and a de facto standard for systems integration.

Before such a system is proposed, several research questions need to be defined and addressed. They were categorised into the following main questions and sub-questions:

(1) What are the possible approaches to address the system incorporation problems and how?

• How to define the universal micropayment service such that various existing MPSs can comply with it without changing their original functionalities?

• What are the suitable methods for systems interconnection and service integration?

• How will the payments interact under this universal payment protocol of the UMS?

(2) How is the UMS modelled and realized?

• What are the user and function requirements for the UMS that suit the multitude of existing MPSs?
- How to design the UMS architecture and its protocol, and how does the UMS payment service work?
- How to implement the UMS in term of its system integration and mapping with other MPSs?
- How to measure the implementation of the UMS and evaluate its performance?

Other issues may arise besides the above mentioned concerns, such as transaction costs and business agreements. The transaction costs may increase for merchants as the service provided by the UMS needs to be compensated. But the final cost upon an average transaction may decrease instead, because the use of the UMS service will largely increase the number of transactions, resulting in a lower cost per transaction. The business agreements between the various MPSs will also need to be settled prior to the process of system integration. However, those issues are not within the scope of this research.

1.4 Research design

Figure 1.2 presents a conceptual diagram that depicts the holistic design of the research flow and how different elements of the research are inter-related. The research involves the investigation of e-payment systems and MPSs previously proposed and currently in use, identification of system requirements, selection of system integration approaches and design of system architecture and protocol, as well as system implementation and analysis. Each of these is represented in the boxes from
top to bottom in the diagram below. The arrow connecting the top and bottom boxes suggests the relationship of elements and flow of research.
Chapter 3 identifies the current problems and highlights the research questions, identifies the suitable approaches to address those questions, and finally determines design methodology for system implementation and measurement.

Chapter 4 defines requirements for UMS (in term of customer, merchant payment system brokers and functions).

Chapter 5 UMS architecture:
- UMS architecture design; and
- Analyse and define the most suitable solution for system’s interconnection and integration.

Chapter 6 UMS protocol:
- Identifying the protocol entities and defining protocol functions and messages.

Chapter 7 UMS implementation:
- Demonstration of UMS payment scenario;
- Theoretically implementation of the proposed UMS using case studies; and
- Analysing the usability of the UMS models and achievability of the UMS incorporation approach.

Conclusion:
- Have the research questions been addressed?
- Limitations, future work and contributions.

Figure 1.2 Research design
1.5 Outline of the thesis

This thesis is structured into eight chapters. A broad overview of literature starting from the review of both e-commerce and networking theory, and general electronic payment models and systems, is presented in the first part of Chapter 2, along with the e-payment related terminology and e-payment system classification characteristics. The second part of Chapter 2 examines the literature on the various MPSs. During the review process, the proposed and operational MPSs have been investigated extensively in term of their functionalities and payment protocols, followed by an analytical summary presented at the end of the Chapter.

Chapter 3 gives the research approach and methodology applied in this study. The main approach and research framework are introduced, and then followed by a discussion on how this approach works in this research. Then the current problems raised from literature review are highlighted, and the desired solutions are discussed. Afterwards, an analysis of system design approaches is given; suitable approaches to system development (integration, implementation and measurement) are analysed and determined. Subsequently, the design methodology is refined along with five stages of the design cycle being identified.

Chapter 4 presents the requirements for the proposed UMS derived from the conclusions of the literature review outlined in Chapter 2. The requirements formulated from the perspectives of the stakeholders, such as customers, merchants, and payment system operators, led to the system functional requirements for system design.
Chapter 5 discusses the design of the UMS architecture, aimed at addressing the functional requirements identified in Chapter 4. Two possible approaches to the main interactions associated with those functional requirements are identified, compared and analysed, and the most suitable option is determined. To effectively implement those approaches in the design of UMS functions, the major UMS components are highlighted, which then leads to the design of the overall UMS architecture. To better understand the UMS architecture, two internal layers of the UMS are identified and explained with the definition of associated parameters.

Chapter 6 introduces the UMS protocols and examines the process of the universal micropayment transactions interacted by these protocols. The differences between the two UMS layers – defined in previous Chapter, is analysed at the beginning. Protocol entities are then identified. Given that the main focus of protocol design is to determine and regulate the entities behaviours – protocol functions, the suitable solutions to realize those behaviours are discussed. The protocol messages carried by the behaviours are presented with the exchanged data, associated parameters and rules of data mapping.

Chapter 7 describes the implementation of the UMS protocol and measurement of its universal payment service. Two cases are selected for the theoretical demonstration of the proposed payment protocol and service, and then followed by the presentation of the protocol functions and cross-system mapping. The end of the chapter includes a detailed analysis of the measurement of the UMS service, and an evaluation of the fulfillment of the UMS requirements defined in Chapter 4.
Finally, Chapter 8 concludes the research. The contributions of the study are highlighted and the limitations of the research are discussed. The directions for future research in the field of development of micropayment systems, with reference to micropayment standards, are also suggested.

1.6 Chapter summary

This chapter laid the foundation of this thesis. It introduced the basic background to this research, presented the specific research objectives, briefly introduced the research problems, gave the possible solutions, determined the research methodology and outlined the structure of the thesis.
Chapter 2  Literature review

Part 1: Overview of electronic payment systems

To design a new electronic payment (e-payment) system, it is necessary to first understand the existing payment systems. The first development of e-payment systems started in the early 1960s, when the Bank of America introduced a new payment instrument, called a credit card, which at present is the most popular form of e-payment in both the cyber and physical world. Another new innovation emerged in 1995, when Mondex introduced Mondex cash – the first form of electronic cash, which is held in a smart-card based electronic purser (Asokan & Phillipe 1997). In the same year, the Mark Twain Bank deployed DigiCash, an e-payment system involving an anonymous form of electronic money that was first introduced by Chaum et al. in 1990. With the fast development of personal computers, electronic wallets (e-wallets) became more acceptable and practical since the late 90s, and once dominated the e-payment market. In recent years, micropayment and mobile payments have thrived and started to gain an increasing market share.

This part of the literature review is focused on e-payment systems in general that are used for online transactions in the Business-to-Consumer (B2C) environment. Section 2.1 and 2.2 defines the related terminology and introduces the evolution of e-payment systems, respectively. Section 2.3 describes the main characteristics of e-payment systems. Section 2.4 presents the classification of e-payment systems, where three types of systems (credit card systems, e-cash systems and micropayment systems) are
briefly discussed. Finally, the security and trust issues of e-payment systems are highlighted in Section 2.5.

2.1 Terminology

Since the terminology related to e-payment systems varies in literature, a consistent terminology will be defined in this section, and will be used consistently in the following chapters of this thesis.

2.1.1 Electronic money

Three types of money exist: cash, bank money and electronic money (e-money). E-money is an electronic medium of exchange in an electronic environment. The European Central Bank (1998) defines e-money as “an electronic store of monetary value on a technical device that may be widely used for making payments to undertakings other than the issuer without necessarily involving bank accounts in the transactions”. E-money is usually issued by banks, but also by other independent and recognized e-money organisations.

E-money differs in many ways from traditional cash. One aspect is the security. Cash is secured by adding physical features such as special watermarks for visual verification at the transaction. E-money is secured with cryptographic codes and digital signatures. The second aspect is the way they are exchanged. Cash is physically exchanged in the forms of notes and coins, while e-money is exchanged in the form of bits and bytes. In addition, cash could make multiple payments, while e-
money can only make a single payment at a time (Baddeley 2004, O’Mahony et al. 2001).

According to its characteristics, e-money is usually stored and exchanged through electronic devices (e.g. computer) or instruments (e.g. smart card). E-money products can be easily grouped into network/software-based schemes and card-based schemes. According to the survey conducted by the Bank for International Settlements (2004), the card-based products have gained relatively more success than the former ones.

2.1.2 Electronic payments

Compared with cash payments performed by using cash, electronic payments (e-payments) are payments made with value transfers using e-money. E-payments are initiated, processed and acknowledged electronically. In literature, e-payments usually involve value transfers between financial institutions, and require that one or both parties (payer and payee), and one or both financial institutions be specified (O’Mahony et al. 2001). Electronic payments are traceable, but the increasing concerns of consumer’s privacy raised the research interests in untraceable e-cash (e.g. Brands 1994, Chaum et al. 1990, etc.), which enables payers to remain anonymous to payees during online transactions.

2.1.3 Electronic payment systems

An electronic payment system is a system that makes exclusive use of electronic communication channels (e.g. computer networks) to perform payment transactions
(O’Mahony et al. 2001). Such systems are widely used in daily life, for instance, paying for physical goods and services, household bills, online content, etc.

This study is mainly focused on e-payment systems that are used to process payment transactions on the Internet. Other types of e-payment systems such as ATM, EFTPOS, e-cheque systems, mobile payment systems, etc. are not within the scope of this research.

2.1.4 Electronic payment service users

In the context of this study, two types of service users are distinguished: customers and merchants. They have different roles in the e-payment environment. Their relationship is shown in Figure 2.1.

![Figure 2.1 Roles of users in e-payment systems](image)

A customer is defined as an individual person equipped with an electronic device (e.g. computer, laptop, mobile phone or PDA) who purchases products from online
merchants. Therefore, a customer acts in two roles: shopper and payer. The shopper requests, receives and consumes products. The payer is a part of the e-payment system, using account credits to initiate a payment via an e-payment system. In literature, other terms to describe a customer are found, such as end user, consumer, buyer and purchaser (OPA 2004, Stiller et al. 1998).

A merchant is defined as an individual person or an organisation that provides products on the Internet to customers, and is paid for those products. Therefore, a merchant comprises two roles: supplier and payee. The supplier delivers products to customers after a payment is received. The payee is another part of the e-payment system, receiving and verifying the e-payment from customers. The relationship between merchant and customer is defined as the B2C relationship. Other alternative terms for a merchant in literature are like seller, retailer, content provider or vendor (OPA 2004, Stiller et al.1998).

2.1.5 Electronic payment methods

According to O’Mahony et al. (2001), there are three principal e-payment methods currently in use on the Internet. The first method is applied by credit card systems, where the card number is encrypted and transmitted over the open network. The second method could be found in digital cheque systems whereby electronic cheques are sent over the Internet and cleared off-network. The third method is based on digital money, which is transmitted and cleared on-line at the time of the transaction, or verified off-line. The fourth method, which appears in the forms of smart cards, stored-value cards and downloaded electronic “wallets”, could be associated with
small-value purchases or micropayments, for units of online content such as articles, pictures, songs, and video streams.

2.2 Evolution of electronic payment systems

The first generation of e-payment systems started in 1992 (Bohle 2002). Credit cards were mostly used for making payments in both the physical and cyber world. Subsequently, great efforts had started to develop, implement and deploy new e-payment systems for use on the Internet. Those efforts were focused on the improved security for high-value transaction (e.g. CyberCash and iKP) and on the anonymity of e-payments (e.g. ECash and MagicMoney). However, their success was very limited; many initiatives failed to generate the market due to the lack of standards for payment systems. Examples of failed systems are FirstVirtual, DigiCash, IBM’s Micropay, and Compaq’s MilliCent, etc.

The failure of these attempts was also related to the lack of user’s acceptance (Kniberg 2002). Around the mid 1990s, many off-line micropayment approaches and systems proposed to support small payment transactions against credit card systems were put on trial on the market. Most of them did not survive because of being unable to reach a large number of transactions or gain wide adoption among users, or because of difficulties with subscribing and using the system, or due to high transaction costs and low speed (Kniberg 2002, Abrazhevich 2004). In fact, credit card system still remains dominant for electronic commerce transactions despite the criticism from the literature for its lack of efficiency and security as well as high transaction fees (Shon & Swatman 1998, Simpson 2004).
The second generation of Internet payment systems emerged after 2002, when the fee-charged content started to get well accepted on the Internet, especially those offering multimedia content. These systems have characteristics such as supporting pre-paid or virtual accounts like NetBill and BitPass systems, supporting low-value transaction like PepperCoin and PayPal systems, and supporting multiple pricing models like the ClickandBuy system and so forth.

Mobile payment systems started to appear in the same period. These systems use other communication networks (e.g. GSM) rather than the Internet. Mobile payment systems allow payments in both real and virtual worlds, and has gained increasing success in some countries. Mobile payment systems could become serious competitors to the current e-payment systems on the Internet, especially in East Asia and Western Europe. Those systems will not be further discussed in this study due to the limitation of the research scope.

2.3 Characteristics of electronic payment systems

According to the literature reviewed, e-payment systems are usually classified based on various characteristics such as security, ease of use, speed, anonymity, online or offline, pre-pay or post-pay, account-based or token-based, etc. (O’Mahony et al. 2001; Abrazhevich 2001a; Yu et al. 2002). In this section, a characterisation model for e-payment systems is presented from the business viewpoint, in terms of e-payment system entities and functions.
2.3.1 Business characteristics

In general e-payment systems, a number of system entities could be identified, such as payer, payee, issuer, acquirer, etc. They play different business roles during a payment transaction. Figure 2.2 shows a business characterisation model, which illustrates the process flow of payment systems, which currently operated by banks, financial institutions and payment system operators.

In this model, various entities are represented as rectangles, having different relationships and interactions with each other.

- The consumer and supplier (as pre-defined in Section 2.1.4), are performing request and delivery roles, respectively. The consumer is defined as an individual person with an electronic device requests and receives products from a supplier, who is defined as an individual person or an organisation that receives requests from consumers and delivers products.
- The payer is performing the buying role of a customer and the payee is performing the selling role of a supplier. In an e-payment, a payer pays an amount of money to a payee.
- The issuer and acquirer play the role of registering payers and payees, to allow them to use an e-payment system.
- The brand and license organisation identifies a particular payment instrument and defines units of e-money. It regulates issuers and acquirers, playing the role of a trusted third party.
• The bank plays the role of handling paid money transfers on behalf of third parties, such as consumers, issuers and acquirers.

![Characterisation model for e-payment systems](image)

**Figure 2.2 Characterisation model for e-payment systems**  
(Developed from Abrazhevich 2004)

The round circle in the above model represents the exchange point of e-money in a payment transaction. The arrows represent the exchanged information and money flows. A payment starts when the consumer initiates the money transfer and is completed when the supplier receives the e-payment acknowledgement.
2.3.2 Functional characteristics

Functional characteristics describe the interactions between e-payment systems and system users, and also describe the information exchanged during these interactions. There are many interactions between systems and users, but in this study, only those interactions related to payment initiation and acknowledgement are discussed. Other interactions, such as registration, login, reviewing of payment histories, and checking account balance, are local behaviours of consumers and suppliers, thus they are not considered further.

The result of the initiation refers to the system acceptance of the payment, whilst the result of the payment acknowledgement refers to the payment confirmation after the payment transaction is completed. An initiation is usually followed by an acknowledgement in a transaction, but the payment authorisation process may differ. Depending on the sequence of the authorisation, the e-payment systems could be generally divided into two basic types: pre-paid and post-paid, as shown in Figure 2.3.

- A pre-paid system requires customers to deposit money to the system. The amount of money transferred to the system will be stored in the form of e-money. The credit is usually checked before the payments are authorised. The customer may not receive authorisation message during the process of a transaction.
- A post-paid system authorises customers to initiate payments without a need of depositing in advance, but customers need to present the system a reliable money source, from which the amounts of money can be transferred to the
system (after a certain time period). If the money source is valid and credit limit is not exceeded, the payments are authorised.

![Diagram showing pre-paid and post-paid systems]

**Figure 2.3 Pre-paid and post-paid systems**

The literature shows many e-money related systems are pre-paid type and most of credit card based systems are post-paid type. They are further described in the next section. The characteristics of the investigated systems are collected based on the information available on the websites of these systems, relevant articles and implementation documents, as well as payment experiments with these systems.

### 2.4 Classification of electronic payment systems

In literature, different classifications of e-payment systems can be found. They are based on the type of e-money, value of payments, and type of payment instruments.

A classification distinguishes between e-cash and account-based systems (Figure 2.4, Abrazhevich 2001a), or between the so-called token and notation-based systems (Camp *et al.* 1995, Pilioura 1999, Weber 1998). The e-cash systems are split into
smart card systems (e.g. Mondex) and online cash systems (e.g. NetCash and ECash).
The account-based systems are split into generic systems (e.g. NetBill and PayPal),
specialised systems (e.g. BitPass and Wallie), and credit card systems (e.g.
CyberCash, SET and Cirrus). Table 2.1 lists the examples of token and notation-based
systems grouped by the payment validation methods, which are namely online, semi-
online and offline.

Figure 2.4 *A classification schema of e-payment systems*
(Source: Abrazhevich 2001a)

<table>
<thead>
<tr>
<th>Model / Validation</th>
<th>Token E-Cash</th>
<th>Notation Account-based</th>
<th>Notation Credit Card</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Online</strong></td>
<td>ECash</td>
<td>BankNet</td>
<td>CyberCash</td>
</tr>
<tr>
<td></td>
<td>MagicMoney</td>
<td>CyberCoin</td>
<td>FirstVirtual</td>
</tr>
<tr>
<td></td>
<td>NetCash</td>
<td>FSTC</td>
<td>iKP</td>
</tr>
<tr>
<td></td>
<td>PayMe</td>
<td>NetBill</td>
<td>SET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NetCheque</td>
<td>VeriFone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PayPal</td>
<td>Cirrus</td>
</tr>
<tr>
<td><strong>Semi-online</strong></td>
<td>MilliCent</td>
<td>MiniPay</td>
<td></td>
</tr>
<tr>
<td><strong>Offline</strong></td>
<td>CAFÉ</td>
<td>PepperCoin</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mondex</td>
<td>BitPass</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NetCard</td>
<td>Wallie</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NetCents</td>
<td>CyberCents</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PayWord</td>
<td>InterCoin</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SVP</td>
<td>NetPay</td>
<td></td>
</tr>
</tbody>
</table>
Other classifications (Weber 1998) presented in Table 2.2 are based on the value of payments, which are namely macro-payment, mini-payment and micro-payment. The typical transaction value range is shown on Table 2.5.

### Table 2.2 Examples of macro, mini and micro payment systems

<table>
<thead>
<tr>
<th>Model / Validation</th>
<th>Macro-payment</th>
<th>Mini-payment</th>
<th>Micro-payment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Online</strong></td>
<td>BankNet</td>
<td>CyberCoin</td>
<td>iTune</td>
</tr>
<tr>
<td></td>
<td>CyberCash</td>
<td>ECash</td>
<td>PayPal</td>
</tr>
<tr>
<td></td>
<td>FirstVirtual</td>
<td>NetCents</td>
<td>PayStone</td>
</tr>
<tr>
<td></td>
<td>SET</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VeriFone</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Semi-online</strong></td>
<td>CAFÉ</td>
<td>MilliCent</td>
<td>Micro-iKP</td>
</tr>
<tr>
<td></td>
<td>NetCents</td>
<td>MiniPay</td>
<td></td>
</tr>
<tr>
<td><strong>Offline</strong></td>
<td>Mondex</td>
<td>CyberCent</td>
<td>PepperCoin</td>
</tr>
<tr>
<td></td>
<td>NetCard</td>
<td>PayMe</td>
<td>BitPass</td>
</tr>
<tr>
<td></td>
<td>SVP</td>
<td>NetCash</td>
<td>Wallie</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NetPay</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PayWord</td>
</tr>
</tbody>
</table>

The state-of-the-art classification of e-payment systems is based on the type of payment instruments, which could be found in O’Mahony *et al.* (2001), and Yu *et al.* (2002). This classification makes a distinction between credit card, e-check, e-cash, micropayment and mobile payment systems (Bohle 2002). The examples of each group are listed in Table 2.3.
In this research, the following three types of payment systems are considered and further discussed:

- Credit card systems, which are the most popular payment systems on the Internet;
- E-cash payment systems; and
- Micropayment systems.

E-cheque systems are only based in North America and mobile payment systems have arisen recently and have a great potential for future Internet payments. They are outside the scope of this study.

### 2.4.1 Credit card systems

Online credit card payments make use of the existing credit card payment infrastructure. Credit cards allow buying on credit, according to the “buy now, pay later” principle (Pilioura 1999). A research conducted in 2000 shows that 95% of

---

**Table 2.3 Examples of state-of-the-art e-payment systems**

<table>
<thead>
<tr>
<th>Credit card</th>
<th>E-cash</th>
<th>E-check</th>
<th>Micropayment</th>
<th>Mobile payment</th>
</tr>
</thead>
<tbody>
<tr>
<td>CyberCash</td>
<td>ECash MagicMoney NetCash CyberCoin PayMe CAFE Mondex NetCard NetCents</td>
<td>ECheck FSTC NetCheque NetBill</td>
<td>PepperCoin BitPass Wallie Click&amp;Buy Way2Pay NetPay PayWord Micro-iKP PayStone</td>
<td>PayBox (Germany) Moxmo (Holland) Paiement CB (France)</td>
</tr>
</tbody>
</table>
online payments were performed using credit card systems (Kerr 2000). At the time of writing, credit card payment is still the most popular e-payment instrument. The major reasons for this are its extensive user-acceptance, mature technique and support for multicurrency payments (O’Mahony et al. 2001).

Credit card systems, however, have several drawbacks. From the online merchant’s point of view, the main drawback is the high transaction cost. For each credit card transaction on average, merchants will be charged around 20 to 30 cents (in U.S. currency) plus a small percentage of total sales (McBride 2003, Nasaw 2004). Table 2.4 gives a few examples of transaction fees charged by systems with credit card processing. Information regarding the various fees is provided on the websites of these system operators. Therefore, credit cards are mostly used to pay larger amounts of money rather than the price of low-value products.

<table>
<thead>
<tr>
<th>Credit card payment processors</th>
<th>Fee 1 (US Dollar)</th>
<th>Fee 2 (US Dollar)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CityBank</td>
<td>2.5% + $0.45 (Visa)</td>
<td>2.1% + $0.45 (MasterCard)</td>
</tr>
<tr>
<td>PayPal</td>
<td>2.2~2.9% + $0.30 (in U.S.)</td>
<td>3.9% + $0.40 (in Europe)</td>
</tr>
<tr>
<td>PayStone</td>
<td>2.8% + $0.10 (below $5)</td>
<td>2.8% + $0.30 (over $5)</td>
</tr>
<tr>
<td>NetBilling</td>
<td>1.5% + $0.15</td>
<td>$0.45</td>
</tr>
</tbody>
</table>

From the customer’s point of view, credit card payments violate their privacy. Credit card numbers are associated with the names of their bearers, and as a consequence both the salesperson and the bank will know the person’s identity and can associate bought goods with the name. Another disadvantage is that they are not available for
everybody (e.g. young people without a stable income). Furthermore, card holders fear that their card number might be fraudulently intercepted by unauthorised third parties (Bohle et al. 2000).

From the bank’s point of view, online credit card payments have a high fraud risk. The estimated fraudulent online credit card payments caused a loss of US$2.6 billion in 2004, according to Moore (2003), and it was estimated to grow up to US$3.6 billion in 2007. Nowadays special care is taken to encrypt web traffic of sensitive sites. The protocols named Secure Socket Layer (SSL) and Transport Layer Security (TLS) (Dierks & Allen 1999, Dierks & Rescorla 2004) were developed to secure the communication between web servers and web browsers. Since the development of SSL and TLS required very little work on the part of the merchant and none at all on the part of the consumer, they have been adopted by the vast majority of online retailers.

At the meantime, both merchants and banks are searching for alternative payment methods rather than credit cards (Wills & Favier 2002). The most relevant attempts to make secure online payments are:

- Secure Electronic Transactions (SET)
  Developed by the alliance of MasterCard and Visa (1997), SET is a group of protocols that supports only payment cards and relies on the existing credit card infrastructure. The SET protocol is implemented into a credit card using public key cryptography, e.g. RSA (Stallings 1999), and partly symmetric encryption techniques, e.g. DES (Stallings 1999). SET ensures a secure triple communication among seller, buyer and bank, so it was considered a practical
approach for easy, fast, and secure payments over the Internet (Asokan & Phillipe 1997). SET also provides privacy for the customers while enabling linking payment and purchase requests for dispute resolution. However, the implementation of the SET technology was unsuccessful. SET’s failure is mainly caused by the technology’s failure to convince the social groups (i.e. private and commercial customers) that it could be implemented without major technical difficulty (Kniberg 2002).

- Secure Payment Application (SPA)

SPA (Gpayments 2002) is an issuer-based authentication mechanism that uses the Universal Cardholder Authentication Field (UCAF) infrastructure of MasterCard. The SPA authenticates payments of customers and addresses the security of all parties involved in an online payment. SPA can be implemented as an externally hosted service with minimal system impact. SPA does not require the use of Public Key Infrastructure (PKI) (Stallings 1999, Centeno 2002a), which is one of the negative factors in the failure of SET.

- 3-Domain secure system

Developed by Visa International Service Association (2002), the 3-D secure system isolates the responsibilities of different parties involved in payments. The card issuers have close relationship with card holders (customers), while acquirers have close relationship with merchants. Communication between issuers and card holders, and acquirers and merchants, takes place during each payment. The three responsibility domains are: (1) the Issuer Domain that contains the customers (card holders) and their banks (issuers), (2) the Acquirer Domain which is the merchants and their banks (acquirers); and (3)
the Interoperability domain being the communication between issuing and acquiring parties using Visa’s infrastructure.

2.4.2 E-cash payment systems

The advent of e-cash (or so-called e-money or digital money) was to challenge the problems of security and privacy of credit cards. E-cash is intended to be both private and secure. There are two main types of e-cash systems: those in which value is stored in a software program on the user’s PC (e.g. ECash system) and those which are based on hardware such as smart card (e.g. Mondex).

- ECash

ECash (Wayner 1994) was developed based on the concept of the digital cash model proposed by Chaum et al. in 1990. The protocol is applied to online and pre-paid payment systems. It contains the properties of anonymity and untraceability which provide privacy to the customer based upon a blind signature (Chaum 1983). The ECash system requires customers to install special software called Electronic Wallet and deposit amounts of money in the ECash bank. Then, a customer needs to replace the money with digital coins in his electronic wallet. The digital cash does not link the relationship with the customer even when he uses it for shopping. The merchant checks only the digital cash to see if it is minted by the ECash bank which does not specify who spent the cash. Other similar systems are NetCash, CyberCoin, and so on.
• Mondex

The Mondex system (www.mondex.com) developed by NatWest bank is primarily working on a tamper-resistant smart card, which contains a microcomputer chip that enables the storage of data on the card itself. But Mondex requires a special device – card reader, which attaches to the user’s computer so that merchant and customer can use the card over the Internet. Different from the security mechanism of ECash which checks the double-spending problem online, Mondex checks the double-spending problem offline by using the smart card. Mondex cards can accept e-cash directly from the user’s bank account, and also supports fund transfer between users’ cards. One of the disadvantages of Mondex is that the card carries real cash in electronic form, creating a high possibility of loss and theft. This system, only operational in the U.K and France, has not gained wide acceptance due to its inconvenience and users’ resistance to the unfamiliar systems and additional hardware (Yang & Garcia 2003). Other similar systems are NetCard, CAFÉ, etc.

In general, e-cash payment systems usually require a customer to deposit money into his account in advance and, for each transaction, the bank checks the account to see if it has sufficient funds for payment. These pre-paid systems generally provide anonymity of the payee, but require both the customer and merchant to install special software or have additional devices.

Compared with credit card, e-cash systems have advantages in terms of privacy and anonymity, but also raise the greatest concerns about duplication of e-money, money
laundering, theft and loss. Like credit card systems, e-cash systems have drawbacks such as high transaction costs and inconvenience, especially when high volume and small payments are involved. So the needs for the systems that could be capable to handle micropayments arise.

2.4.3 Micropayment systems

The Micropayment Systems (MPSs) being reviewed in the literature, are now playing an indispensable role in the marketplace of e-commerce. More and more websites on the Internet rely on the customer’s ability to pay small amounts for information, services, music, games, and so on. Thus micropayment broadens the application area in the marketing of information distributed through the web.

Defining micropayments

The definition of micropayment varies with different researchers. One of the leaders – RSA, the founders of RSA algorithms had given the following definition:

“Micropayments are payments of small sums of money, generally smaller than those in which physical currency is available. It is used to pay for content access or for small quantities of network resources”

(RSA Security Inc. 2000).

However, there is no exact specification of how small a payment must be in order to be considered a micropayment. Initially, micropayment sizes starting at one cent have been discussed for years, however, current systems are focusing on the threshold at
ten cents to two dollars as the designed range of transaction sizes. Table 2.5 shows the transaction value differences between micropayment and other categories of payments.

**Table 2.5** Transaction value ranges of different payment categories

<table>
<thead>
<tr>
<th>Value / Payment</th>
<th>Minimum transaction value</th>
<th>Typical transaction value</th>
<th>Maximum transaction value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macro</td>
<td>$10.00</td>
<td>$50.00</td>
<td>$500.00</td>
</tr>
<tr>
<td>Mini</td>
<td>$2.00</td>
<td>$5.00</td>
<td>$10.00</td>
</tr>
<tr>
<td>Micro</td>
<td>$0.01</td>
<td>$1.00</td>
<td>$2.00</td>
</tr>
</tbody>
</table>

For these micro-transactions, it is still possible to use traditional payment networks, but that does not automatically imply that they are economically feasible or user-friendly. Especially when delivering low cost information with a value of only a few cents or even lower, such as downloading a picture file or an article in a magazine, the traditional e-payment approach is far from ideal. Usually, a credit card transaction costs between 20 and 30 cents in addition to 2% of the total value (Nasaw 2004), and an e-cheque normally incurs 10 cents on average. Those transaction costs will erode the profit margin of online content suppliers or may even be higher than the value of the products. Besides those overhead costs, the processing of the non-tangible low-value online content also requires, in most cases, a method of instant or real-time payment without major time lags (Shao et al. 2004).
Micropayment systems

It is obvious that MPSs are promising and efficient alternatives when credit card systems are economically unfeasible to transfer small amounts of money. Thus the efficiency leads to our definition (Shao & Nguyen 2005) for those systems:

“Micropayment systems are those electronic payment systems that support low-value payments for online products at low transaction costs with a minimal delay and instant delivery.”

Micropayment environment and models

In the literature, a general micropayment model has three entities (Hwang 2001): customer (the buyer), merchant (the vendor) and broker. The broker normally refers to the MPS. It could be provided and operated by a MPS operator, a financial institution or an Internet service provider. Figure 2.5 models a typical micropayment environment in which the relationships among three entities are maintained. It is assumed that the relationship between customers and merchants is short-term, irregular or occasional.

![Figure 2.5 Micropayment environment](image)
In micropayment environment, the MPSs receive macro payments from customers. Customers then make a large number of micropayments to merchants. Merchants receive redemptions from the system (or broker) in the form of macro payments. Consequently, aggregation method is considered to be the key to those systems: small payments are aggregated until they are settled in a macro payment (e.g. credit card payment, bank transfer) (Micali & Rivest 2002). Figure 2.6 illustrates the typical business model for the current operational MPSs.

![Figure 2.6 Business model for micropayment systems](image)

**Micropayment issues on system requirement and design**

The requirements for the MPS are not only their capability of handling small money at a low cost with a minimal delay, but also secure transaction. However, the risk and scale of loss, or fraud for consumers and suppliers is small. These systems, therefore, do not need features such as high security, non-repudiation, money-back guarantee, etc. (Chi 1997). In addition, compared with the traditional e-payment systems like credit card systems, there may be another requirement for micropayment, namely that
of user anonymity (Yang & Garcia 2003). Therefore, the design and techniques used for such a system are usually quite different from those of the traditional e-payment systems. Section 2.8 will describe the typical MPSs and analyse their transaction functionalities in further details.

2.5 Security and trust issues of electronic payment systems

2.5.1 Security issues

The security model used for present e-commerce and other client-to-server network applications is based on a public-key cryptography system using certificates issued by a trusted signing authority and an implementation of the SSL and TLS protocols.

The encryption algorithms available today for securing online transactions have been generally sufficient enough for electronic payment systems since e-commerce was first developed a decade ago. Although there are many arguments against the use of the weaker algorithms, such as MD5 and DES (still deployed in many applications), it is still unrealistic for any individual to break them today due to the cost of computation, and few people doubt that those weaker encryption algorithms are comparatively cost-effective.

However, securing a payment transaction requires both the secured communication channel and the verified identification of the users or parties (both customer and merchant). Cryptography could deal with both tasks. The former requirement could be easily solved by implementation of the SSL and TLS protocols, while the latte is not
as easy. Most transactions today are completed via identity check. It means that a single identity check of the server is conducted by the customer side, but the customer’s identity is rarely checked cryptographically by merchant. It is because a merchant may care whether the money is transferred and the product is delivered, but not really care who pays for it. The password authentication for the general customer identity check may not efficiently prevent fraud and identity theft. Therefore, it is a belief that the use of customer-side certificates will significantly reduce the opportunity for identity theft and attempts to commit fraud. However, issuing large amount of certificates to customers seems impractical and also inconvenient for widespread adoption. Customers also require to remain anonymous to merchants, especially when doing micropayments.

2.5.2 Trust issues

Many definitions for trust were proposed and examples could be found in (Mayer et al. 1995 and Hardin 2001). The majority of trust-related research was performed from the sociological and psychological viewpoints, such as (Doney & Cannon 1997, Rempel et al. 1985), but those theories could be also be relevant to present Internet payments. Many studies concluded that the lack of trust is the most likely inhibitor of e-commerce expansion today. Because trust strongly affects user’s decisions on conducting online payment, it is always regarded as the key to the success of Internet commerce. Trust must be established and maintained continuously throughout the entire e-business process involving activities such as online transaction, product tracing and feedback response, etc.
From the literature studied, it is found that trust is not just a simple idea, but a complex concept with a bundle of pre-conditions associated with it. Tan and Thoen (2000) presented a generic trust model that could be used for the design of trust related value-added services in e-commerce. This model identified that a transaction trust is a combination of trusts in control mechanisms and in the involved parties:

- **Trust and control**
  
  Use of control mechanism could increase trader’s trust, though the trust and control is regarded as a complementary relation (Holland and Lockett 1998). The general view is that the more there is of trust, the less there is of control. However, in contrast with this view is the concept introduced by (Das and Teng 1998) that the trust and control are parallel. They argued that if there is not enough party trust in a certain situation, then some control mechanism is prescribed. It could be easily explained by the SET protocol as an example. The SET protocol was developed as one of the most secure and suitable protocols for Internet transactions, aiming to increase the general level of trust in e-commerce, but the new protocol is too complicated to be understood and evaluated by users, which therefore, results in the lack of trust in the SET protocol itself.

- **Trust and party**
  
  Trusted Third Party (TTP) is often used to resolve the trust problems in Internet commerce. The TTP, such as a bank or a certificate issuer, is intended to be trusted by both customer and merchant due to the role it plays in the Internet transaction.
The current literatures about the trust in e-commerce also suggested that the trust might be also affected by the following factors:

- **Trust and system**
  
The first step towards building trust is to rely on the communication system, especially the electronic payment system. Such trust could be realized by a system consisting of security, reliability, availability and maintainability attributes (Jones *et al.* 2000, Dimitrakos 2001). For instance, any single weak or broken link could result in a complete loss of user trust in the new system. Trust depends on identity in the system (Daignault 2002). If there is a lack of trust in identity, then the entire foundation for the e-commerce will be meaningless.

- **Trust and security**
  
  Security is always the top concern for a trust-worthy system. To build trust in a new e-payment system, there is a need to ensure the system is able to detect fraud, so that users have full confidence with it. Such solution needs to cover the security of all parties involved. Applying a strong security mechanism in a payment system could definitely help users build up trust in that system and increase their confidence on doing online trading, but security is not the major determinant on user trusts.

- **Trust and communality**
  
  It is also argued by Tan and Thoen (2000) that the trust in e-payment systems is primarily based on commonality of trust. It could be explained by the current phenomenon that the credit card payment is still dominant in the
Internet payment market with little public understanding of its cryptographic technology and security of the protocol used. The success of the credit card protocol is totally reliant on the commonality of trust, even though there are so much questioning of its security weakness from the protocol developer’s point of view. The early IBM iKP, and later improved SET protocols, did address the security fragility left by credit card systems, but those protocols were not universally accepted within the industry and nor were the public convinced by the massive scale of SET applications.

- Trust and usability
  Better usability of e-payment system will definitely increase its usage, but usability alone will not determine its success. Few people would conduct transactions through an untrustworthy payment system, no matter how easy the system is to use. However, some researchers, such as (Ang & Lee 2000) show the fact that trust is improved by increasing usability. Lanford and Hubscher’s study (2004) also gave a list of guidelines on how to improve usability with a view towards improving trustworthiness.

- Trust and privacy
  Liu’s findings (Liu et al. 2005) indicated that privacy has a strong influence on whether an individual trusts e-business, and in turn influences user’s behavioural intentions to conduct transactions on the site. Similar finding was also disclosed in Smith & Shao’s survey (2007), which has shown that e-commerce is losing a significant amount of income partly due to the privacy concerns of its potential user base. They also concluded that to e-businesses,
the ability to give users’ control of their privacy in an attempt to create an acceptable level of trust is highly essential.

- Trust and risk

Trust is always associated with the risk of loss. Tan and Thoen (2000) found that trust is also determined by potential gain and involved risk. Compared with traditional e-trades with comparably large amounts of money involved in a transaction, the risk for an online micropayment (sometime with less than a dollar in value) is quite negligible. Though, the fraud of a micro-transaction is still unavoidable, but generally, a payment is considered safe when the cost of breaking its security is higher than the value of that payment.
Chapter 2 Literature review

Part 2: Micropayment systems and related issues

The widespread development of Micropayment Systems (MPSs) on the Internet has provided an alternative revenue source for merchants beyond advertising and subscriptions. Currently many online merchants are starting to use such systems and the critical mass of users will probably be reached very soon, especially in some industrial countries where Internet usage is pervasive. The potential market for online content with micropayments is huge, which was mentioned in Section 1.1.2. Therefore, there is a need to analyse this new ‘player’ that arises in our everyday life.

The main objective of this research is to ensure the inter-operability between existing MPSs by introducing the Universal Micropayment System (UMS) that interconnects these systems to make the use of cross-system micropayments easier and straightforward. Before discussing the design of this interconnection, it is important to get a good understanding of existing MPSs. After an initial analysis on the MPS technology in the previous chapter (Section 2.4.3), this chapter will further discuss the architectures and protocols that are being used for the payment transactions of the examined MPSs.

MPSs have been developed in the past decade, so this part will begin with the revolution of the MPSs presented in Section 2.6. Section 2.7 analyses the general technique and cryptography applied in the MPSs since the accuracy and security are always critical issues in payment systems. Section 2.8 contains a study of various
MPSs that are currently defined as the state-of-the-art in banking and other financial sectors. Finally, this chapter is concluded by presenting a detailed summary and identifying the fundamental differences of the examined MPSs.

2.6 Revolution of micropayment systems

Based on the literature, two generations of MPS are distinguished by Parhonyi et al. (2005). The first generation began around 1994 and lasted until the end of the 1990s. The developers of these systems primarily aimed at the introduction of the electronic form of cash (called e-coins or digital cash) on the Internet. Examples of the first-generation systems are:

- PayWord and MicroMint, developed by Rivest and Shamir (1996);
- MiniPay, developed by Herzberg and Yochai (1997);
- NetCard, developed at the Cambridge University (Anderson et al. 1997);
- iKP, developed by IBM (Bellare et al. 1995); and
- MilliCent, developed by Digital Equipment Corporation (Glassman et al. 1995), etc.

There exist a few account-based systems, which transfer money from customer accounts into merchant accounts similar to banking systems, some examples are:

- CyberCoin, developed by CyberCash Inc. in 1996;
- Mondex, developed by MasterCard in 1995; and
- CAFÉ, the ESPRIT Project (Boly 1994), etc.
All first generation MPSs failed one after the other, stopped after a public trial or remained at a theoretical level. The most important reasons for their failure, from the literature points of view, are (i) these systems were not deemed trustworthy, (ii) very low coverage and lack of funding until these systems reached a critical payment volume, (iii) inconvenient usage, (iv) lack of appropriate security mechanisms and (v) lack of anonymity (Kniberg 2002, Parhonyi et al. 2005).

The second generation (or current) MPSs emerged after 2000. These systems are almost without exception account-based. Most of them have their region-based customers and merchants. The system developers mainly focused on how to bind many small payments into one transaction to be performed on the existing transaction channel. Examples of those systems are analysed in detail on Section 2.8.

### 2.7 Micropayment techniques and cryptography

When payments are to be made over a telecommunications network such as the Internet, accuracy and security become critical. The design of MPSs is different from that of traditional e-payment systems in terms of techniques and security mechanisms.

#### 2.7.1 Early adopted micropayment methods and techniques

From the literature, the early micropayment researchers focused on the generation of e-coins or tokens, exchange in a secure, anonymous and untraceable manner, and validation and fraud avoidance. Therefore, they were mainly concerned with the following issues:
• how to minimize the online communications during a transaction in order to make the payment quickly and straightforwardly; and

• how to reduce the heavy computation for payment identification and authentication in order to minimize the transaction costs.

Based on the above issues, many possible solutions were proposed. Some solutions were further developed and put into the market. Among those early systems, some of the most typical and authoritative ones were PayWord, MilliCent and MiniPay, and they generated lots of imitators.

• PayWord (Ronald et al. 1997) used broker (bank) issued certificates to make it possible to keep the transaction offline from the broker. PayWord also first applied a hash function (Damgård 1990) to minimize the usage of the costly public key computation, because it was proved by Rivest et al. (1978) that hash operations, such as MD5 and SHA (Rivest 1992) are 100 times faster than RSA signature verification, and 10,000 times faster than RSA signature generation. Relying on the features (one-way random number generation) of the hash computation, the broker could authorise registered users to mint hashed-coin, called PayWord for transaction, and also assure the merchants for the redemption of PayWord. There were many other similar or improved schemes, such as micro-iKP (Hauser et al. 1996), NetCard (Anderson et al. 1997), SubScrip (Furche & Wrightson 1996), largely based on the PayWord proposal.
MilliCent (Glassman et al. 1995) offered a simple approach to deal with offline authorisation checking. It attempted to avoid contacting any third party to verify for a payment. MilliCent introduced a specific strip – a form of e-token that was only valid with a particular merchant for a limited period of time, so that it was optimized for repeated micropayments because payment processes were mostly offline and limited use of public key cryptography was required. However, the merchant-specified strip resulted in inflexibility and also required trusts that merchants would not cheat customers. NetCents (Poutanen et al. 1998) solved the problems of MilliCent, making the designated e-token available to any merchant, but required costly asymmetric cryptography and extra communications between merchants. The new version of MilliCent had been researched and developed by Digital Equipment Corporation since 2003, and released into the Japan market for trial usage.

MiniPay (Herzberg & Hilik 1997), another improved version of MilliCent, featured low cost and high security, including non-repudiation and protection against denial of service. MiniPay’s architecture involved multiple parties and it was based on peer to peer relationships, where public keys were exchanged and authenticated using existing relationships between the peers. Its solution required asymmetric cryptography and was somewhat similar to the Secure Electronic Transaction (SET) protocol (MasterCard & Visa 1997), a credit card payment protocol, but it did not require online authorisation for a payment. However, its solution could lead to an over spending problem.

Different from the above schemes, account-based NetBill (Sirbu & Tygar 1995) relied on a central trusted server that was involved in every transaction
for double spending detection to ensure a secure and partially anonymous real
time digital payment. NetBill’s approach was based on the well tested
Kerberos protocol (Cox et al. 1995), used a combination of public key
cryptography and a variant of symmetric key cryptography to ensure that all
its communications were secure, and all transactions were authorised.

In conclusion, the above mentioned schemes, in their early stages, either still relied on
asymmetric key applications or were more burdensome for brokers in terms of
minting coins and online verification. But to minimize transaction cost, most systems
applied lightweight security measures to some extent, making them not as secure as
traditional e-payment systems like credit card and e-cheque. This is because a balance
must be kept between the costs of implementing the security measures and the
protected value of the payment.

2.7.2 Currently adopted micropayment methods and techniques

With the rapid development of computer processors and data communication
bandwidth, computation is no longer a big issue. The new generation MPSs adopt a
digital signature computation and have made it a security standard. The computational
cost is quickly becoming a non-issue due to more powerful processors. In addition,
there are more secure and efficient signature schemes are currently available.

However, given the lessons from the previous system failures, to minimize user’s
resistance to new systems, currently operational MPSs turn to focus on user-friendly
design rather than on full security considerations. Today, aggregated billing methods,
stored value accounts, virtual card approaches and back-end applications are now coming together to make micropayment processing possible, popular and profitable. Some systems are also starting to support multi-currency payments and multiform charging methods like pay-per-click, pay-per-view, pay-by-minutes, etc., so as to maximally meet customers and merchants’ requirements. In the next section, several successful operational MPSs were selected for analysing the factors behind them.

2.8 State-of-the-art micropayment systems

In this study, second generation of MPSs, which were operational at the time of writing, are mainly focused. With the limitation of the thesis scope, only a few representative MPSs were presented in this section. These systems were selected from the payment systems repository of the Electronic Payment Systems Observatory\(^1\), the Google directories on MPSs\(^2\), Micropayment Technologies Press\(^3\), and EPayNews\(^4\). Other systems that are not presented here resemble, to a certain extent, the selected ones and will be listed in the summary (Section 2.9).

The time sequence of the payments performed by these systems and the parameters that are exchanged in various interactions are also discussed. The time sequence diagrams of the following systems were completed based on:

Other_Payment_Systems/Micropayments/
\(^3\) Micropayment technologies. http://www.micropaymenttechnologies.com
• the FAQ and system demonstration for customers and merchants;
• the implementation documents for merchants;
• using and testing the systems via their related websites;
• inspecting the source of web pages supplied by merchants; and
• contacting the related system support.

The following sub-sections describe the functionality of existing MPSs in terms of the payment methods used, transaction time sequence and messages exchanged.

2.8.1 BitPass

BitPass (www.bitpass.com) is a web-based MPS developed in 2002 at Stanford University. It is a pre-paid account based system that incorporates a “virtual card” payment concept, allowing the user to make a purchase for digital content.

Customers need to buy a “BitPass virtual card” with a specific payment (e.g. US$3 to $60) via PayPal or a credit card, then use their virtual card like a pre-paid phone card to make a purchase (McBride 2003). After that they could register to open a BitPass account directly from the website of content merchants (or vendors) which are already registered to BitPass. During registration they need to provide information such as email address and the bought card’s number. The created account will be accessible using the correct email address and password combination set during the registration. Later, customers are able to use this registration to pay other merchants. Customers can log into the payment system, acknowledge initiated payments, review the history
of their payments, buy more virtual cards and assign them to their account, or change their registration information. Customers do not pay for using the system.

Merchants also need to register first to receive an account. Merchants using BitPass need to follow an easy set up procedure and product registration. Then they add the resulting premium links to their websites. BitPass provides them with the gateway software, which will receive payment confirmations from BitPass and control the product access of paying customers. Merchants are paid out periodically. For very small items priced under $5, the fee is 15%. This makes very small prices quite feasible. For item prices over $5, the transaction fee is 5% plus $0.5 (Gillmor 2003). There is no set up or monthly fees. Currently, BitPass has already more than 200 online merchants registered (Reed 2003). Well known merchants participating in the use of BitPass include the scottmccloud.com that sells comic strips less than 1 dollar and bigfriendlycorporation.com that sells mp3 music files from 50 cents per track to 5 dollars per album.

The great advantage of this system is that it provides both the seller and customer with easy of use. For the customer, the BitPass virtual card allows them to buy items as they go without the need to keep a physical store-value card or have an e-wallet installed in their home PCs. The BitPass service integrates into the customer’s browser with no plug-ins or applications to download. For online sellers, BitPass has also simplified the processes, which only include: download a single file and upload it to their web server, change a few permissions and make BitPass enabled. However, the disadvantage to BitPass is that it requires a customer to know in advance how much he or she eventually wants to spend online.
BitPass payments

Suppose a customer encounters a content server and wants to purchase a piece of content. A click on the BitPass icon, next to the description and price of the selected content, will initiate a BitPass payment. The customer then needs to provide his/her email address and password in a BitPass login window. After that, BitPass requests the customer to confirm the payment. When the payment has been authorised, BitPass verifies the customer’s account balance. If the balance allows the new payment, BitPass sends, via the customer, a ticket with the payment confirmation to the gateway of the content server. Otherwise, the customer may need to transfer money to their account first. The gateway verifies this ticket and allows or rejects the access to the content.

From the BitPass documentation provided on its website, the BitPass time sequence diagram and its transaction parameters are simulated and shown in Figure 2.7 and Table 2.6, respectively.

The message sequence diagrams in Figure 2.7 illustrate the entities involved (vertical lines) and the messages exchanged between these entities (arrows). Messages with a Roman number (I, II, III etc.) occur before the transaction and do not occur for every payment. The other messages occur for every single payment.
Figure 2.7 BitPass time sequence diagram

Table 2.6 Parameters of BitPass interactions

<table>
<thead>
<tr>
<th>Sequence</th>
<th>HTTP message</th>
<th>Parameters (as named by BitPass)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Content request</td>
<td>HTTP REQ</td>
<td>Registered content ID (e.g. URL)</td>
</tr>
<tr>
<td>2. Send registered content info</td>
<td>HTTP RESP</td>
<td>URL</td>
</tr>
<tr>
<td>3. Payment initiate</td>
<td>HTTPS REQ</td>
<td>-same as sequence 2-</td>
</tr>
<tr>
<td>4. Login request</td>
<td>HTTPS RESP</td>
<td>ItemName, Amount, Duration, MerchantSite</td>
</tr>
<tr>
<td>5. Login response</td>
<td>HTTPS REQ</td>
<td>ItemID, TimeStamp, Ticket, EmailAddress, Password</td>
</tr>
<tr>
<td>6. Acknowledgement request</td>
<td>HTTPS RESP</td>
<td>N/A</td>
</tr>
<tr>
<td>7. Acknowledgement response</td>
<td>HTTPS REQ</td>
<td>N/A</td>
</tr>
<tr>
<td>8. Payment confirmation</td>
<td>HTTPS REQ</td>
<td>Merchant URL, Ticket</td>
</tr>
<tr>
<td>9. Send confirmation and request content</td>
<td>HTTPS RESP</td>
<td>Ticket and content URL</td>
</tr>
<tr>
<td>10. Content delivery</td>
<td>HTTPS RESP</td>
<td>Content download page</td>
</tr>
</tbody>
</table>
The information exchanged in the various messages is presented in terms of parameters of the messages. Table 2.6 contains the parameters that specify the type of the message, such as http request or http response.

Note that this system has not been fully tested and some of the parameters could not be discovered.

### 2.8.2 PepperCoin

PepperCoin (www.peppercoin.com), founded by two professors from MIT in 2001, is another micropayment processor that is also built on the existing payment infrastructure. It is a post-paid, account based MPS, which is based on cryptography, digital certificates and mathematical algorithms, aggregating many small purchases into larger credit transactions. To cut cost, it waits until there are enough purchases to make it worthwhile submitting the charge to the credit card company (Rivest 2004 & Geer 2004).

Merchants are required to register for using PepperCoin. Then they need to download an application called PepperMill to create so-called PepperBoxes. PepperBoxes are files that contain individual pieces of encrypted products together with product information (e.g. product identifier, product description, product type, and price). Customers can download these files, but cannot open them. If merchants receive payment information from customers, they will send this information to PepperCoin and a decryption key to the customers. Merchants are periodically paid out. Merchants pay a per transaction fee for using PepperCoin, only 7% to 9% of the transferred value
Among the well known merchants participating in the use of PepperCoin were musicrebellion.com, celebrityrants.com and bigfrankrecords.com.

Customers are also required to register. They need then to download and install an application called PepperPanel. This application will store authorisations from PepperCoin that the customers are eligible to pay merchants. This application is used to open PepperBoxes and pay for them. PepperPanel reads the product information stored in the file and allows customers to send the payment information to the appropriate merchants. After that, the customers receive the decryption keys to be able to extract and use the products. Customers then receive a list of completed payments, and the total amount spent on products is deducted from their credit card account provided during the registration. Customers use PepperCoin for free.

The disadvantage to PepperCoin is that it's complicated at the set up stage and requires both customers and merchants to download software. For online customers, credit card information is also required only once at their account sign up. The PepperCoin software installed in customers’ desktop is used to charge the purchase against customer’s account. So unlike BitPass system, PepperCoin is only suitable for usage at a home computer rather than at publicly accessed computers. Then PepperCoin bills their credit card when the total amount of charges makes the transaction worthwhile, generally once a month. However, compared with BitPass and PayPal, the standard charge rate of PepperCoin is lower (see Table 2.11). Recently released PepperCoin 2.0 has reduced the transaction expense to less than 10 cents (Jewell 2004).
PepperCoin payments

Suppose a customer downloads a PepperBox and opens it with his/her PepperPanel in order to pay for it. PepperPanel sends the customer’s payment information to the content server. The content server sends the information to PepperCoin and provides the customer with the decryption key. No money transfer occur immediately with every payment. PepperCoin transfers the money only on a small fraction of payments of a given content server (e.g. one money transfer occurs out of 100 payments initiated by all customers).

A statistical method (Rivest 1997) is used to select the payment that will be processed by PepperCoin. This method cannot be controlled by the customer nor content servers. The selection of a payment can occur in every 100 payments sent to one content server, but it can happen after 91, 105 or 122 payments. Then the value of the selected payment is multiplied with the serial number of the payment. For example, if a customer pays 20 cents for a music file and this payment is selected being the 100th consecutive payment received, then the content server will receive US$20 from the PepperCoin. At the same time, the customers’ credit card is charged to pay PepperCoin the aggregated value of the payments made since the last settlement.

In this way, a content server can receive a little bit more or less than PepperCoin collects from the customers. The developers of this system proved that the fluctuation of the amounts received by content servers balances out over the time. The statistics and encryption of PepperCoin ensure that the system remains fair to all parties in the long run (Rivest 2004). As a result, PepperCoin transfers fewer macro payments than the number of micropayments. This allows an important reduction of transaction fees.
Generally, the transaction fee would be around 27 cents on a 99 cents sale, which can be lowered below 10 cents using PepperCoin.

According to the PepperCoin technique related articles and documentation published on its website, the PepperCoin time sequence diagram and its transaction parameters are reflected in Figure 2.8 and Table 2.7, respectively.

![PepperCoin time sequence diagram]

**Figure 2.8** PepperCoin time sequence diagram
Table 2.7 Parameters of PepperCoin interactions

<table>
<thead>
<tr>
<th>Sequence</th>
<th>HTTP message</th>
<th>Parameters (as named by PepperCoin)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>HTTP REQ</td>
<td>ContentID (e.g. URL)</td>
</tr>
<tr>
<td>2.</td>
<td>HTTPS RESP</td>
<td>SessionKey, MerchantID, OrderDescr, Amount, secret key</td>
</tr>
<tr>
<td>3.</td>
<td>HTTPS RESP</td>
<td>SessionKey</td>
</tr>
<tr>
<td>4.</td>
<td>HTTPS REQ</td>
<td>-same as sequence 3-</td>
</tr>
<tr>
<td></td>
<td>HTTPS RESP</td>
<td>MerchantSite (name, address), Amount, OrderDescr</td>
</tr>
<tr>
<td>5.</td>
<td>HTTPS REQ</td>
<td>Username, Password</td>
</tr>
<tr>
<td>6.</td>
<td>HTTPS RESP</td>
<td>Product information (stored in PepperPanel)</td>
</tr>
<tr>
<td>7.</td>
<td>HTTPS REQ</td>
<td>N/A</td>
</tr>
<tr>
<td>8.</td>
<td>HTTPS RESP</td>
<td>Return payment record, and decryption key</td>
</tr>
<tr>
<td></td>
<td>HTTPS RESP</td>
<td>Return payment ID, time</td>
</tr>
<tr>
<td>9.</td>
<td>HTTPS REQ</td>
<td>Return decrypted URL</td>
</tr>
<tr>
<td>10.</td>
<td>HTTPS RESP</td>
<td>Content download page</td>
</tr>
</tbody>
</table>

2.8.3 ClickandBuy

FirstGate, German-based E-wallet provider, launched ‘ClickandBuy’ micropayment service (www.clickandbuy.com) for Internet content and e-service in 2000. ClickandBuy now works with about 14,000 online providers of articles, information, games and other content, and has 8.65 million customers (ClickandBuy 2009a). ClickandBuy has recently entered into music industry, providing e-payment solution for iTune in Europe, processing deposits and withdrawals in several currencies (ClickandBuy 2009b). Their service is available in more than 20 countries, making ClickAndBuy one of the biggest micropayment solution providers.
The ClickandBuy system is a post-paid, account based MPS. More advanced than other similar systems in terms of security mechanisms, ClickandBuy uses passwords and Secure Sockets Layer (SSL) encryption, and also records the IP addresses of customers’ machines for reference in case problems occur (Geer 2004). Like PepperCoin, to reduce processing costs, ClickandBuy aggregates customers’ micro-purchases across merchants to enable a single processing of multiple transactions. ClickandBuy bills customers’ credit cards, debit cards, or phone accounts, once they have accrued a few dollars in charges, and then remits money to merchants via checks or bank transfers.

Customers are required to register and provide personal and banking information (e.g. credit card number) in order to open a post-paid account with ClickandBuy. Customers can access their accounts by opening a session (i.e. logging into the system using a user name and password combination set during the registration). During a session they can acknowledge initiated payments, view their payments, check the balance of the account, and change the information provided at the registration. Customers can also acknowledge payments to pay for subscriptions. In this case, payments will be performed periodically and automatically. Customers receive periodically an indication that a money transfer took place, which restored the balance of their accounts. The money transfer is initiated by ClickandBuy using the information provided at the registration. The period can be changed depending on the number and volume of the acknowledged payments. Using ClickandBuy is free for customers.
Merchants also need to register and pay a one time subscription fee for using ClickandBuy. Merchants using ClickandBuy need to follow a set up procedure and product registration similar to the one described for BitPass (see Section 2.8.1).

During the registration, the access path (or link), description, price and availability of the product (measured in time) is provided to ClickandBuy. Then they need to set up their websites on which they offer their products. Because ClickandBuy is an accounting system, merchants need to register their products, and then protect those products such that only ClickandBuy will have access to it. In return, the merchants receive a premium link for each registered product. These links will be added to their web pages. Products can only be sold in individual unit, which means that customers cannot select multiple products and pay in one amount. The prices of products vary between 0.1 to 5 Euros because these are the minimum and maximum payment values supported. Merchants will be able to see the completed payments in a management environment provided by ClickandBuy. ClickandBuy will further handle the payments, retrieve the paid product from the merchants and deliver it to the customers. Hence, merchants do not receive indications of the successfully completed payments.

Same as PepperCoin, the money received from customers is paid out monthly to the merchants that also receive a detailed list of completed payments. Because merchants need to pay for the payment service, the commission will be deducted from the total amount to be transferred. Also merchants are charged an up-front fee for set up, integration, and consulting. Standard rates for transaction are not clear at this stage. But the user-friendly design of ClickandBuy enables online merchants to implement
single or multiple Internet pricing models, such as per-per-click, pay-per-minute, transaction module (connection to a shopping cart system), subscription module and donation module.

**ClickandBuy payments**

Suppose a customer encounters a content server and wants to buy a piece of content. The customer decides to pay and clicks on the ClickandBuy icon next to the description and price of that piece of content, and initiates a payment. ClickandBuy opens a login window. In this window the customer sees the necessary information about the payment, and then he/she enters their user name and password to open a session. After that, ClickandBuy requests the customer to acknowledge the payment. Then the payment is performed and ClickandBuy retrieves the paid content and delivers it to the paying customer. The customer does not receive confirmation, but instead gains immediately access to the paid content.

In case a customer did not get the content, or the content is other than it was described, the customer needs to contact via email the concerned content server or the payment system operator and lodge a complaint. After analysing the complaint, it is possible to refund the customer. It is either ClickandBuy or the content server that initiates the refund. A similar system is WebCent (webcent.web.de) introduced in Germany.
2.8.4 Wallie

Wallie (www.wallie-card.nl) is a pre-paid account based MPS introduced in the Netherlands by the Distri Group, the biggest distributor of pre-paid telephone cards. Wallie can only be used in the Netherlands. Wallie includes an issuer that issues pre-paid cards, an acquirer that redeems payments made with the issued cards, an identification service provider that identifies the source and destination accounts of each transaction and a branding organisation.

Customers can buy Wallie cards starting from 5 Euros at one of the 3500 outlet stores (Bank of International Settlements 2004). The card contains a pre-paid account number, which consists of 16 digits and 3 letters. The customer initiates Wallie payments and receives confirmations of the completed payments. Additionally, the customer can verify the balance of the pre-paid account in a new session by entering the account number on the website of Wallie. The history of completed payments is not provided to the customer, however. The account cannot be over spent. The customer bears the financial loss if they lose the card. Wallie is free of charge to customers.

Merchants need to register to use the payment service of Wallie. Merchants need to set up their websites to redirect customers to the payment system. The merchants send customers the payment information (e.g. merchant identifier, a product identifier, value of payment expressed in euro cents), which is then sent to Wallie to initiate a payment. After the set up procedure, merchants will also be able to receive notifications from Wallie about the completed payments. Each payment is referred to
the appropriate merchant. Merchants can also review, in real time, the information about completed payments. Each merchant receives periodically (monthly) a detailed list of all payments, and Wallie transfers the money received from customers to the merchant’s account specified during registration. Each merchant pays per transaction costs, which are deducted from the total amount of money transferred to the particular merchant. Because Wallie is a pre-paid system, merchants do not face the risk of losing money.

**Wallie payments**

Suppose a customer encounters a content server to buy a piece of content. The customer decides to pay and clicks on the Wallie icon present on the website of the content server. With this click, the customer initiates a Wallie payment. Wallie will open a new window for the customer to provide the identification information. In this window the customer receives information about the content server and amount of money to be paid, and then he/she discloses the account number on the card. If the account number is correct, the customer is asked to acknowledge the payment. After the customer’s acknowledgement is given, the balance of the account is verified by Wallie. If no over spending found, Wallie performs the payment and then indicates to both the customer and content server of the successful transaction. The customer receives this indication in the form of a printable receipt, which contains the content server’s name, a unique payment transaction key (generated by Wallie), the date, time, and value of the payment, and then the payment session is automatically closed by Wallie. The content server receives the notification of a so-called “callback-URL”, which contains the transaction key (provided to the customer as well), the content
identifier and the value of the payment. After that the content server can send the content to the customer.

If the balance of the account is lower than the amount to be paid, then the customer may provide a second account number and open a new session. A maximum of 150 Euros can be paid in one transaction. The Wallie time sequence diagram and its transaction parameters are shown in Figure 2.9 and Table 2.8, respectively.

Like the BitPass system (Section 2.8.1) in the U.S., the main advantage of the Wallie system is that customers do not have to register and do not have to download and install special software to be able to perform micropayments. Additionally, customers can remain anonymous to merchants. Unlike BitPass’s virtual card, a major disadvantage of Wallie is that the online buying of cards is not supported so that customers need physically to go to a store to buy the cards and face the risk of losing them.

Similar systems are, for instance, PaySafeCard (www.paysafecard.com) offered in Austria and Germany, Micromoney (www.micromoney.de) introduced by Deutsche Telekom in Germany, Centipix (www.centipix.com) designed for downloads of JPEG images, PayAsYouClick (www.payasyouclick.com) dedicated for adult sites in the U.K. and Mywebcard (www.mywebcard.dk) used in Denmark.
Figure 2.9 Wallie time sequence diagram

Table 2.8 Parameters of Wallie interactions

<table>
<thead>
<tr>
<th>Sequence</th>
<th>HTTP message</th>
<th>Parameters (as named by Wallie)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Content request</td>
<td>HTTP REQ</td>
<td>Content ID (e.g. URL)</td>
</tr>
<tr>
<td>2. Send payment info</td>
<td>HTTP RESP</td>
<td>Wallie URL, MerchID, ShoppingCartID, Amount</td>
</tr>
<tr>
<td>3. Payment initiation</td>
<td>HTTPS REQ</td>
<td>Same as sequence 2</td>
</tr>
<tr>
<td>4. Login request</td>
<td>HTTPS RESP</td>
<td>MerchantName, Amount</td>
</tr>
<tr>
<td>5. Login response</td>
<td>HTTPS REQ</td>
<td>AccountID, Password</td>
</tr>
<tr>
<td>6. Acknowledgement request</td>
<td>HTTPS RESP</td>
<td>MerchName, Amount, AccountId, CurrentAccBalance, NewAccBalance</td>
</tr>
<tr>
<td>7. Acknowledgement response</td>
<td>HTTPS REQ</td>
<td>ButtonValue</td>
</tr>
<tr>
<td>8. Payment confirmation</td>
<td>HTTP REQ</td>
<td>ResultCode, ShoppingCartID, TransDate, TransTime, Password, TransferAmount, TransferCosts,</td>
</tr>
<tr>
<td>9. Payment confirmation</td>
<td>HTTPS RESP</td>
<td>MerchName, Sleutel, Date, Time, Amount, ThanxURL</td>
</tr>
<tr>
<td>10. Use ReturnURL</td>
<td>HTTP REQ</td>
<td>ThanxURL</td>
</tr>
<tr>
<td>11. Content delivery</td>
<td>HTTP RESP</td>
<td>Content download page</td>
</tr>
</tbody>
</table>
2.8.5 PayPal

Founded in 1998, acquired by eBay since 2002, PayPal (www.paypal.com) enables any individual or business with an email address to securely, easily and quickly send and receive payments online. PayPal's service builds on the existing financial infrastructure of bank accounts and credit cards. PayPal has gained 64 million account members in 45 countries around the world.

PayPal’s processing cost is high for micropayments. For business accounts, a standard rate (2.9% of transaction amount plus US$ 0.30) for receiving payment is applied, and a cross border fee of 1% is additional when payments are received from another country. However, since working with Apple iTune, PayPal created a new fee (2.5% plus 9 cents) exclusively for digital music retailers (Naraine 2003), making itself one of most competitive micropayment players.

Customers are required to register for using PayPal. A customer can perform registration at a PayPal website or through the registration link on merchants registered with PayPal. During the registration they need to provide personal, email, and credit card information. After that they can open a session by logging into the system using their email address and a previously set password. After registration they could also transfer money into their PayPal accounts. PayPal also supports person-to-person (P2P) payment. After a sum of money is transferred to a PayPal account, a registered user can then transfer any of this money to anyone else with an account at PayPal, simply by entering the recipient's email address and the payment amount.
Payments between individuals are free, so that PayPal users can trade with PayPal users in eBay, since payments are all backed by the robust existing systems of the likes of Visa and MasterCard.

Unlike in the case of ClickandBuy and BitPass, the merchants need only a short set up of their websites to allow customers to pay for music with PayPal. For each song, they add information such as the name of the album, artist, name of the song file, price, and two URLs in case the payment is successful or rejected.

**PayPal payments**

When a customer decides to pay for downloading a music file and clicks on the PayPal icon next to the price of the selected MP3, a login window is opened for users to initiate the payment. In this window the customer sees the name of the content server, and the amount to be paid. They then enter their email address and password to open a payment session. After the customer initiates the payment, the customer receives a confirmation, which is provided to the content server. Then the content server sends the content to the customer. Also PayPal provides an additional confirmation to the customer in an email. The payment is then charged from customer’s PayPal account. The PayPal time sequence diagram and its transaction parameters are shown in Figure 2.10 and Table 2.9, respectively.
Similar systems are, for instance, PayNova (www.paynova.com) served exclusively in the U.S., PayStone (www.paystone.com) used in Canada, and TechnoCash (www.technocash.com) used in Australia and New Zealand.
Table 2.9 Parameters of PayPal interactions

<table>
<thead>
<tr>
<th>Sequence</th>
<th>HTTP message</th>
<th>Parameters (as named by PayPal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Content request</td>
<td>HTTP REQ</td>
<td>Content URL</td>
</tr>
<tr>
<td>2. Send payment inf.</td>
<td>HTTP RESP</td>
<td>PayPal URL, MerchantId, MerchantName, TransactionID, ItemName, Amount, Currency, SuccessURL, FailureURL</td>
</tr>
<tr>
<td>3. Payment initiation</td>
<td>HTTPS REQ</td>
<td>-same as sequence 2-</td>
</tr>
<tr>
<td>4. Login request</td>
<td>HTTPS RESP</td>
<td>MerchantID, TransactionID, ItemName, Amount, Currency, SuccessURL, FailureURL</td>
</tr>
<tr>
<td>5. Login response</td>
<td>HTTPS REQ</td>
<td>EmailAddress, Password</td>
</tr>
<tr>
<td>6. Acknowledgement request</td>
<td>HTTPS RESP</td>
<td>MerchantID, TransactionID, ItemName, Amount, Currency</td>
</tr>
<tr>
<td>8(a). Payment confirmation</td>
<td>HTTP RESP</td>
<td>MerchantName, MerchantID, Amount, Currency, ItemName, TransactionID, PurchaseID(PayPal ref No. for customer), TotalAmount</td>
</tr>
<tr>
<td>8(b). Payment confirmation</td>
<td>HTTPS REQ</td>
<td>MerchantName, MerchantID, Amount, Currency, ItemName, TransactionID, OrderID(PayPal ref No. for merchant), TotalAmount</td>
</tr>
<tr>
<td>9. Use ReturnURL</td>
<td>HTTP REQ</td>
<td>SuccessURL (or ErrNo, ErrDesc if failure)</td>
</tr>
<tr>
<td>11 Content delivery</td>
<td>HTTP RESP</td>
<td>Content download page</td>
</tr>
</tbody>
</table>

2.9 Summary and analysis

This section presents the summary and analysis of the business roles and functional characteristics of the investigated MPSs. It also includes characteristics of other studied MPSs, which were not presented in detail in the previous sections.
2.9.1 Summary of business roles and functional characteristics

The summary of studied MPSs are presented in Table 2.10, in which each ‘x’ mark implies that the payment system contains the marked role or certain characteristic, while ‘-’ mark means it does not contain the role or characteristic. Because there are few MPSs that support only merchant’s initiated payments in a B2C context, this category is not considered. There is no column for the double confirmed payments; instead an ‘x’ mark is placed in the columns of customer and merchant’s confirmed payments. The ‘n/a’ notation in the cell means that it is unclear or no information about the system is available, or the characteristic could not be checked.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BitPass</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>US$</td>
<td>0.10</td>
<td>60</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>PepperCoin</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>US$</td>
<td>n/a</td>
<td>20</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>ClickandBuy</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>€</td>
<td>0.10</td>
<td>5</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Wallie</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>€</td>
<td>1.00</td>
<td>150</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PayPal</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>US$</td>
<td>0.99</td>
<td>100</td>
<td>0</td>
<td>x</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Centipix</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>US$</td>
<td>0.50</td>
<td>50</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PayAsYouClick</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>GBP</td>
<td>0.10</td>
<td>100</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PayNova</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>€</td>
<td>0.10</td>
<td>n/a</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PayStone</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>US$</td>
<td>0.25</td>
<td>500</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.10 Business and Functional characteristics
Table 2.11 presents the transaction or service fee that each system charges merchants for using their service. In most situations, there is no cost to customers for using the systems to pay for the content.

Table 2.11 Fee comparison of different systems

*(Partly adopted from Shao & Nguyen 2005)*

<table>
<thead>
<tr>
<th>Micropayment processors</th>
<th>US currency</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BitPass</td>
<td>15% (total value) for purchase up to $5</td>
<td>Fee 1</td>
<td>Fee 2</td>
</tr>
<tr>
<td>PepperCoin</td>
<td>7~9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ClickandBuy</td>
<td>20% or negotiated monthly fee applied</td>
<td>Fee 1</td>
<td>Fee 2</td>
</tr>
<tr>
<td></td>
<td>5% + 6c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wallie</td>
<td>20%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PayPal</td>
<td>2.5% + 9c for music sites only</td>
<td>Fee 1</td>
<td>Fee 2</td>
</tr>
<tr>
<td></td>
<td>2.2~2.9% + 30c for other operators</td>
<td>Fee 1</td>
<td>Fee 2</td>
</tr>
<tr>
<td>Centipix</td>
<td>Monthly fee applied</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PayAsYouClick</td>
<td>Monthly fee + 5%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PayNova</td>
<td>n/a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PayStone</td>
<td>2.8% + 10c below $5</td>
<td>Fee 1</td>
<td>Fee 2</td>
</tr>
<tr>
<td></td>
<td>2.8% + 30c over $5</td>
<td>Fee 1</td>
<td>Fee 2</td>
</tr>
</tbody>
</table>

2.9.2 Analysis of functional characteristics

In this section, the systems’ functional characteristics were analysed at a high abstraction level. These characteristics only consider the interactions between users of the system and the system itself. Indeed, there are various interactions during a payment transaction.

Based on the investigated MPSs, those interactions could be divided into two types: local interactions and payment interactions. The former type includes interactions such as registration, login, reviewing of payments history, checking account balance,
etc. As they have only local significance for the customer and merchant, so they are not further considered. The latter involves the payment imitations and confirmations. We abstract from the different ways in which the initiation and confirmation of payments can be implemented, because only the results of these interactions are important for the interconnection problems that this study is aiming to address. It is discussed in detail as follows.

**Payment initiations**

According to the payment scenarios of the studied MPSs, the payment initiations, in most cases, are performed by one or both system users, who are providing adequate payment information during the process. Generally, the systems used by merchants generate the largest part of the payment information, because these systems determine the price of the products a customer should pay. The generated information usually contains a unique merchant identifier, which is known by the payment system, and that identifies a destination account where money should be transferred. The payment information also contains the value and currency of the payment, and a unique product transaction identifier that is used by the merchant to identify the product transactions, so the paid products can be delivered. Nevertheless, a customer identifier, which is known to the payment system, and that identifies a source account from which money should be transferred, is also needed for each payment. This information is added to the previously generated payment information.

The methods of performing payment initiations differ. Concluded from the investigated systems, three alternatives were identified to initiate a payment, shown in Figure 2.11. The vertical lines represent the interaction points between a MPS and its
users, arrows represent payment initiations, and the direction of the arrows indicates the direction of the information flow.

![Diagram of payment initiations](image)

**Figure 2.11 Payment initiations**

- The first alternative is that the customer initiates a payment. Prior to this, the merchant provides the payment information to the customer. Most investigated systems could be grouped into this category.
- The second alternative is that the merchant initiates a payment. Prior to such an initiation, the merchant receives identification information of the customer.
- The third alternative is that both users initiate a payment. Because both users initiate the same payment, they both have to provide (partial) payment information such that the payment system can correlate the received information, and then make the payment. This means that both users need to interact with each other before the initiation in order to exchange parts of the payment information.

It was found that customers initiate the payments for most existing systems. Regardless of the initiation type, the most common parameters of initiations are the customer and merchant identifiers, the product transaction (or order) identifier and the
amount of money (value and currency). The other parameters of the payment initiations may differ but have local significances to the system in use, they are less important.

**Payment confirmations**

Completed payments can be acknowledged or confirmed to the customers and merchants in two ways: (1) the MPS provides a confirmation to the customer and/or merchant; and (2) the MPS provides no explicit confirmation, instead it retrieves the paid product from the merchant, and delivers it to the customer.

Similar to payment initiations, three possible solutions to confirm payments were identified, they are displayed in Figure 2.12.

![Diagram of Payment Confirmations](image)

**Figure 2.12 Payment confirmations**

- First solution is that only the customer is confirmed. In this case, the customer will then send the received confirmation information to the merchant in order to obtain the paid product.
• Second solution is that only the merchant receives a confirmation. In this case, the merchant sends the received confirmation information to the customer and/or delivers the paid product.

• Third solution is that both the customer and merchant receive confirmations. In that case, the confirmations may take place concurrently, but there may also be temporal differences between them. Usually no additional interaction is needed between the customer and merchant, and the latter one will deliver the paid product.

In the most investigated cases, merchants receive payment confirmations. The most common parameters of confirmations are the product transaction identifier (or context of payment) and payment identifiers (or transaction number, or ticket identifier). Other local parameters of the confirmations are specific to the MPS in use, they are less considered in further study.

**Range of payments and payment methods**

The minimum and maximum values, and the currency of money transfers are those functional characteristics that define the range of payments accepted by a MPS, and that will be of great importance when interconnecting multiple MPSs. The minimum payment value supported by the payment systems may differ and this may create incompatibility between some systems. The maximum payment value creates fewer problems, because a bigger payment may be broken up into smaller payments. Generally, the payment systems support only one major currency (e.g. US dollar $ or Euro €), only a few support other or multiple currencies.
As shown in Figure 2.11 and 2.12, the great majority of examined MPSs are pre-paid and only a few are post-paid type. For the latter, customers still need to present a source of money (such as credit card information) before initiating payments. Among the reasons of the tendency to deploy pre-paid method is the need to avoid over spending and limit the fraud possibilities by guaranteeing the payments to merchants. It is also important to notice that post-paid systems require a long term contract with customers, which makes it more difficult for younger people to become users of a post-paid system.

2.10 Chapter summary

MPSs still do not have the same properties as cash: widespread acceptability, guaranteed payment, no transaction fee and anonymity. The majority of current systems have no cost at the customer side, while merchants need to pay for using the system. However, current practice shows that each system lets customers bear the risk of losing their money, and the acceptability of current systems is limited by the fact that many systems function within national borders only.

The earlier MPSs generally provided high security with low cost, but most of them gained little users’ support and disappeared from the market, and few have survived. Until recently a new wave of MPS competition arrived with the thriving of the digital music industry. At the time of writing, however, none of the presented MPSs had a breakthrough success. No wonder why some researchers argue the existent necessity of the MPSs. Among them, for instance, Shirky’s ‘mental cost in micropayment'
theory (Shirky 2000) was controversial. His research shows that micropayment approach – transferring a few cents to a dollar for paid content, could not be acceptable in a broad scale, because users prefer predictable and simple, flat rate pricing instead of pay-as-you-go pricing schemes. Putting those theories aside, it is clear that the biggest barriers to micropayments are not problems of security and technology issues, but user approval.

The ‘chicken and egg’ problem, whether the customer comes first or the merchant does, has no longer been debated, because the market and users already exist and keep growing. According to a recent market research, more than 10 million American customers purchased digital content for less than $2 in 2005, a near 150% increase over the previous year (Turner 2005). There is no doubt that the online payment giant PayPal, which was designed for large value Internet transaction, now has a set foot in the micropayment market. Nevertheless, learning from the failures of the previous systems, the new generation MPSs have much better chance to be successful than their predecessors.

The literature found that most Micropayment System (MPS) operators at present have implemented their Internet commerce computing environment using web technology. They usually present web applications for easily delivering specific payment functions to attract online merchants and customers. Due to different functions having located in diverse backend systems used by different payment service providers/operators, new problems rise: many competing MPSs have maintained their own users (both merchants and customers); no one has yet gained global acceptance
like online credit cards have; and then as a result, users are forced to use multiple MPSs to pay and receive money.

Consequently, doubt increases. Do merchants and customers need to use multiple systems to reach each other? Is there any ideal solution for all stakeholders – merchants, customers and system operators? Those problems need to be addressed in order to maximise the user approval.

In the next chapter those problems, possible solutions and related research questions will be further explained, and the research approaches to achieve the ideal solution will be highlighted.
Chapter 3  Research methodology

This chapter is to define the research approach and methodology for this study. In the literature review chapter, the current issues on Micropayment Systems (MPSs) and a desired solution were briefly discussed. In this chapter, Enterprise Architecture (EA) approach and its framework are introduced; the purposes of using this approach and how this approach works for the research, as well as the main questions addressed by this approach are highlighted in Section 3.1. According to those questions identified at different levels of EA, we constructed Section 3.2 to 3.4, where the current MPS situation and problems are identified based on the findings in the Literature Review Chapter, as well as a desired solution, is presented in Section 3.2. To address those issues, the possible approaches to achieve the desired situation are analysed and a suitable approach is determined, and then followed by the expected outcomes of the determined approach presented in Section 3.3, where a proposed system – Universal Micropayment System (UMS) is also introduced. To realize the proposed system, the approaches to system development are analysed in Section 3.4. The detailed process of the UMS architecture design is given in Section 3.5, where five stages of the design cycle are identified. Section 3.6 summarises the chapter.

3.1 Enterprise Architecture approach

Enterprise architecture (EA) is an approach to align business and IT within an organisation. This approach has been adopted in this research. EA can be defined as
(West et al. 2002) “the blueprint that documents all the information systems within the enterprise, their relationships, and how they interact to fulfil the enterprises mission.”

The primary purpose of using the EA approach is to ensure the alignment of IT strategies and current business. In short, from the business point of view, using this approach is to ensure that investment spent on IT can be justified. Another identical purpose is that the process in every level of the EA blueprint could be developed by common modelling techniques. This would help to bridge the gap between people from different disciplines.

3.1.1 EA framework

EA is characterized by the use of frameworks that support the analysis of the enterprise from the business level down to the IT level (Vasconcelos et al. 2004). After Zachman (1989) first introduced the framework for information systems architecture, his framework has then been further enhanced by Sowa and Zachman (1992) and commonly accepted in EA related projects. Businesses describe EA using a framework, which is a structure for documenting the architecture of their IT systems.

The adoption of the EA framework has given several benefits to this study:

- First, it gave a complete view of all activities involved in providing e-payment services for the e-commerce organisation;
- Second, it showed all entities involved in providing e-service and their relationship;
Third, it expressed EA design principles for all entities; and
Fourth, it organised and structured what needs to be done in the organisation and its’ related entities in order to satisfy the organisational goals.

The EA frameworks used in this study have addressed the elements of business strategy, guiding principles/design framework, requirements, overall business process, methods, standard, techniques, and hardware/software tools, etc. The decisions related strategy, business goals, information needs, data mapping, selection of system-independent application and tools need to be guided to ensure maximum effectiveness and efficiency for the organisation.

Figure 3.1 demonstrates the general EA framework for this research; it also models the life cycle of the EA approach applied in this study.

**Figure 3.1 The life cycle of EA approach adopted in this study**
3.1.2 How EA approach works in this research

This research includes various elements such as innovations in the system design and integration, expansion on business scale, smoothness of business processes, the quality and timeliness of information exchange, etc. EA approach here presents all elements involved as an entire body.

Generally, EA involves multiple disciplines. EA approaches could range from highly abstract approaches focusing on the enterprise as a whole, to approaches aimed at specific aspects of the EA in detail (FAWG 2007). Thus, a general enterprise approach requires various architecture levels, shown in Figure 3.2, which provide different perspectives and address different but related details.

![Figure 3.2 Enterprise Architecture levels](image)

The above diagram also sheds light on the relationship among enterprise architecture, business architecture and solution architectures. The software/hardware architectures, database architecture or network architecture are partial contributions to the solution architecture disciplines, which have been highly focused in this study and will be presented in detail from Chapter 5, 6 and 7.
Section 1.3 has presented many research questions, and they can be summarised into the following three main questions. EA approach used in this study is to answer those questions:

- What system is required? (Enterprise level)
- What should the system looks like? (Business level)
- How to develop and measure the system? (Solution level)

The first question is based on the identified problems and ideal solutions from the stakeholders’ point of view; the second question focuses on the elements in the system requirements and their structure, so that the architecture needs to have the models and tools for producing the system descriptions; while the third question is solved by defining appropriate approaches to realize the system.

3.2 Current issues and desired micropayment environment

At the enterprise level, all stakeholders including online merchants, customers and system operators should be considered. The current situation needs to be analysed and the major problems identified, as well as the ideal solution needs to be addressed. Section 1.2 has briefly described current problems and ideal solution; the following sub-sections will explain those issues in a further detail.
3.2.1 Current situation of MPSs

Concluded from the literature review part, it appears that the systems that support small value payment transferred online between customer and merchant are still in an increasing demand for Internet commerce. Many MPSs have already been proposed and some have been commercialized for the micropayment market. However, until now, no MPS has managed to reach a dominant position among customer and merchant. Most commercially operational systems are currently being used within restricted communities with registered merchants in the same business area, nor support multi-currency or cross-system transactions.

The current situation in the Internet commerce also shows that different merchants use different MPSs. This is driven by the fact that merchants are trying to reach a maximum number of customers by using multiple payment systems. For instance, a survey conducted in the US reported that one third of merchants that supply low-value content use more than three different MPSs to receive online payments, another one third use two, and less than one third still stick to a single MPS for their business (Card 2005). Similar practice is experienced by customers too; they also need to register with multiple MPSs to be able to buy online content from different merchants. Figure 3.3 illustrates the current situation of micropayment for low-value content on the Internet.
3.2.2 Problem statement

The current situation causes various problems that all stakeholders (customers, merchants and system operators) may run into. Those problems could be user-trust related issues and user-friendly related issues. The former one might be the critical concern because it directly results in the success or failure of a system. Building customers’ trust on a system that is new to them is a tough and long term process. After sticking to one payment system, switching over to another system seems a harder decision making process for customers. Similar to merchants, merchants are forced to trust and register with the various service brokers that operate those MPSs. The latter might involve the problems encountered by both customer and merchants, such as: (a) the need to install multiple software packages (e.g. e-wallet on the customer side and web-based payment portal on the merchant side), and sometimes hardware devices; (b) manage multiple accounts or e-wallets, and keep in mind differently formatted passwords; (c) learn the usage of several MPSs and contact
multiple help lines in case of difficulties, and so on. Therefore, the effect of such inconveniences in the long run might cause the users’ to be resistant to using these e-payment systems frequently, though the micropayment market needs huge volume of transactions for it to survive.

3.2.3 Desired micropayment environment

To encourage the user’s willingness (both customer and merchant) to keep using MPSs, building an easy to use payment environment is a prerequisite. A desired micropayment environment from the customer’s point of view is that they be able to use a single system or payment platform for paying all merchants, regardless the MPSs used by merchants. Of course, different customers could choose to use different systems. A similar environment is expected on the merchant side, that a merchant could use one system of their choice for receiving micropayments from all customers, regardless the MPSs used by customers. Figure 3.4 illustrates the desired micropayment environment from both the customer and merchant’s perspectives.

---

**Figure 3.4** Desired micropayment environment from users’ perspectives
3.3 An analysis of system design approaches

At the business architecture level, the main research question arises regarding the current micropayment issues and user’s expectations. From system operators’ viewpoint, the question is how the desired micropayment environment could be realized, and what the ideal system looks like?

Based on the literature review conducted for this research, we could identify two approaches that may possibly address those issues and achieve the desired solution for both system users and operators. The first one is the market-driven approach; it is to adopt an existing MPS or create a new MPS that could be globally introduced and accepted as a unified platform, just as the credit card system does. The second is the service-oriented approach, where it is to create a unified platform that existing MPSs could incorporate with to provide a universal micropayment. Both alternatives impose a shift in the ordinary way of performing a micro transaction. Considering the factors such as cost and time efficiency, ease to implement, local regulation and standards related issues, the most suitable approach will be determined after the following discussion.

3.3.1 Market-driven approach

This approach could be easily conceived by a system designer as it is a common idea drawn from the success of the credit card system. However, this approach seems unlikely to be realized in the near future due to the following two reasons.
First, making the customer and merchant agree on one existing MPS will be very difficult to achieve at this stage. The obstacle is that both users are not easily convinced to trust and switch over to an unfamiliar MPS, and also other MPS brokers would not give up their current market in support of another system. The consequence of adopting this is another rise of a new wave of competition.

Second, creating a new MPS with the aim at global acceptance appears hardly to be likely in the near future. The credit card system took over thirty years to be globally accepted since it was first introduced in 1965, and only until recent years has it been widely adopted in the cyber world. Besides the technique issue, the main obstacle of this approach is the standardisation process for online micropayment. This process may take quite long period, usually years. It also requires involvement of local financial organisations and tax authorities. Additionally, the legislation and regulations are different from country to country, making it much harder for a new system to be quickly adopted world wide.

In conclusion, this approach might be applied as a long term strategy of the system development, but it has a high chance of failure as the desired solution. This fact was proved by the history of MPSs; many early systems were initially designed to pursue global acceptance and domination, but one failed after another, and none of them gained global success within a short time.
3.3.2 Service-oriented approach

This approach is to retain current MPSs in their market position (i.e. keeping their own users – customers and merchants) and make them incorporable by introducing a universal payment platform (a type of e-service for data exchange), which integrates their existing payment services into a uniform level, but not changing their original functionalities. As a consequence, a serial of cross-system micropayments will be performed. In order to achieve the desired micropayment environment, this approach requires a universal payment protocol to regulate the payment transaction. It also requires an integrated payment portal built on top of the participating MPSs so as to bridge the system differences.

Compared with the previous, this approach is lightweight because the added-in e-service could be portable and platform free. Though it will still experience some of the problems mentioned above, it is more likely to be realized without long term efforts, because it could provide advantages as follows:

1) Both customer and merchant need to trust only one system and rely on its broker (operator) which they might already be used to. So they could still use their existing account or e-wallet to complete a transaction without a need for further software or hardware installation. Consequently, their original trading experiences and habits remain unchanged;

2) System implementation is more feasible and cost efficient against the first approach; this approach only requires a lightweight modification on current MPSs, and investments are only related to the implementation costs of the universal payment portal and protocol;
(3) The technical standardisation process is much easier than that of the market-driven approach, because it only deals with the protocol for the cross-system micropayment; and

(4) Legislation and regulations are already standardized, since these MPSs have been operational in the market, and their brokers have also been regulated by financial organisations and tax authorities.

### 3.3.3 Outcomes of the determined approach

As a result of the approach analysis, the service-oriented approach was determined as the suitable approach to address the problems identified from the current micropayment situation. This approach will also direct the rest of the research.

The initial concept of this approach was from NetPay, a micropayment system proposed by Dai et al. (1999 & 2007). NetPay is an improved version of the PayWord system, an early stage MPS, in which the e-coin (called payword) was specific to individual vendors. NetPay introduced micro-money, called touchstone that solved the problems of vendor-specific payword, and made it acceptable by multiple vendors. Inspired by the innovation of touchstone, this approach might be implemented across multiple MPSs.

The core outcome of this approach is the introduction of the Universal Micropayment System (UMS), which plays a role of a universal micropayment platform. The core component of the UMS involves a payment portal, a part of which is built on the top of the existing MPSs to provide the system incorporation service. Figure 3.5 extended
from Figure 3.3, presenting the approach of using the UMS to incorporate existing MPSs.

![Diagram showing the new approach for incorporating existing MPSs](image)

**Figure 3.5 The new approach for incorporating existing MPSs**

The initial analysis of the direct advantages from this approach has been highlighted in the previous section. Beside those advantages, the main significance of implementing this approach, from the economic point of view, is the rapidly increasing customer and merchant bases. This expectation is realistic, because the proposed UMS will not only maintain the existing end user base of each individual MPS, but also attract more new users (both customer and merchant) to participate. New merchants are attracted because there is a significant expanding customer base that will pay them, and consequently new customers are attracted due to the increasing merchant base that will provide them with products. The concept of this healthy cycle to expand the user base is illustrated in Figure 3.6.
3.4 Approaches to system development

At the solution architecture level, the question about how to develop and measure the desired system has to be addressed. The key to addressing this question, from the system developers view point, is to identify the possible approaches to integrate the defined “e-service” with the existing payment platforms and to measure its desired performance.

3.4.1 Approaches to system cooperation and integration

To realize the universal micropayment service provided by the UMS, the approaches for cooperation and integration of current MPSs needs to be determined.

The system cooperation issues should consider system compatibility and system interconnection. The former is represented as a data-exchange interaction while the latter as a functional correlation. Compatibility could be achieved by a method of
system parameter exchanges. This method involves a conversion of parameters at interaction points. The interconnection could be realized by a method of system mapping. Both methods are applied and presented in the UMS architectural design chapter.

The system integration needs modifications of current systems in terms of hardware and software updating. In order to minimize the modifications and also to retain systems’ local functionalities, this integration could be reached by a new method of adding extra system behaviour on top of existing MPSs. The result of this method is a defined UMS protocol, and a payment portal, which will perform the extra behaviour on MPSs to integrate local payment services into a universal service under that protocol. This method is applied and implemented in the UMS protocol design chapter.

Given the fact that the system data and functions differ from one system to another, identifying the primary system parameters and functionalities is essential. The result of this process is provided from the survey of current MPSs during the literature review.

3.4.2 Approaches to system implementation and measurement

We used case studies to realize the possible implementation and measurement of the proposed UMS. The case study is an ideal technique for this qualitative study; it justifies the approaches determined for this research. The purpose of the case studies is to demonstrate and examine whether the implementations of the proposed UMS and
its payment protocol are theoretically achievable. The case studies show the existing payment functions of selected MPSs, and also present how the payment services of those systems are incorporated and integrated with the UMS protocol. Additionally, the case studies may prove if the UMS and its protocol are implementable.

Case selection is accomplished based on the system’s availability and accessibility. The selection process and statement will be presented in Chapter 7. The implementation and testing of the proposed UMS in terms of its functionality and compatibility will be (and could only be) realized by means of using two case studies. In addition, the performance of the UMS will also be measured by those cases so as to verify whether the functional requirements are satisfied.

Due to the restriction of research time and available resources, the UMS protocol functions could be realized mostly by applying the techniques of system mapping, and measured theoretically by using selected cases. Implementing and testing the UMS architecture and protocol on all examined MPSs appears very hard, both economically impractical and technically impracticable; it is because most of the operational MPS brokers would not disclose their system code in support of the testing performed by another system.

3.5 Design methodology

Based on the sequence of the above mentioned questions addressed at the different levels of EA approach, the process of the UMS architecture design could be formed. To structure this design process, Kleinjohann (2000)’s design methodology was used,
which displays the systematic logic of a design cycle, consisting of five stages shown in Figure 3.7.

![Logical design process circle](image)

**Figure 3.7 Logical design process circle**

The first stage investigates the MPSs, both previously proposed and currently operational. Their system characteristics and payment functions are extensively examined and analysed. This stage was completed and presented in Chapter 2.

The second stage determines the user’s requirements and then formulates the functional requirements. The latter are based on the main functional characteristics of existing MPSs concluded in the first design stage, and the user’s requirements.

The third stage of the design cycle produces the architecture design for the proposed
UMS, where the interaction approaches are identified, discussed and determined. The related interactions and associated interaction parameters are also decided and refined, respectively. Those interactions represent the universal micropayment services, which will be experienced by the end users as an external behaviour of the participating MPSs.

The fourth stage is based on the outcomes of the third stage, because the determined UMS services could bring about the identifications of the UMS protocol elements, as well as related protocol behaviors (or functions). The expected delivery of the designed protocol functions are the harmonization of the payment services of the participating MPSs and generation of the universal micropayment service.

The fifth stage is the implementation of the UMS, in terms of its protocol functions and system incorporation. The measurement of the implementation process is accomplished by the use of case studies. The performance will be analysed, and the fulfillment of previous requirements is measured.
3.6 Chapter summary

The success of a research project largely depends on how well the research is designed and executed. The research questions and suitable approaches guide the whole research design process and methodology. The nature of the research questions of the present study and research outcomes has determined that it is a qualitatively oriented research.

The EA approach – a well-known hierarchical approach to align business and IT, was used throughout the entire research, from problem identification and desired solutions at stakeholder viewpoint, to system design at business owner viewpoint, and to system development at system operator and user viewpoint. In order to realize the proposed system, the EA frameworks were applied throughout the entire design process. To justify the proposed approach and implement the proposed system, this study incorporated case studies to analyse and measure the design approach.

To better understand the different levels of EA approach, we also illustrated four stages for a systematic design cycle. The rest of the chapters will follow this cycle to present the detailed system designs and implementations.

In the next chapter the requirements for the UMS will be identified and explained. User requirements and requirements of system owner/operator form the basis of the system functional requirements, which will construct the baseline for the system architecture design.
Chapter 4  Requirements for
Universal Micropayment System (UMS)

As introduced in the previous chapter, the proposed Universal Micropayment System (UMS) is desire to achieve seamless integration and fully cooperation with current payment functions (services) of existing Micropayment Systems (MPSs). Before the UMS architecture is proposed, a better understanding of requirements for the UMS is essential. This chapter is to identify those requirements and determine the important ones for further system design.

According to the conclusion of the literature review chapter, the requirements for the UMS could be generally grouped into three main categories: (1) user requirements, (2) Micropayment System Broker (MSB) or operator requirements, and (3) functional requirements. They are structured in a top down manner shown in Figure 4.1.

![Figure 4.1 The requirements for UMS](image-url)
The requirements of the first category are inspired by Walczuch and Duppen (2002) and derived from the conclusions of Chapter 2. A second requirement category is driven from the expectations of the current business where the proposed system is expected to function, whilst the first two requirement categories determine the third in a greater extent, as they largely influence the functionality design for the UMS. Other requirements related to the operational issues of the UMS, such as the legal and regulatory requirements are outside the scope of this research.

This chapter is structured as follows. The user requirements and MSB requirements are discussed in Section 4.1 and Section 4.2, respectively. Based on those requirements and the main functional characteristics of the existing MPSs (presented in Section 2.9), the functional requirements are determined and defined in Section 4.3. A chapter summary is given at the end.

4.1 User requirements for UMS

The customer and merchant (as defined in Section 2.1.4) form the end users group. The fulfilment of their requirements determines largely the success of the UMS’s acceptability and penetrability. The requirements from customer’s and merchant’s points of view may differ in some aspects, but from the e-commerce viewpoint, they are quite similar in general.

The user requirements focus on the needs and expectations of customers and merchants with respect to the functions and characteristics of the UMS. The weight of
each requirement may vary from user to user. The following main factors are identified:

- a uniform payment platform or portal;
- a user-friendly payment system;
- a secure system;
- a trusty system; and
- a system with anonymity and privacy (especially from the customer point of view).

The core factor associated with these requirements is the implementation and operation costs, which charge customers and merchants for using the payment system. The degree to which the various requirements are implemented influences the cost for end users. For example, the more secure and trusty a system is experienced by the users, the more expensive it will be for them. However, customers are only prepared to pay for the value of the product while using payment systems. Odlyzko’s study (2003) indicated that a flat rate could maximize the micropayment usage; it was found that customers prefer fixed costs for predictability reasons, when they are conducting small payments on the Internet, and that’s why the subscription pricing model is still dominating the market of online paid content. It is also argued by Shirky (2000) that a ‘mental cost’ associated with the processing of an online micropayment would largely reduce the customers’ willingness of making tiny purchases.

Consequently, current experience shows that most MPSs are free of charge at the customer side. Therefore the burden of transaction costs is applied to the merchant
side. Most current MPSs require merchants to pay for using these systems in the forms periodical subscribing fees (e.g. PepperCoin and ClickandPay systems), and transaction fees (e.g. BitPass and wallie systems), and an initial sign up fee (most of MPSs). Naturally, merchants require these costs to be as low as possible.

4.1.1 A uniform payment platform

The primary requirement from the users’ points of view was indicated from the diagram shown in Figure 3.4. They prefer to use a uniform payment system or a platform that is able to process all micropayments. Both customers and merchants expect to reach the majority coverage of each other, without the need for signing up with multiple systems.

This requirement indicates that the UMS should have the ability to incorporate and integrate the existing MPSs, as many as possible, such that customers and merchants can maximally reach each other. For instance, customer A wants to use one account of payment system X that he/she already has registered with to pay all merchants, not only who uses the same system X, but also who uses different systems Y, and Z. In contrast, a merchant requires such a platform that could deal with the reception of potential multi-payments processed by various systems. This requirement also implies that a chain of micropayments have to be performed in a very efficient, straightforward and harmonious manner.

The requirement for a uniform payment platform also means that the cross-system payments in different currencies should be supported. As summarised in Section 2.9,
most micropayment systems are developed and operated locally with a single
currency. However, the need for global payments emerges because the Internet
provides an international network without country boundaries. This requirement has
already been considered and highlighted by many system developers, so that it also
becomes one of the major concerns in the study.

In addition, it is essential that the UMS should enable the support of efficient transfer
of very tiny payment (e.g. one cent) for all targeted online content types that are listed
in Section 1.1.2. As described in Chapter 1, one of the motivations for this research is
based on an irrefragable deduction that a significant market for low-value
(e.g. ranging between one cent and two dollars) intangible online content will soon
emerge. To support the selling and buying such type of content, end users (both
merchants and customers) not only require a payment system that can efficiently
transfer micro-money, but can also appreciate such a system that can reach maximum
coverage of the existing and potential user bases.

4.1.2 A user-friendly payment system

Both customers and merchants will only accept a user-friendly payment system.
Kniberg’s study (2002) on interviews with merchants and payment system brokers
revealed that end users consider the ease of use the most important requirement for
payment systems.

The ease of use is the feature that expresses how easy and comfortable the learning
and using of a payment system are, even for a user who had not used the system
before. For the case of the UMS, the ease of use means not only straightforward payment transactions, but also simple and transparent interaction with the participating systems by keeping minimal changes to the user’s habits and experiences. Otherwise, they might be unwilling to switch over to a new system and keep using it for a long run. For instance, customers that have a user name and password combination or an account number to log into the existing system they use, should be able to continuously use the same authentication methods for the UMS payment service.

E-payment systems experienced the failure of SET – a very secure payment standard introduced in Section 2.4.1. It failed because its usage was too complicated for its end users, such that it finally resulted in user’s resistance. Therefore, the ease of use will significantly influence the success of the UMS.

4.1.3 A secure payment system

Though many researchers have proved that the success of the system is not dependant on how secure the system is, but the user’s acceptance, security is always considered one of the top issues. Since the Internet micropayment appeared a decade ago and has become more popular in recent years, the attempts of misuse of those systems in order to commit fraud on the Internet are becoming very common (Abrazhevich 2001b). That’s why security is still a factor that influences the widespread acceptance of a payment system by users.
Security is a subjective concept to certain extent, and has different perceptions from end users (Centeno 2002b). For instance, customers expect that their e-money will not be misused by a third party, and want to ensure that merchants could not abuse the system, so that they will receive the paid content after the payment is completed.

On the contrary, merchants expect that e-money received from customers could be redeemed and expect the system to provide the non-repudiation with respect to the customers. Accordingly, the security of the proposed system is required to be strong enough to detect attacks or to detect the third parties responsible for attacks.

As a result, this research adopts the main security concerns summarised from O’Mahony (2001) and Stallings (1999), they are confidentiality, authentication and authorisation, data integrity and non-repudiation. Each has distinct purpose described as follows:

- Confidentiality ensures that unauthorised parties cannot access the sensitive payment information that might be used later for fraudulent purposes. Customers want to be sure that their transactions remain secret from others.
- Authentication and authorisation are two functions of the MPSs that establish the identity of the customer and determine their right to make payments. These functions will guarantee customers that no other user could use their money.
- Data integrity ensures that payment information is not altered after a payment is initiated, and this information can only be accessed by authorised parties. Measures that provide integrity allow the detection of any unauthorised attempt to modify payment information.
• Non-repudiation is a property of a payment system that does not allow users to deny their actions that resulted in money transfers. Merchants want that customers cannot deny receiving the products.

Theoretically, the proposed UMS should be secure enough to detect and prevent any attack. However, systems that deal with micropayments only need limited and lightweight security measures. This is because a balance must be kept between the costs of implementing the security measures and the protected value of a payment. In general, a payment is considered to be safe if the costs of breaking its security are higher than the value of that payment. Therefore, it is not suggested a complete redesign of additional security measures for the proposed UMS is required.

4.1.4 A trustworthy system

Users require all parties involved in the operation of online transaction should be trustworthy whenever money is concerned. Several studies (e.g. Egger 2001, Kniberg 2002 and Centeno 2002b) concluded that trust is a precondition for the success of e-commerce and a vital requirement for payment systems. Kniberg (2002) also suggested that trust is more important to users than security, and users are more likely to use a less secure payment system operated by a trusty MSB (or system operator), rather than using a very secure payment system from a less trusty MSB. Subsequently, trust in the MSB becomes a vital requirement for end users.

Building the trust in the MSB has a different meaning for merchants and for customers. Merchants require that MSBs could transfer the collected amount of
money periodically as defined in their contracts. Customers expect that they will receive the paid content from merchants. In addition, customers also want MSBs to register trusty merchants, and on the contrary, merchants also expect only honest customers to register with the system.

Users, both customers and merchants expect to be able to use the payment systems of their choices, which are already managed and operated by one of MSBs or a financial institute they trust so far (it is assumed that if the users trust a MSB or a financial institute, they will trust the payment system provided by the MSB). It simply indicates that the trust concerns could be shifted from user side to MSB side. Consequently, if the trust has already been built between current MSBs and the operator of the UMS, then the trust in the UMS from the end user side would become a non-issue.

The previous finding shows that building trust between users and systems is a very difficult task and can not be solved through purely technical means. Therefore, except for the system security consideration, to build trust with the UMS, there is also a need to provide a level of confidence and usability substantially similar to what has been standardized in the existing MPSs.

4.1.5 A system with anonymity and privacy

Customers require remaining anonymous to both merchants and MSBs. Anonymity has two aspects from the customers’ point of view. First, customer information is anonymous to merchants. For instance, customers want to remain unidentified when they are downloading online adult content (which is currently one of the most
profitable online content types); second, many customers prefer micropayments not to be recorded. It simply means that they do not want to see each payment on their periodical bill sent from the MSBs (no matter whether pre-paid or post-paid bills). However, customers’ anonymity to merchants could be achievable, while their anonymity to MSBs may be unavoidable or less important, because their information has already been disclosed to the MSBs of their choice during the signup stage.

Customers require that their privacy is protected by both merchants and MSBs. It becomes one of the vital factors that determine the possible success of a payment system among customers (Centeno 2002b). Customers own their personal information and have a right to privacy, which should not be violated. Studies of Forrester Research (Kelley et al. 2001) state that 60% of online customers have fears about the misuse of their personal information (e.g. sending them unsolicited promotional materials). However, merchants and MSBs demand the basic customer information to be able to perform their business activity and statistics, such as requiring credit card information for deposit guarantee, sending confirmation messages to customers’ emails and analysing the customers’ preferences, etc. Therefore, both merchants and MSBs need to have ability to protect customers’ information from external attacks.
4.2 MSB requirements for UMS

The MSB requirements for the UMS could be influenced to some extent, but probably not fully determined, by user requirements. The requirements from the MSBs’ point of view may include the requirements of the existing MSBs and future brokers. Those requirements include the support for small amounts of money, wide penetration and extensive acceptability of the system, as well as future increase of transaction volumes, etc.

The proposed UMS is aiming to interconnect existing MPSs and cooperate effectively at a universal level. For such system’s interconnection and integration, the following requirements from the MSB side are defined as follows:

- minimal modifications applied to their systems;
- original system functionalities remain in place;
- compatibility, reliability and scalability; and
- and trust.

4.2.1 Minimal modifications applied to existing systems

MSBs require that the integration of payment services involves minimal modification to their systems’ infrastructures. They require that their systems are still operational as they used to be. It indicates that the system integration needs to be efficiently feasible, in term of cost, time and on-going human-involved efforts. Besides that, they also require benefits from the potential profits or possible increased user bases, all of
which might make those small modifications economically manageable and worthwhile for the system development.

4.2.2 Original system functionalities remain in place

MSBs require that their systems be incorporated with a new system platform with minimum changes of their original system functionalities, especially those changes on the original payment interface, and transaction performance should be kept at a minimum. As one of user requirements described in Section 4.1.2, MSBs are unlikely to undertake the risk of changing user’s existing habits and experiences. In addition, some functions have significant local system features that need to remain functional with the integration of a cross-system payment service.

4.2.3 Compatibility, reliability and scalability

MSBs require that the performance of the UMS is compatible to that of their existing systems, in terms of functionality, security mechanism, billing methods. It means the integration of the UMS should minimize the changes to their current business model and business operation.

MSBs require high reliability of the universal payment portal, because they have contracts with end users who want to use their systems and services continuously. Their expectations require that the cross-system payment service has the ability to process huge amount of payments simultaneously without creating a bottleneck in other systems.
Meanwhile, MSBs also require scalability of the UMS. This requirement highlights the fact that the components of the UMS should have scale capacity, for when the user base expands and the volume of universal (cross-system) payments increases. MSBs require this because they have existing and potential users, whose needs are required to be fulfilled, even if the services they provided are integrated into the UMS payment services.

4.2.4 Trust

Foley and Quillinan’s study (2002) related to the trust issue within the e-commerce relationship has revealed that customer is the least honest party in that business relationship, then merchant is the second, and the payment system operator or broker is the most trustworthy party compared with the customer and the merchant. Trust in the UMS from the MSBs’ viewpoint has a different meaning from the one from users’ viewpoints (which has been described in Sections 4.1.4).

It is assumed that those MSBs who agree on the integration of the UMS would trust the operator of the UMS. However, they may not trust other MSBs that cooperate with the UMS, such that the trust issue among the MSBs needs to be addressed. One achievable possibility is the use of business agreements, which are required to assure interoperability among various system brokers. These agreements go beyond our research scope and also are less important for the design cycle presented in this thesis.
4.3 Functional requirements for UMS

The user and MSB requirements determine the functional requirements for the UMS, whilst the functional characteristics of the existing MPSs determine how those functions could be delivered through the universal payment service. The UMS functional requirements define the main characteristics and functionalities of the UMS as expected by both users and MSBs.

The primary functions and payment interactions of the examined MPSs were summarised in Section 2.9. According to the findings and the requirements from both users and MSBs, the main functional requirements for the UMS could be analysed from the following aspects: (1) access authentication; (2) payment initiation; (3) payment confirmation and (4) the payment method.

4.3.1 Access authentication

As practiced by current operational MPSs, the access authentication is an essential function for the UMS, because it could extend the current customer and merchant sources. The UMS needs to inherit the authentication methods used by the existing systems. For instance, as one of the UMS users, the customer should be able to use his original login method to access his account in the MPS he has used so far. This is also one of the expectations from the end users. Before a cross-system payment is to be performed, the customer is also supposed to acknowledge the use of the UMS service. This function requires a uniform payment interface or a system-independent interface that users trust and are fully confident with.
4.3.2 Payment initiation

As summarised in Section 2.9, the majority of existing MPSs require customers to initiate a payment. Customers are always involved in payment initiations as they have to acknowledge that their accounts are going to be charged. The joint initiation shown in Table 2.10 is an alternative method of payment initiation used in real practice by few numbers of MPSs; it may be impractical for the UMS. Additionally, customers require and appreciate a user-friendly payment integration, in which the payment functionality should be similar to that of their systems currently in use. As a result, their payment methods are kept consistent with their previous habits (which meets one of the user requirement described in Section 4.1.2).

4.3.3 Payment confirmation

As shown in Table 2.10, the majority of existing MPSs provide confirmation to merchants, though only a few apply joint confirmation method, such as PepperCoin system. Merchants need confirmation from the UMS immediately after a universal payment is completed, so that they can deliver the paid content to the customer. Customers might consider the confirmation as a type of payment proof, but this research considers that the delivered content could sufficiently prove their payments. This simplification is justified because only a small amount of money is involved in each payment.

Furthermore, merchants also require and appreciate a user-friendly payment integration, in which the way they collect payments resembles the way they are
familiar with. Consequently, their experiences and habits do not have to change. In the case of systems that provide confirmation only to customers, then the customers need to send the confirmation to the merchants, so that the paid content could be delivered. Because this situation is less common in current practice, this type of confirmation method will not be considered in the design of the UMS.

4.3.4 Payment method

As found in literature, most of the existing MPSs could be grouped into the pre-paid or post-paid category. To cooperate with the majority of systems, the UMS needs to function as pre-paid and post-paid simultaneously, depending on the MPSs interacted with it; otherwise the UMS could not reach the maximum coverage of MPSs for possible system integration. Other requirement issues regarding the supported value ranges and currencies of micropayments have been discussed in the user requirement in Section 4.1.1.

4.4 Chapter summary

This chapter formulated the requirements for the UMS. Those requirements are driven from user, MSB, and business function aspects.

Regarding user requirements, both from the customer and merchant viewpoints, are quite similar. Users require a trusty universal payment platform, which could extend their original favorite interface and also provide secured cross-system payment services with anonymity and privacy.
Requirements from the MSBs, include the need for minimum modifications of their original system infrastructures and the continuance of current local functionalities as well as needs for compatibility, reliability and scalability of the UMS.

The functional requirements are determined by user and MSB requirements, and are largely influenced by current MPSs’ functionalities, in terms of access authentication, payment initiation, confirmation, and the payment methods. A summary of the functional requirements for the UMS is drawn in Table 4.1.

<table>
<thead>
<tr>
<th>Functional requirement list for UMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General requirements</strong></td>
</tr>
<tr>
<td>Function scalability</td>
</tr>
<tr>
<td>Function reliability</td>
</tr>
<tr>
<td>Function implementability</td>
</tr>
<tr>
<td><strong>Specific requirements</strong></td>
</tr>
<tr>
<td>User authentication (same methods)</td>
</tr>
<tr>
<td>Trusty payment interface</td>
</tr>
<tr>
<td>Customers initiate payment</td>
</tr>
<tr>
<td>Jointly payment initiation method</td>
</tr>
<tr>
<td>Merchant confirmed payment</td>
</tr>
<tr>
<td>Customer confirmed payment</td>
</tr>
<tr>
<td>Multiple currency supported</td>
</tr>
<tr>
<td>Minimum value ranges supported</td>
</tr>
<tr>
<td>Maximum value ranges supported</td>
</tr>
<tr>
<td>Pre-paid authorisation method</td>
</tr>
<tr>
<td>Post-paid authorisation method</td>
</tr>
</tbody>
</table>

All the above requirements provide the baseline for the UMS architecture design, which will be presented in the next Chapter.
Chapter 5 UMS architecture

This chapter presents the architecture design for the Universal Micropayment System (UMS). Section 5.1 identifies the core interactions of a UMS transaction and then discusses possible and suitable approaches to realize those interactions. Driven by the functional requirements described in the previous chapter, the UMS architecture is proposed and presented in Section 5.2. To better understand and explain the architecture, two interior layers of the UMS architecture, Universal Access Layer (UAL) and Universal Transaction Layer (UTL), are identified, with associated service interaction parameters defined and presented in Section 5.3 and 5.4, respectively. Section 5.5 summarises the chapter.

5.1 Universal micropayment interactions and approaches

The behaviours of the UMS consist of the interactions between the UMS and its users, and interactions between the UMS and multiple Micropayment Systems (MPSs). In the following sub-sections, the possible approaches to perform those interactions are discussed and the suitable solutions for both types of interactions are identified, which are presented in Section 5.1.1 and 5.1.2, respectively.
5.1.1 Interaction between UMS and its users

Within the universal micropayment environment, system users are also divided into two groups: customer and merchant. The customer who uses the UMS to make universal payments to merchant or content supplier is defined as a *Buyer*, and the merchant who uses the UMS to receive payments from customers is defined as a *Vendor*. To clearly specify the users of the UMS payment service, we consistently apply the defined terms to denote the UMS users for the rest of this thesis.

The UMS payment service characterizes the external behaviour of the existing MPSs, which will be experienced by its users – *Buyer* and *Vendor*. Two approaches are distinguished for the UMS users to interact with the UMS.

**Approach I**

One approach is to use the existing interfaces (or payment platforms) of current MPSs. Using this approach can not avoid the problem that the MPSs used by buyers may generate payment initiation information in many different formats. The information with various formats is then sent to vendors that probably use other MPSs. However, these vendors can only accept payment information that is specific for their MPSs. As a result, this information may have less chance of being compatible or acceptable by the vendors’ systems due to the various types of data.

To solve this problem and enable payment initiation with all types of data, the UMS needs a rather complex functionality to handle the definition of data mappings
between any two types of payment information. Due to the large number of existing MPSs that all require payment initiation information with their own formatting style, this approach seems to be much more difficult to realize.

**Approach II**

The other alternative is to standardize the interface or portal, between service users and the UMS, and to make it system-independent. Such portal should be extended and accessed from the original payment interface that users are familiar with. In this way, buyers and vendors will always interact with the UMS in the same way, and will provide the payment information in the same format, regardless of the specific MPSs used by each party.

This approach seems more realistic and has a higher chance of success than the first one. Hence this approach will be applied in the design of the payment service for UMS.

**5.1.2 Interaction between UMS and multiple MPSs**

Within the context of computer networks, the incorporation and integration of existing systems to create a larger and more complicated system is a substantial challenge, which has been mentioned in literature before, especially the embedding of an assorted or incompatible sub-system into the main one as an undependable sub-network (Barry 2003).
In the case of UMS, the cross-system payment service is actually interconnecting the multiple independent MPSs (sub-systems) into one complex system network. Within this universal micropayment network, two groups of accounts with different roles are identified: debtor account and creditor account. The entity that uses a debtor account to make a universal transaction is defined as *Payer*, and the entity that receives and responds to the transaction is defined as *Payee*. To clearly specify the accounts for the UMS transaction service, those defined terms are consistently applied to denote the UMS account roles for the further system design.

According to the method developed by Parhonyi et al. (2005) for system connection, two possible approaches were inspired and identified to address the system’s interaction issues in this research.

**Approach III**

One approach is to directly connect the two participating MPSs via the UMS, as the-system-in-middle, which is playing the role of a universal payer and payee, and performing transactions under the defined payment protocol. Because the design of the UMS components is specific to the pair of MPSs, this interaction approach requires the system mapping definition exclusively designed for this pair of systems. The advantage of this approach is to allow the UMS to integrate seamlessly to the existing MPSs without rebuilding new functionality or embedding certain application for cross-system interactions.

However, this interaction approach actually indicates that two mapping procedures need to be defined for each pair of inter-related MPSs. Since the UMS should be able
to function as payer and payee for them, it needs to generate and receive transaction data for and from all MPS involved. Consequently, the system interactions need to be cross-directional. For instance, when \( n \) systems need to be involved in a universal transaction, \( n^* (n-1) \) mapping procedures need to be pre-defined by the UMS. Additionally, the UMS requires a complex functionality and huge data storage capacity to handle those operations. This interaction approach is only able to be achieved on an individual basis, when the number of participating systems is comparably small. Given the fact that the number of current MPSs is rising and the future e-payment market is expanding, this approach will consequently be much more difficult to realize.

**Approach IV**

The other alternative approach to realizing a universal micropayment is to standardize the micropayment transactions between UMS and multiple MPSs. It simply means that all payment interactions among systems could be carried out under a same mapping procedure, and all cross-system transactions could be achieved under a pre-defined payment protocol, regardless of system-specific payment data structured in different formats. An advantage of this approach over the first one is that only one set of mapping procedures needs to be defined for all participating MPSs. Another advantage is that if a new MPS is signing up to the UMS environment, then only one mapping procedure (defining a new mapping rule between the new system and UMS) needs to be fulfilled, rather than \( n \) times mapping procedures performed in the previous approach. By this method, it dramatically reduces the number of system mappings compared to the previous one.
However, the main point of this approach is the requirement of building a new functionality for cross-system interactions at the side of the participating MPSs. To minimize the local modifications occurring at the side of system operator (or MSB) (as defined in the requirements in Section 4.2.1), the method of embedding a UMS component (a software application) on top of existing MPSs is introduced. The only assumption is that the agreements are made between MSBs and operator of the UMS before the service integration for a universal transaction. As mentioned before, such agreement is excluded from the research scope. The universal micropayment service will not only inherit and resemble the existing payment services provided by MPSs, but also remain the local functionalities of those systems unchanged (as defined in the requirements in Section 4.2.2). Therefore, the concerns for the system modification could be considered negligible. In conclusion, the system interactions seem more suitable and easier to be implemented than that of the first approach. This alternative approach seems more realistic and has a higher chance of success than the first one. Hence, this approach for system interaction will be applied in the design of the UMS payment services.

Figure 5.1 depicts the interactions between the UMS and multiple MPSs. The underlined buyer’s system (MPS-buyer) and vendor’s system (MPS-vendor) interact via a universal payment portal that bridges the payer and payer entities specific to the MPS-buyer and MPS-vendor, respectively.
5.2 UMS architecture

With regard to the functional requirements given in Section 4.4, Table 5.1 presents a traceability matrix to those requirements for the possible approaches identified in the previous section.

Table 5.1 Traceability matrix to functional requirements

<table>
<thead>
<tr>
<th>Functional requirements</th>
<th>Approach I</th>
<th>Approach II</th>
<th>Approach III</th>
<th>Approach IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function scalability (General)</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>Function reliability (General)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Function implementability (General)</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>User authentication (same methods)</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Trusty payment interface</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Customers initiate payment</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Jointly payment initiation method</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Merchant confirmed payment</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>Customer confirmed payment</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Multiple currency</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>Minimum value ranges supported</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Maximum value ranges supported</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Pre-paid authorisation method</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>Post-paid authorisation method</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
</tr>
</tbody>
</table>
It is clear that the approaches III and IV could fulfil the general functional requirements in terms of scalability, reliability and implementability. For specific requirements, approach II is more suitable than approach I in handling interactions between UMS and its users, approach IV is also standout over approach III in dealing with interactions between UMS and multiple MPSs.

According to the interactions identified and suitable interaction approaches determined above, the architecture of the proposed UMS is diagrammed in Figure 5.2, in which buyer and vendor are the primary users of the UMS. Between the users and the participating MPSs is laid the UMS protocol, which consists of three main components: Universal Payer \((Payer_{uni})\), Universal Payee \((Payee_{uni})\) and the Universal Micropayment Portal (UMP). The \(Payer_{uni}\) and \(Payee_{uni}\) components are actually the UMS applications that need to be integrated into the local MPSs, whilst the UMP is connecting and cooperating with them to accomplish a cross-system universal micropayment. The protocol layer is built upon the multiple MPSs, which represent the systems of choice from both MPS-buyer and MPS-vendor. The \(Payer_{uni}\) and \(Payee_{uni}\) components also interact with the buyer’s account \((Payer)\) and vendor’s account \((Payee)\), respectively, to support system mapping and parameter exchange for universal payment service.

Within the context of the UMS architecture design, two different layers are distinguished: Universal Access Layer (UAL) and Universal Transaction Layer (UTL), which will be further explained in the following sections (5.3 and 5.4).
The process flow starts from the payment initiation of a buyer, ends at the transaction confirmation of a vendor. The core UMS interactions in this flow are data exchange and system mapping, such as data exchange between buyer and MPS-buyer, between vendor and MPS-vendor, and system mapping between MPS-buyer and UMS, between UMS and MPS-vendor.

The data exchanges occurring at two layers result in different means of parameter definitions and different parameter values. The system mapping here is, in particular, referring to the mapping of system parameters, which are structured in various formats from system to system. The universal payment protocol of the UMS refers to the rules of how the exchanged parameters are defined and how the cross-system interactions are performed. The UMS protocol layer is built between UAL and UTL, handling the exchanges of mapped and UMS-compatible parameter values. Buyer and vendor, payers and payees of the systems are connected at different layers through the Data Exchange Point (DEP), where the Data Exchange Interaction (DEI) occurs.
5.3 UMS architecture - Universal Access Layer

The interactions (related to universal payment) between the UMS and its users (described in 5.1.1) occur at this layer. Those interactions for a universal payment are regulated by deciding which user initiates cross-system payments and which receives payment confirmations. According to the functional requirements, universal micropayments are initiated by buyers and are acknowledged or confirmed to vendors. The universal payment service involves the defined DEPs and DEIs, correlated parameters, as well as associated constraints that control their interactions.

As a consequence, the entity structure of UAL is shown in Figure 5.3. Two DEIs, universal payment initiation and confirmation, are identified at this layer. They occur at $DEP_a$ (between Buyer and the UMS) and $DEP_b$ (between the UMS and Vendor), respectively. The payment initiation at UAL is presented as a Universal Payment Request ($Uni-Pay-Req$ in short) message and the confirmation presented as a Universal Payment Response ($Uni-Pay-Resp$ in short) message. This terminology will be applied for the rest of research.

![Figure 5.3 Universal Access Layer](image-url)
5.3.1 Payment initiation DEI

The buyer executes the *Uni-Pay-Req* interaction first to initiate a micropayment. The *Uni-Pay-Req* indicates that the buyer needs to pay a required sum of money to a specified vendor. Prior to the execution of the payment initiation, the MPS-buyer needs to identify and authenticate the buyer. Once the *Uni-Pay-Req* interaction has been performed, it is not possible for the buyer to reverse, cancel or deny the payment. That means if this interaction is performed reliably, then non-repudiation is avoided. Because only the result of an initiation is relevant, we abstract it from other interactions (such as log in to the system, credit checking, etc.), which are part of the initiations from existing MPSs. It needs to be noted here that the incorporated MPSs may process a chain of data exchanges for a single UMS payment, therefore, conducting a *Uni-Pay-Req* interaction might require a slightly longer transaction time than processing of an ordinary micropayment initiation by individual MPS.

The parameters of the *Uni-Pay-Req* interaction provide the essential information to the UMS in order to perform a cross-system payment. In practice, different MPSs use different sets of parameters in payment initiations. In general, four distinct parameters are primarily identified in all payment initiations of the studied MPSs, they are: buyer’s identifier (*Buyer*$_{id}$), vendor’s identifier (*Vendor*$_{id}$), product or ordering identifier (*product*$_{id}$) and transferred value of product (*Value*). Other local parameters, such as vendor’s name (*Vendor*$_{name}$) and URL (*Vendor*$_{url}$), product description (*Desc*) and it’s encrypted URL (*URL*$_{key}$), session expired time (*Session*), success (*URL*$_{succ}$) and failure URLs (*URL*$_{fail}$), etc., are all system-specific parameters,
which are particularly used for the purpose of tracing purchases at local MPSs, so they would be ignored at UAL. Table 5.1 summarises the possible system parameters and also presents the example values of primary parameters for Uni-Pay-Req interaction.

Table 5.2 Uni-Pay-Req parameters at UAL

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Sample value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary parameters</td>
<td></td>
</tr>
<tr>
<td>Buyer(_{ID})</td>
<td>JohnWell: 1976380</td>
</tr>
<tr>
<td>Vendor(_{ID})</td>
<td>N30-3</td>
</tr>
<tr>
<td>product(_{ID})</td>
<td>9980-51500mzrr</td>
</tr>
<tr>
<td>Value</td>
<td>55c</td>
</tr>
<tr>
<td>Other local Parameters</td>
<td></td>
</tr>
<tr>
<td>Vendor(<em>{name}), Vendor(</em>{URL}), Desc, Session, URL(<em>{key}), URL(</em>{succ}), URL(_{fail}), Time, etc.</td>
<td></td>
</tr>
</tbody>
</table>

The information of buyers and vendors are stored within the original MPSs of their choice, instead of being stored within the UMS. The parameters – Buyer\(_{ID}\) and Vendor\(_{ID}\) are the identifiers that are used for authentication of account access because they are directly associated with their accounts. In Uni-Pay-Req, the buyer’s identifier usually consists of a username and password set, while the vendor’s identifier normally contains a combination of a merchant identifier and a serial number of the system that merchant is currently contracted to. These identifiers should not change with the introduction of the UMS, such that both buyers and vendors do not need to change their original payment habits (as requested in Section 4.1.2).

However, these identifiers need to be kept unique within the UMS. Based on existing identifiers, the following simple method (denoted 1-6) for creating uniqueness of buyer and vendor’s IDs within the UMS is proposed and described in Table 5.2.
By applying this method, buyers and vendors keep using their identifiers, which are extended with a unique identifier of their own systems. This method guarantees the uniqueness of the buyer and vendor’s IDs, since there are no two existing MPSs with the same identifiers or names.

### Table 5.3 Denotation of the uniqueness of buyer and vendor’s ID within UMS

<table>
<thead>
<tr>
<th>Description</th>
<th>Denotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1). $n$ denotes the number of MPSs, and $\Omega$ denotes the set of MPSs</td>
<td>$\Omega = { MPS_i</td>
</tr>
<tr>
<td>(2). the set of all buyer IDs of a system $MPS_i$</td>
<td>$B_i = { B_j</td>
</tr>
<tr>
<td>(3). the set of all vendor IDs of a system $MPS_i$</td>
<td>$V_i = { V_j</td>
</tr>
<tr>
<td>(4). suppose each $MPS_i$ has a unique ID, and $\Delta$ denotes the set of these IDs</td>
<td>$\Delta = { ID \mid ID_i$ is the unique ID of $MPS_i, MPS_i \in \Omega, i = 1, n }$</td>
</tr>
<tr>
<td>(5). the set of buyer ID for the UMS is created by combining the unique $MPS_i$ ID with the buyer IDs from $B_i$</td>
<td>$Buyer_{ID} = { &lt; ID_i, B &gt;</td>
</tr>
<tr>
<td>(6). the set of vendor ID for the UMS is created by combining the unique $MPS_i$ ID with the vendor IDs from $V_i$</td>
<td>$Vendor_{ID} = { &lt; ID_i, V &gt;</td>
</tr>
</tbody>
</table>

### 5.3.2 Payment confirmation DEI

The MPS-vendor is responsible for performing the Uni-Pay-Resp interaction after a Uni-Pay-Req occurs and its’ acceptance is confirmed. With this interaction at UAL, the vendor will be informed by its’ MPS of the completion of the universal payment.

The vendor is not allowed to refuse or deny the confirmation, so that non-repudiation is prevented. The MPS uses the $Vendor_{ID}$ specified in the Uni-Pay-Req message, and
stores the confirmation parameters for payment tracing purposes when a conflict occurs. The vendor also uses those data for further business purposes, such as financial accounting, product usage analysis, handling customer complaints, and so on.

The message carried with the Uni-Pay-Resp interaction contains parameters of the payment confirmation, which provide the necessary information to the vendor, who is then responsible to deliver product to the buyer. Those parameters also indicate to the vendor which product has been paid for and needs to be delivered. Though, in real practice, confirmation parameters are diversely configured from system to system. According the literature, three primary parameters are distinguished in the payment confirmations of the studied MPSs. They are product identifier (product_id), transferred value (Trans_Val) and payment identifier (payment_id). The Trans_Val might be useful, because it helps reduce the cheating from buyers if this parameter is indicated to vendors. Other local parameters, such as Vendor_id, version of MPS used (MPS_Vers), type of payment (PType), date and time of the payment (Date, Time), are some of system-specific parameters, so they would not be considered at UAL. Table 5.3 summarizes the possible system parameters and also presents the example values of primary parameters for Uni-Pay-Resp interaction.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Sample value</th>
</tr>
</thead>
<tbody>
<tr>
<td>product_id</td>
<td>9980-51500mzrr</td>
</tr>
<tr>
<td>Trans_Val</td>
<td>55c</td>
</tr>
<tr>
<td>payment_id</td>
<td>C9980-2007323-1655</td>
</tr>
</tbody>
</table>

Table 5.4 Uni-Pay-Resp parameters at UAL
5.4 UMS architecture - Universal Transaction Layer

The interactions (related to universal transaction) between the UMS and multiple MPSs (described in 5.1.2) occur at this layer. These interactions for a universal transaction are regulated by deciding which account the cross-system transaction conducted from and to. In short, it decides *Payer* and *Payee*, referring to different accounts. According to the functional requirements, a cross-system transaction starts from payer and completes at payee. The universal transaction service also involves the defined DEPs and DEIs, correlated parameters, as well as associated constraints that control these interactions. The entity structure of UTL is shown in Figure 5.4.

![Figure 5.4 Universal Transaction Layer](image)

Four DEIs, universal transaction initiations and confirmations, are identified at this layer. They occur at $DEP_C$ and $DEP_E$ (between *Payers* and the UMS), and at $DEP_D$ and $DEP_F$ (between the UMS and *Payees*), respectively. The transaction initiation at UTL is presented as a Universal Transaction Request (*Uni-Trans-Req* in short) message and the transaction confirmation presented as a Universal Transaction...
Response (Uni-Trans-Resp in short) message. This terminology will be applied for the rest of research.

5.4.1 Universal transaction initiation DEI

The payer of MPS-buyer performs the Uni-Trans-Req interaction first to launch a universal transaction. Prior to the acceptance of the transaction initiation by the UMS, a few procedures need to be completed. First, the MPS needs to identify the payer and authorize the transaction, if the paid amount is not over the credit limit, no matter whether the type of the account is pre-paid or post-paid. Second, the UMS needs to confirm the existence of the destination payee of MPS-vendor. Third, the UMS also needs to check whether both the MPSs are supporting the amount (or equivalent amount when different currencies are used) that will be transferred. Once the transaction initiation is accepted by the UMS, it determines that the payer has already been debited a certain sum of money, which has been credited to the payee through the UMS. This process cannot be reversed or cancelled. In other words, an accepted initiation means that the transaction is successfully completed. It is noted that the Payeruni and Payeeuni components of the UMS are working behind of each universal payment transaction to bridge the protocol gap and perform the parameter matching for the incorporated MPSs. This will be further discussed in details in the next Chapter.

Parameters of the Uni-Trans-Req message provide the essential information to the UMS in order to perform the universal transaction. As determined in the Uni-Pay-Req message (see Section 5.3.1), four primary parameters (BuyerId, VendorId, productId,
and value) are defined in each payment initiation DEI. To formulate the universal transaction protocol, new and primary parameters are defined and correlated to the parameters of Uni-Pay-Req. They are payer identifier (Payer\textsubscript{id}), payee identifier (Payee\textsubscript{id}), content identifier (Content\textsubscript{id}) and universal micromoney (uniMicro), in terms of amount and currency for content value.

As mentioned before, Payer\textsubscript{id} and Payee\textsubscript{id} refer to the identities of debtor account and creditor account, respectively. At UTL, there is no existing product information exchanged between the payer of MPS-buyer and the UMS, nor between the UMS and the payee of MPS-vendor. For the purpose of tracing each universal transaction, a new parameter - content identifier (Content\textsubscript{id}) is introduced. Its value combines a product\textsubscript{id} of the Uni-Pay-Req message, a payee’s system identifier and an index number issued by the UMS, in order to identify what has been paid for via the UMS and what content will be delivered to the buyer from the vendor. The uniMicro parameter refers to the exact value that will be transferred in the universal transaction. It may differ from the value parameter of the Uni-Pay-Req message sent from a local system, in terms of amount (i.e. it may be converted) and currency code.

Other parameters, such as the purchasing\textsubscript{ID}, version of the system in use, URL of the system portal, encrypted content URL, etc., are not playing critical roles in processing a universal transaction, so they are ignored at UTL. Table 5.4 lists the primary parameters for the Uni-Trans-Req interaction and also presents the example values of those parameters.
Table 5.5 Uni-Trans-Req parameters at UTL

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Sample value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Payer_{ID}$</td>
<td>JohnWell, 1976380#pwd : 1019wc</td>
</tr>
<tr>
<td>$Payee_{ID}$</td>
<td>N303 : musicrockandroll.net.au</td>
</tr>
<tr>
<td>$Content_{ID}$</td>
<td>9980-51500mzrr : NP528 : u320</td>
</tr>
<tr>
<td>$uniMicro$</td>
<td>0.55 : US</td>
</tr>
</tbody>
</table>

The payer of MPS-buyer and the payee of MPS-vendor are stored and maintained by the existing MPSs, whilst $Payer_{uni}$ and $Payee_{uni}$ are housed by both MPSs and UMS. Both payer of MPS-buyer and payee of MPS-vendor are determined by their systems and are also associated with the parameters $Buyer_{ID}$ and $Vendor_{ID}$, respectively. For example, a $Payer_{ID}$ could be a combination of $Buyer_{ID}$ and the identifier of the MPS that the buyer signed up with, while a $Payee_{ID}$ could be constructed with $Vendor_{ID}$ and vendor’s name or vendor’s certificate or signature, etc. No matter what information forms these identifiers, $Payer_{ID}$ of the MPS-buyer and $Payee_{ID}$ of the MPS-vendor should have a one-to-one relationship to the $Buyer_{ID}$ and $Vendor_{ID}$, respectively, such that they maintain uniqueness to their corresponding MPSs. The identifiers of $Payer_{uni}$ and $Payee_{uni}$ are issued by the UMS, so they are unique to the MPSs.

However, different from the $Buyer_{ID}$ and $Vendor_{ID}$ parameters, the $Payer_{ID}$ and $Payee_{ID}$ parameters only need to uniquely identify debtor and creditor accounts within the system they belong to. It is unnecessary to remain their uniqueness to the UMS because a buyer or a vendor may have already signed up with more than one MPS.
5.4.2 Universal transaction confirmation DEI

The UMS is responsible for executing the *Uni-Trans-Resp* interaction after a *Uni-Trans-Req* has occurred and its acceptance has been confirmed. With this interaction at UTL, the payee will be credited by its MPS after the completion of a universal transaction. The payee is also not able to refuse or deny the confirmed transaction, so that if the *Uni-Trans-Req* interaction is reliable and consistent, non-repudiation will happen. The UMS uses the *Payer* specified in the *Uni-Trans-Req* message and stores the confirmation parameters for tracing the universal transaction across multiple MPSs.

The message carried with the *Uni-Trans-Resp* interaction contains confirmation parameters, which provide the necessary information to the MPS-vendor, which is then able to inform the vendor of the completion of the universal transaction. In order to formulate the universal transaction protocol, three primary parameters for the *Uni-Trans-Resp* are defined and correlated to the parameters of the *Uni-Pay-Resp* message. They include *Content*, *uniMicro* and *uniTrans*. The process of *Uni-Trans-Resp* is executed after the completion of the money being transferred from the given debtor account to the creditor account. Table 5.5 lists the primary parameters for the *Uni-Trans-Resp* interaction and also presents the example values of those parameters.

**Table 5.6 Uni-Trans-Resp parameters at UTL**

<table>
<thead>
<tr>
<th>Primary parameters</th>
<th>Parameter</th>
<th>Sample value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>Content</em></td>
<td>9980-51500mzrr : NP528</td>
</tr>
<tr>
<td></td>
<td><em>uniMicro</em></td>
<td>0.68 : AU ( US : 0.55eq)</td>
</tr>
<tr>
<td></td>
<td><em>uniTrans</em></td>
<td>C9980-2007323-1655 : webcoin-1019wc</td>
</tr>
</tbody>
</table>
Note that the *Uni-Trans-Req* and *Uni-Trans-Resp* interactions occur between the payer of MPS-buyer and the *Payee\textsubscript{uni}* of the UMS, and between the *Payer\textsubscript{uni}* of the UMS and the payee of MPS-vendor.

### 5.5 Chapter summary

This chapter explained in depth of the UMS architecture design, which is based on the UMS requirements defined in the previous chapter.

To process the universal payment and the cross-payment transaction, possible system interaction approaches are discussed and suitable approaches are determined. Two layers of the UMS architecture (UAL and UTL) are then identified, where a chain of interactions between the UMS and the MPSs occur. The interactions at UAL inherit the common access methods used by existing MPSs, whilst interactions at UTL extend the data exchange to map multiple MPSs for universal cross-system transaction.

The interactions at both UAL and UTL also enable the UMS payment service to fully cooperate with and integrated seamlessly into multiple MPSs by defining and mapping primary parameters, such that existing MPS services could be incorporated with the UMS service, without requiring mass functional modifications or system upgrading. In addition, the advantage of the determined interaction approaches and the UMS architecture make it easy and efficient for the design of the UMS protocol, which is presented in the next Chapter.
Chapter 6  UMS payment protocol

This chapter presents the detailed protocol design for the Universal Micropayment System (UMS), and also describes the process of the universal micropayment transaction across multiple Micropayment Systems (MPSs) under such protocol. The purpose of the UMS protocol is to define rules of data exchange for the interaction gaps at Universal Access Layer (UAL) and Universal Transaction Layer (UTL), which were introduced in Section 5.3 and 5.4, respectively.

To improve the understanding of the protocol design, a brief introduction of the UMS protocol is given in Section 6.1, where the difference between the MPS service and the UMS service is discussed first. To realize the protocol design, the protocol entities and their associated parameters are identified and presented in Section 6.2. Then, the protocol behaviours (or functions) of the identified entities are determined; they form Section 6.3, where the suitable solutions on how to execute those functions are discussed. The protocol messages carried by each function are also presented in this Section. Section 6.4 summarises the chapter.

6.1 Introduction to UMS protocol

The proposed UMS requires a universal payment protocol to regulate the system cooperation and integration of universal payment service to existing MPSs. To understand the protocol behaviours and related functions, it’s needed to first analyse
the service differences between the UMS and the existing MPSs.

Within a general micropayment context, the UMS payment service is quite similar to the micro-transaction services provided by existing MPSs. Similar services are those such as the process of debiting payee and crediting payee, the process of transaction request and response messages, and the process of data exchange for system-compatible parameters, etc. Comparing the payment interactions which occur at two layers of the UMS with those of existing systems, it is noticed that there is no big difference at the level of UAL. It indicates that current MPS users do not need to significantly change their way of accessing their accounts, and making and receiving payments. However, at the level of UTL, though the direction of message transfers remains the same, the number of interactions increases due to the involvement of the UMS service.

The implementation of the proposed protocol will involve an installation of the Universal Micropayment Portal (UMP) – a UMS software application, which is built upon and integrated with the participating MPSs. The UMP interconnects the accounts from multiple systems, enabling the data exchange between payer and payee of MPSs to comply with the protocol. The core component of the UMP also plays a payee role at the MPS of the buyer side (MPS-buyer) and a payer role at the MPS of the vendor side (MPS-vendor). It will be further discussed in the later sections.
6.2 UMS protocol entities

At the start of the protocol design, protocol entities needed to be defined, which could lead to protocol behaviours and associated functions performed by those entities. Those protocol behaviours and functions will be discussed in Section 6.3.

The UMS protocol layers were introduced in Section 5.2. The primary parameters related to UAL and UTL of the UMS architecture were also identified in the previous chapter. Based on the design of the UMS architecture, the primary UMS protocol entities are identified and defined as follows:

- Debtor entity
- Creditor entity
- Property entity
- Token entity
- Reference entity

The following sub-sections will discuss those entities with their associated parameters in further detail.

6.2.1 Debtor

The debtor protocol entity contains buyer and payer identifiers at the different UMS layers, UAL and UTL, respectively.

At the level of UAL, the buyer identifier \( Buyer_{id} \) parameter of the \textit{Uni-Pay-Req}
Data Exchange Interaction (DEI) is a unique identifier, which identifies the account of the buyer for the UMS. Actually, this account is stored and maintained within the existing MPSs that are subscribed to by the buyer. The \( \text{Buyer}_{ID} \) needs to be authenticated within the MPS-buyer before performing a payment initiation (see Section 5.3.1 for details).

At the level of UTL, the payer identifier (\( Payer_{ID} \)) parameter of the Uni-Trans-Req DEI is unique for a specific MPS, and identifies a debtor account within this MPS. This identifier is also stored and maintained within the related MPS, which is incorporating with the UMS. The purposes of the debtor account identifiers for the underlying MPSs are as follows:

- The \( Payer_{ID} \) for the MPS-buyer is determined based on the \( Buyer_{ID} \). Since there is one-to-one relationship between these IDs, so they refer to the same debtor account.

- The \( Payer_{ID} \) for the MPS-vendor is the debtor account identifier, which is actually a universal UMP account (\( Payer_{uni} \)) within the MPS-vendor. This identifier is determined by and one-to-one related to the MPS-vendor, because different MPS-vendors have different \( Payer_{uni} \).

Table 6.1 presents definitions and descriptions of the parameters related to debtor entity at both levels.
Table 6.1 *Debtor entity related parameters at UAL and UTL*

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Buyer_{ID}$</td>
<td>Unique identifier determined by the MPS-buyer and issued to the buyer. This ID is used to authenticate the buyer and to determine the payer within the MPS-buyer. The value of this parameter is a combination of user name and password decided by the buyer.</td>
</tr>
<tr>
<td>$Payer_{ID}$</td>
<td>Unique identifier determined by both the MPSs and the UMS, and issued to the debtor account. This ID is used to authenticate the debtor of its belonged system, and to determine the debtor account of a universal micropayment. The value of this parameter is a combination of values of $Buyer_{ID}$ and its ID of the MPS-buyer ($MPS_{ID}$). Denoted as: $Payer_{ID} = { Buyer_{ID} : MPS_{ID} }$ for MPS-buyer</td>
</tr>
</tbody>
</table>

6.2.2 Creditor

The creditor protocol entity contains vendor and payee identifiers at the different UMS levels, UAL and UTL, respectively.

At the level of UAL, the vendor identifier ($Vendor_{ID}$) parameter of the *Uni-Pay-Req* DEI is a unique identifier, which identifies the account of the vendor for the UMS. This account is stored and maintained within the existing MPSs chosen by the vendor. The $Vendor_{ID}$ needs to be verified (i.e. the process of checking the existence of vendor’s account in the MPS-vendor) at the UMS protocol layer (see Section 5.3.2 for details). This ID is also used by the MPS-vendor to send payment confirmation at completion.

At the level of UTL, the payee identifier ($Payee_{ID}$) parameter of the *Uni-Trans-Req*
DEIs is unique for a specific MPS, and identifies a creditor account within this specific MPS. This account is also stored and maintained within the related MPS which is incorporating with the UMS. The purposes of the creditor account identifiers for the underlying MPSs are as follows:

- The `Payee_ID` for the MPS-buyer is the creditor account identifier, which is an UMP account (\textit{Payee}_{uni}) within the MPS-buyer. This identifier is determined by and one-to-one related to the MPS-buyer, because different MPS-buyers also have different \textit{Payee}_{uni}.

- The `Payee_ID` for the MPS-vendor is determined based on the \textit{Vendor}_{ID}, since there is also one-to-one relationship between these IDs, so they refer to the same creditor account.

Table 6.2 presents definitions and descriptions of the parameters related to creditor entity at both levels.

\begin{table}[h]
\centering
\begin{tabular}{|l|p{18cm}|}
\hline
\textbf{Parameters} & \textbf{Description} \\
\hline
\textit{Vendor}_{ID} & Unique identifier determined by the MPS-vendor and issued to the vendor. This ID is used to identify the vendor and to determine the payee within the MPS-vendor. The vendor provides this ID to buyer before a universal micropayment initiation is performed by the buyer. \\
\hline
\textit{Payee}_{ID} & Unique identifier determined by both the MPSs and the UMS, and issued to the creditor account. This ID is used to identify the creditor of its belonged system, and to determine the creditor account of a universal micropayment. The value of this parameter is a combination of values of \textit{Vendor}_{ID} and it’s ID of the MPS-vendor (\textit{MPS}_{ID}). Denoted as: \textit{Payee}_{ID} = \{\textit{Vendor}_{ID} : \textit{MPS}_{ID}\} for MPS-vendor \\
\hline
\end{tabular}
\caption{Creditor entity related parameters at UAL and UTL}
\end{table}
6.2.3 Property

The property protocol entity contains product and content identifiers, presented at the different UMS levels, UAL and UTL, respectively.

At the level of UAL, the buyer receives a product identifier (\textit{product}_{ID}) parameter from the vendor prior to a payment. The buyer initiates a universal payment using this identifier. The UMS returns this identifier in a confirmation message to the vendor. The vendor will then know which product has been paid for and needs to be delivered.

At the level of UTL, the content identifier (\textit{Content}_{ID}) parameter will be used to identify the paid content. In fact, the \textit{product}_{ID} may be unique to its transaction performed by the MPS, but it may not be unique to the universal transaction performed by the UMS. Therefore, a conversion process is required at this level in order to keep it unique within the UMS. The protocol function of mapping \textit{product}_{ID} to \textit{Content}_{ID} parameter within the UMP will be further described in Section 6.3.5. This function also makes this parameter transferable across the MPSs involved, such that the \textit{Content}_{ID} parameter for universal micropayment transaction could be uniformed and globally unique (see Section 5.4.1 for how to relate the \textit{product}_{ID} onto the \textit{Content}_{ID} at the UTL).

Table 6.3 presents the determination of the parameters related to paid content entity at both levels.
Table 6.3 Property related parameters at UAL and UTL

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$product_{ID}$</td>
<td>Unique product identifier determined and generated by the product database system of the vendor. The vendor provides this ID to the buyer prior to each payment initiation.</td>
</tr>
<tr>
<td>$Content_{ID}$</td>
<td>Unique identifier determined and generated by the UMS. It is associated with $product_{ID}$ in the UAL, identifying what has been paid for. It will be stored in the UMS for universal payment tracing purpose. The value of this parameter is a combination of values of $product_{ID}$, vendor’s $MPS_{ID}$ and an index number ( $Index_{UMS}$ ) generated by the UMS. Denoted as: $Content_{ID} = { product_{ID} : MPS_{ID} : Index_{UMS} }$</td>
</tr>
</tbody>
</table>

6.2.4 Token

The token protocol entity refers to the electronic cash (or micro-money) transferred for micropayment. It contains value and uniMicro parameters at the different UMS levels, UAL and UTL, respectively. Because the minimum and maximum amounts, as well as the currencies supported by multiple MPSs may vary, the micro-money specified in the UAL and UTL will be different.

At the level of UAL, the value of the product is determined by the vendor’s database and sent to the buyer prior to a micropayment.

At the level of UTL, a token is presented in terms of currency and amount of value. As one of the requirements defined in Section 4.3.3 for the UMS design, the UMP has to inherit the features of the existing MPSs, such as the supported type of currencies and supported range of amounts, and so on. Therefore, the UMP needs to solve the
parameter difference between those levels. This means that it needs to verify whether the amount of value specified in the Uni-Pay-Req is supported and transferable by the participating MPSs at the UTL (See Section 5.4.1 for details on how to convert currency and value). The UMS will execute the logical algorithms according to the following circumstances:

- If the currency is not supported by either of the integrated MPSs, the UMP needs to exchange the currency, and then verify whether the exchanged amount of money is supported by both integrated MPSs; and
- If the specified amount of money is not supported by either one of the integrated MPSs, the UMS cannot transfer such amount and then the universal payment initiation will be rejected.

Table 6.4 presents the determination of the parameters related to token entity at both levels.

Table 6.4 *Token entity related parameters at UAL and UTL*

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Value</em></td>
<td>The value parameter will be sent to the buyer by the vendor along with the <code>product_id</code> prior to each payment initiation. It is determined by the product database system of the vendor’s site.</td>
</tr>
<tr>
<td><em>uniMicro</em></td>
<td>The required value of content, in terms of currency and amount of the value, will be transferred across multiple MPSs via the UMS for a universal transaction.</td>
</tr>
</tbody>
</table>
6.2.5 Reference

Similar to the token identifier, the reference entity has two parameters related at different levels. They are the payment identifier (\( Payment_{ID} \)) parameter and the universal transaction identifier (\( uniTrans_{ID} \)) parameter, which contain different values at UAL and UTL.

At the level of UAL, the \( Payment_{ID} \) is a unique identifier generated by a specific MPS, allowing the users to trace transferred micropayments within their MPSs. The vendor will store this parameter in most cases.

At the level of UTL, the \( uniTrans_{ID} \) serves a similar purpose, but it is generated by the UMS in order to trace universal micropayments for the specific MPSs. The UMP receives the \( Payment_{ID} \) in \textit{Uni-Pay-Resp} from the MPS-buyer, and then the UMP needs to map it to a unique \( uniTrans_{ID} \). To enable the tracing purpose, the \( Payment_{ID} \) and \( uniTrans_{ID} \) parameters should be associated to each other. The storage of the two identifiers in the UMS is unnecessary, because both the vendor and MPS-vendor will store the \( Payment_{ID} \). Consequently, the UMS only needs to retain the \( uniTrans_{ID} \).

Since the volume of universal payments may be very high, and each payment value is comparably low, the storage period could be relatively short. It could be settled at the pre-regulated business agreements, so that it may differ from one MPS to another.

Table 6.5 presents the determination of the parameters related to payment reference entity at both levels.
### Table 6.5 Reference entity related parameters at UAL and UTL

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Payment_{id}$</td>
<td>Unique payment identifier generated by and stored in the MPS. It can be used to trace the universal payments by local MPSs, whenever conflict situations occur between buyer and vendor.</td>
</tr>
</tbody>
</table>
| $uniTrans_{id}$ | Unique transaction identifier generated by and stored in the UMS. It can be used to trace the universal transactions, whenever conflict situations occur between the MPSs and the UMS. The value of this parameter is a combination of a $Payment_{id}$ and $MPS_{id}$.
Denoted as: $uniTrans_{id} = \{ Payment_{id} : MPS_{id} \}$ |

### 6.3 Protocol functions and messages

The protocol entities have been defined and presented in the previous section. Because they are reciprocally dependent, in order to realize these protocol entities, it is necessary to specify their relationships, associated behaviours and related functions. Within the UMS, each protocol entity is a protocol function carrier and executer. Given the characteristics of those entities, their behaviours could be grouped into particular protocol functions. This section will discuss those inter-related functions and present them in a systematic way. The primary protocol functions are identified and listed as follows:

- **Authenticating buyer** – a function to check the authentication of the buyer during the process of a universal payment initiation (explained further in Section 6.3.1);
- **Identifying vendor** – a function to validate the identification of the vendor after
a *Uni-Pay-Req* interaction is initiated (explained further in Section 6.3.2);

- **Allocating accounts** – a function to determine the payer and payee identifiers for the debtor account and creditor account, respectively (explained further in Section 6.3.3);

- **Verifying money** – a function to verify whether the multiple MPSs support the currency and amount specified in the *Uni-Pay-Req* message. This function may also involve a currency conversion (explained further in Section 6.3.4);

- **Creating a unique content ID** – a function to determine a universal content identifier with universal uniqueness for a universal payment (explained further in Section 6.3.5);

- **Transferring payment information** – a function to initiate transactions between the MPSs and the UMS, and generate a unique $\text{uniTrans}_ID$ (explained further in Section 6.3.6); and

- **Storing payment information** – a function to select and store payment information in the UMS (explained further in Section 6.3.7).

The possible approaches to assign those protocol functions will be discussed in the following sub-sections. The function assignment will be made as simple as possible, which means the original functions of the existing MPSs will be not be overloaded, such that the implementation of the UMS and its’ payment portal built on top of the those MPSs (demonstrated in the next chapter) become more realistic and achievable.
6.3.1 Authenticating buyer

This protocol function is to check the authenticity of buyers during the process of the Uni-Pay-Req interaction. A universal micropayment could be initiated only if the buyer’s identity is proved genuine. Otherwise, any further payment interaction will be denied. This process is usually experienced by and familiar to buyers because they should have to login to the local system they used so far to access their money within their MPS. The authentication process is a starting function defined in functional requirements in Section 4.4. There are two approaches that the function could be realized: authentication taking place in the UMS or in MPS-buyer.

When the buyer’s authentication is processed by the UMS, the UMS then needs a very large central database to store and maintain all of the buyer’s information. As it allows the access to buyer’s money, this valuable information should be protected by proper security mechanisms and efficient access control management, such that this approach will enormously increase the security and administration burden at the UMS side. Furthermore, because of the huge volume of the existing registered buyers and rising new buyers of current MPSs, the process of authenticating buyers within the UMS will generate a bottleneck in retrieving information from the UMS database. In addition, the local MPSs also need to maintain their customer’s detail for authentication and other commercial purposes, and the administration and authentication of buyer’s information at the UMS side will cause unnecessary data redundancy. Also, the system brokers will unwillingly disclose their database to other third-party system operators. Due to the above mentioned disadvantageous factors, this approach seems unrealistic and hard to achieve.
There are many advantages of carrying out the buyers’ authentication within the MPSs of their choice. The authentication process is already functioning and managed by the participating MPSs during their local system operation. Also, the MPSs may have their own security methods to protect their customer’s information. By using this approach, the UMS database only needs to maintain information of those participating MPSs, rather than information of all buyers. As a result, the data storage requirement for the UMS is minimized and the security threats are also avoided. Given the above factors, this approach is applied to authenticate buyers for universal micropayment transactions.

6.3.2 Identifying vendor

This protocol function is to validate the identification of vendors after the *Uni-Pay-Req* interaction is initiated. It simply means to check the existence of the vendors’ account at the MPS of their choice. A universal micropayment could be initiated only if the vendor’s identity is recognized by the participating MPS. Otherwise, any further payment interaction will be denied. There are also two approaches that the function could be performed: verification taking place in the UMS or in MPS-vendor.

Similar to the buyer’s authentication, the process of identifying vendor at the UMS side requires the UMS to not only provide additional functionality, but also to store all the vendors’ information in its central database, which, in this case, needs to be accessed for each universal transaction. Due to the fact that retrieving a vendor’s data may create huge task and require a much longer time lag to complete a universal
transaction; and also due to the fact that unnecessary data redundancy is caused by this approach, we consider the identification task be performed at the vendor’s MPS.

Since the MPS-vendor keeps and maintains the vendor’s information, identifying a vendor within its MPS is an easy task. According to the literature review, verifying vendors before each payment is one of the common procedures conducted by most of examined MPSs. Additionally, depending on the way they store and manage their vulnerable data, the identification process and method varies from system to system. Compared with the previous approach, this alternative is the suitable solution for identifying the vendor.

However, this identification approach results in the involvement of an additional interaction. This interaction could be completed from the MPS-buyer to the MPS-vendor or from the UMP to the MPS-vendor, because both the MPS-buyer and the UMS will receive the VendorID before the Uni-Pay-Req interaction takes place. The additional interaction refers to that fact that the VendorID needs to be sent to its’ MPS for verifying the existence of vendor’s account.

Considering that the participating MPS-vendor has contracted with the UMS, but has no agreement made with participating MPS-buyer, for the trust concern, it’s better for the UMS to request the MPS-vendor for the identification check. Furthermore, the identification has to be verified successfully by the MPS-vendor before the Uni-Trans-Req initiates, otherwise, the buyer may bear the money loss. In fact, the identification task could be simply fulfilled by the local MPSs in the usual way they identify their users. The process of this interaction could also be easily realized by
general forms of HTTPs request and response. Consequently, this additional interaction is extended into the whole manner of a universal micropayment.

6.3.3 Allocating accounts

This protocol function is to determine which account has to be debited and which be credited after a *Uni-Pay-Req* interaction initiates. A universal payment could be initiated only if the debtor and creditor accounts are successfully determined by the UMS. Otherwise, the transferred value will be led nowhere, which may finally result in one of the parties risking the loss of money. As mentioned in Section 5.4.1, the payer and payee parameters are actually account identifiers, so this function is actually determining the *Payer*<sub>ID</sub> of debtor accounts and the *Payee*<sub>ID</sub> of creditor accounts for both the buyer and vendor’s MPSs.

As mentioned in Section 6.2.1 and 6.2.2, the *Payer*<sub>ID</sub> of the MPS-buyer and the *Payee*<sub>ID</sub> of the MPS-vendor are determined by and one-to-one related with the *Buyer*<sub>ID</sub> and *Vendor*<sub>ID</sub>, respectively. The *Payee*<sub>ID</sub> of the MPS-buyer and the *Payer*<sub>ID</sub> of the MPS-vendor are actually the UMS accounts contracted with both systems, referring to the creditor account of the MPS-buyer and debtor account of the MPS-vendor, respectively. As a consequence, their identifiers depend on their related *MPS*<sub>ID</sub>, which should be identified by the UMS before each *Uni-Trans-Req* interaction occurs.

To realize that, it requires the UMS have a database table containing *Payer*<sub>uni</sub> and *Payee*<sub>uni</sub> parameters for all participating MPSs.
Figure 6.1 shows the relationship between payers and payees in different MPSs. It is noted that the UMS portal stands between the MPS-buyer and MPS-vendor, playing the payee’s role for the former and payer’s role for the latter, respectively.

![Figure 6.1 Payers and Payees in different MPSs](image)

One of the purposes of this function is to preserve the anonymity of the buyer, which is one of the user’s requirements formulated in Section 4.1.5. Since the cross-system transactions are executed at a universal level, where only the payer and payer’s identifiers are exchanged, rather than buyer and vendor’s information, neither the UMS nor MPS-vendor will be able to know who is actually purchasing the content.

### 6.3.4 Verifying money

This protocol function is to check whether the value specified in the *Uni-Pay-Req* message is supported by both buyer and vendor’s MPSs. As defined in Section 5.3.1, the *value* parameter is determined by vendor and sent to buyer. If either type of the value (currency) or amount of value could not be satisfied by both systems, the further *Uni-Trans-Req* will discontinue. The checking algorithm could be performed either in
the MPS-buyer or in the UMS. This process could be divided into two steps: checking compatible currency and then checking the supported amount.

According to the current MPSs investigated in the literature, some systems run local currency only, whilst many intend to support multi-currencies. In order to realize the universal micropayment, a fixed currency exchange mechanism needs to be unified and followed by all participating MPSs. Therefore, the exchange rate has to be settled and updated by either an agreed bank or by a certain exchange market. This agreement should be made between the UMS and MPSs as a part of their contract settlement (this will be excluded from the scope of this thesis because only the technical concerns are the focus of this research).

With respect to the amount, current literature found that all studied MPSs have their default minimum and maximum transaction value supported. Given the characteristics of MPSs, the critical factor of performing this function seems to be where the verification is done, instead of how it is done. We have identified that both the MPSs and the UMS have the capability to handle this. If handled by the MPSs, one of the concerns is the risk of money loss on buyer’s side when dishonest vendors change their supported value settings. If we assume the fact that the UMS is fully trusted by contracted MPSs, then the UMS would be a better function performer than a local MPS.

To fulfill this function, the UMS needs to have a special table in its central database to contain information about minimum and maximum supported amounts of each MPS, as well as their default currency. It could have added attributes to each MPS record,
since the UMS already has table fields for storing MPS’s data, like $MPS_{id}$, name, version, etc. The UMS also needs an individual table to contain the currency exchange rate or a link to it, where exchange calculation automatically functions.

To execute such function, the value parameter is first extracted from the Uni-Pay-Req message; if the default currencies supported by both MPSs differ, the currency will be converted, based on the agreed exchange rate until it’s supported by both systems. After that, the amount of value is checked; if the original value or the converted amount is acceptable by both systems (for this, the minimum and maximum transfer amount of both systems will be compared), then the Uni-Trans-Req initiation could continue. Otherwise, if either one of the conditions is unsatisfied, the universal transaction stops.

### 6.3.5 Creating unique content identifier

This protocol function is to map the product identifier (of local MPS) to the universal content identifier (of the UMS) before a universal transaction is initiated. This function will also keep the uniqueness of the $Content_{id}$ for each UMS payment. As introduced in Section 5.4.1, the $Content_{id}$ is associated with the $product_{id}$ that is determined and issued by the vendor and sent to the buyer prior to a payment. In order to keep the uniqueness of such parameter, both to the UMS and the MPSs, the creation of this parameter will combine together the original $product_{id}$, the $MPS_{id}$ of the MPS-vendor and a unique UMS index number ($Index_{UMS}$). 

There are several concerns of using such a combination method to convert the $product_{id}$ to $Content_{id}$, rather than directly using $product_{id}$. First, the $product_{id}$ might not be uniquely generated by the vendor, or a same $product_{id}$ might be used by more than one vendor. Second, suppose it’s unique to the vendor, who might register with more than one MPS, but keep using the same $product_{id}$ to accept micropayments. In that case, duplicate situations will occur at the universal payment (if without the value of the $Index_{UMS}$) when the buyer pays for the same product more than once. Third, it’s complicated for database administration (at both the UMS and MPS-vendor) and hard to trace universal transactions when a conflict situation occurs.

The function of creating a new $Content_{id}$ parameter and assigning value to it will be performed by the UMS after receiving a $Uni-Pay-Req$ message that contains the $product_{id}$. The UMS will then extract this parameter value, and generate a random (or sequential) index number that is uniquely used for this parameter, and combine them together with the $MPS_{id}$ of the MPS-vendor that is already known to the UMS, and finally constitute the $Content_{id}$.

### 6.3.6 Transferring payment information

This protocol function is to perform a universal micropayment and confirm to the vendor at the completion of a universal transaction. The process of the function refers to the $Uni-Trans-Req$ and $Uni-Trans-Resp$ interactions performed at the participating MPSs with the involvement of the UMP. Prior to those interactions, the UMP receives a $Uni-Pay-Req$ message from the buyer.
Before this function is executed, the following conditions need to be fulfilled:

- The buyer’s identifier is authenticated and associated account \((Payer_{ID})\) is identified (the result of the protocol function in Section 6.3.1);
- The vendor is verified, and associated account \((Payee_{ID})\) currently exists and valid (the result of the protocol function in Section 6.3.2);
- The micro money of the universal payment, in term of currency and amount of value are verified, currencies are matched and amount are supported by both MPSs (the result of the protocol function in Section 6.3.4); and
- The unique \(Content_{ID}\) is generated and its value is mapped from \(product_{ID}\) (the result of the protocol function in Section 6.3.5).

In fact, the above functions are requested by the MPS-buyer, and performed by the UMS. Such a serial of Universal Verification Requests \((Uni-Verif-Req\) in short) result in a Universal Verification Response \((Uni-Verif-Resp\) in short), both of which are taking place at the protocol layer, where the UMP is involved to transfer those messages.

Then, the UMS will send the \(Content_{ID}, Payee_{ID}\) and supported micro money to the MPS-buyer in the form of a \(Uni-Verif-Resp\), a message response to \(Uni-Verif-Req\). Although this is an additional chain of interactions between the UMS and MPSs, it’s necessary, because the payer of the MPS-buyer could initiate the \(Uni-Trans-Req\) only after receiving those data.

From this point on, the universal transaction is realized by means of executing the \(Uni-Trans-Req\) and \(Uni-Trans-Resp\) interactions. The transfer ends after the vendor
has confirmed the completion of transaction with a reference. Therefore, this function could be divided into two main sub-tasks: initiating a universal transfer and generating universal transaction ID.

For the first task, the initiation of universal transaction occurs at both MPS-buyer and MPS-vendor. The Figure 6.2 shows the interaction flow of a universal payment, including the process of the Uni-Trans-Req and Uni-Trans-Resp. The first pair of transaction request and response occurs from the MPS-buyer to the UMP, and the second pair occurs from the UMP to the MPS-vendor. The UMP here bridges the parameter gap between two MPSs.

![Interaction flow of universal micropayment](image)

**Figure 6.2 Interaction flow of universal micropayment**

For the second task, the universal transaction identifier needs to be unique to each universal payment. Both the UMS and MPSs are able to generate uniTransID uniquely. Considering that the PaymentID generated by local MPSs is also unique to each micropayment processed; and also due to the fact that combining the unique MPSID and local unique PaymentID of MPSs could create an identifier with global uniqueness, this combination method will be used to determine a universal
\[uniTrans\_ID\]. Therefore, this sub-task is to map the \[Payment\_ID\] (of local MPS) to the universal \[uniTrans\_ID\] (of the UMS) after a universal transaction is initiated. As the \[uniTrans\_ID\] is associated with the \[Payment\_ID\], the creation of the \[uniTrans\_ID\] parameter will combine together the original \[Payment\_ID\], the \[MPS\_ID\] of the MPSs that process the \[Payment\_ID\].

Figure 6.3 extents the architecture diagrams presented in Figure 5.2 and illustrates the UMS protocol, the payment interactions at both UAL and UTL (not shown in the diagram), and the associated parameters (defined in section 5.3 and 5.4) of both protocol layers.

![Figure 6.3 Universal micropayment interactions with associated parameters](image)

Figure 6.3 *Universal micropayment interactions with associated parameters*
6.3.7 Storing payment information

This protocol function is to selectively store the information involved in all interactions of a universal payment transaction within the UMS database. With this function, the universal micropayments are made traceable and accessible for further business statistical analysis. In order to reduce the data redundancy and support efficient data retrieval, the optimized methods for data selecting and storing are taken into account.

In order to efficiently trace universal transactions, minimize data redundancy, and make the database easy to maintain, only the uniquely determined parameter values are selected and stored in the UMS database. Therefore, two subsequent universal interactions with valuable parameters are identified; they are the \textit{Uni-Trans-Resp} from the MPS-buyer to the UMP, and the \textit{Uni-Trans-Req} from the UMP to the MPS-vendor. However, the local MPSs may not need to store such additional information, because the payment information they stored within their database could easily relate to the universal payment information stored in the UMS. Table 6.6 lists the parameters stored in the UMS database after each universal transaction completes.

\begin{table}[h]
\centering
\caption{Parameters stored in UMS database for each universal transaction}
\begin{tabular}{|c|c|c|c|}
\hline
\textit{Uni-Trans-Resp} & \textit{Content}_\text{ID} & \textit{uniMicro} & \textit{uniTrans}_\text{ID} \\
from MPS-buyer to UMP & & & \\
\hline
\textit{Uni-Trans-Resp} & \textit{Payer}_\text{ID} & \textit{Payee}_\text{ID} & \textit{Content}_\text{ID} & \textit{uniMicro} \\
from UMP to MPS-vendor & & & \\
\hline
\end{tabular}
\end{table}
6.3.8 Summary of protocol functions

The process of a universal micropayment starts after the buyer initiates a payment request, and ends after it is confirmed to the appropriate vendor. The universal transaction is realized by a chain of protocol functions, which if any fails will result in the failure of the payment. Those protocol functions require the involvement of the UMS portal – UMP and MPSs; whilst some functions are executed within the UMP only. Those functions are quickly performed in a sequential order with an insignificant time lag, such that one universal transaction might be completed in a very short time. Figure 6.4 illustrates the sequence of protocol functions involved for a universal transaction.

![Protocol function sequence diagram]

**Figure 6.4 Protocol function sequence**

The messages carried out with those protocol functions need to be transferred through systems via the open network (i.e. the Internet), such that the secure HTTP transfer for
HTTP request and response are an essential requirement. To better understand those protocol functions, a Swim lane diagram is provided in Figure 6.5 as to clearly state the process flow of a universal micropayment transaction. Table 6.7 gives the denotation for the shapes within the diagram.

<table>
<thead>
<tr>
<th>Buyer</th>
<th>MPS of buyer</th>
<th>UMS</th>
<th>MPS of vendor</th>
<th>vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buyer initiates a payment to Vendor</td>
<td>MPS verifies the buyer</td>
<td>UMS verifies the money</td>
<td>MPS verifies the vendor</td>
<td>Notifies vendor &amp; a transaction ends at the local system</td>
</tr>
<tr>
<td>Authentication fails &amp; transaction fails</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Notifies vendor &amp; a transaction fails</td>
</tr>
<tr>
<td>Notifies buyer &amp; transaction ends at local system</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Notifies vendor &amp; a transaction fails</td>
</tr>
<tr>
<td>Buyer initiates a payment to Vendor</td>
<td>MPS verifies the buyer</td>
<td>UMS verifies the money</td>
<td>MPS verifies the vendor</td>
<td>Notifies vendor &amp; a transaction ends at the local system</td>
</tr>
<tr>
<td>Authentication fails &amp; transaction fails</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Notifies vendor &amp; a transaction fails</td>
</tr>
<tr>
<td>Notifies buyer &amp; transaction ends at local system</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Notifies vendor &amp; a transaction fails</td>
</tr>
</tbody>
</table>

**Figure 6.5 Swim Lane diagram for UMS protocol functions**
To successfully measure protocol functions, a number of assumptions will be made throughout the design phase. First, it’s assumed that the network for data exchange between the protocol entities is sustainable and always reliable during the payments; second, the buyer and vendor trust the MPSs of their choice, and the MPSs trust the UMS that they are incorporating with; third, the UMS maintains a database with structured tables, containing updated information about contracted MPSs, so as to enable cross-system interactions.

6.4 Chapter summary

In this chapter, the UMS protocol was defined for universal micropayment transactions. Under the design context, the functional aspects of the protocol are the main focus.

To better understand such protocols, the protocol-related entities were identified first, with related parameters’ definition and relationship. During the determination of protocol functions, the alternative solutions for realizing those functions were identified and analysed. The design of the protocol functions follows a basic principle,
which is minimizing the system modification for the participating MPSs and maximizing the significance of their original local functionalities. However, to realize the universal transaction, the UMS protocol layer should be built on top of existing MPSs, with an integrated UMP, to interconnect with accounts within MPSs for each cross-system transaction.

With regard to the functional requirements given in Section 4.4, Table 6.9 presents a traceability matrix to those requirements for the above protocol functions listed in Table 6.8.

**Table 6.8** List of protocol functions with section number

<table>
<thead>
<tr>
<th>Function</th>
<th>6.3.1</th>
<th>6.3.2</th>
<th>6.3.3</th>
<th>6.3.4</th>
<th>6.3.5</th>
<th>6.3.6</th>
<th>6.3.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function scalability (General)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Function reliability (General)</td>
<td>X</td>
<td>—</td>
<td>—</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Function implementability (General)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>User authentication (same methods)</td>
<td>X</td>
<td>X</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Trusty payment interface</td>
<td>X</td>
<td>X</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Customers initiate payment</td>
<td>X</td>
<td>—</td>
<td>X</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Jointly payment initiation method</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Merchant confirmed payment</td>
<td>—</td>
<td>X</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Customer confirmed payment</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Multiple currency</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>X</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Minimum value ranges supported</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>X</td>
<td>—</td>
<td>X</td>
<td>—</td>
</tr>
<tr>
<td>Maximum value ranges supported</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>X</td>
<td>—</td>
<td>X</td>
<td>—</td>
</tr>
<tr>
<td>Pre-paid authorisation method</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>X</td>
</tr>
<tr>
<td>Post-paid authorisation method</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>X</td>
<td>—</td>
</tr>
</tbody>
</table>

**Table 6.9** Traceability matrix to functional requirements
It is shown in the traceability matrix that general functional requirements in terms of scalability, reliability and implementability are fulfilled except the reliability issues for identifying vendor (6.3.2) and allocating accounts (6.3.3) functions. It is due to the fact that the reliability of those functions is subject to the vendor’s server and MPS of the vendor, any of which creating network bottleneck could result in a transaction failure. The matrix above is also showing that except the UMS internal functions (6.3.5 and 6.3.7), the rest could meet the specific functional requirements in a great extend.

The next chapter will describe the implementation of the proposed protocol functions and measure their performance by using case studies.
Chapter 7
Implementation and measurement

This chapter describes the implementation of the proposed Universal Micropayment System (UMS) in terms of its protocol functions, systems cooperation and service integration, and then to measure whether the predefined requirements are fulfilled or not. A case study approach is used to determine the system implementation and measurement. Section 7.1 presents two studied cases. To better understand the process of system implementation in a theoretical way, a typical universal payment scenario demonstration is provided in Section 7.2, to indicate how the UMS actually works. Though it could not be physically proved that the theoretical implementation could be suitable to all existing Micropayment Systems (MPSs) at this stage, from the literature point of view, the proposed interaction approaches used by the UMS could be achievable and feasible to most of the MPSs investigated so far. Therefore, this simulated implementation is used for the further analysis of estimated performance. Section 7.3 demonstrates the implementation of the UMS protocol, where two selected MPSs are used to examine the functionality of the protocol entities. Section 7.4 demonstrates the system cooperation and service integration through system parameter mapping. The evaluation is presented in Section 7.5 where the fulfilment of the requirements defined in Chapter 4 is measured. At the end, a summary is given for this chapter.
7.1 Case studies

We used case study approach to realize the implementation and measurement of the proposed UMS. The purpose of the case study is to investigate whether the implementations of the universal payment protocol and system cooperation are theoretically achievable. The case studies show the functionalities of the selected MPSs and also describe how existing systems could be cooperated and integrated with the UMS to realize the universal payment service. Finally, with the use of case studies, the performance of the UMS could be measured.

7.1.1 Case study selection

Two cases of MPSs were selected: Webcoin and NetPay. Their system functionalities and detailed transaction mechanisms are both available and testable at both the theoretical and laboratorial levels. The general descriptions about these MPSs are shown as follows.

Webcoin system was an internal ICT project proposed by Shao and Nguyen (2008) at Griffith University in 2005. It was created based on an early micropayment system called PayWord with improved security mechanism and new features added. It is a web-based, pre-paid system that enables low-value e-coins transferable for purchase of online digital content. NetPay system was proposed at Southern Cross University and
operated at university trial stage. The java based payment system was designed to support the micropayments transferred online to pay for those low-value goods. The payment model and security method used by the two systems are typical among the MPSs investigated.

To realize the universal payment cross multiple systems, slight modifications were made in both Webcoin and NetPay. An application component of the Universal Micropayment Portal (UMP) introduced in the previous chapter was built on top of both systems to integrate their existing payment transactions into universal transactions. The UMP component imbedded in Webcoin is named *uni-Webcoin* and that in NetPay is *uni-NetPay*. They are actually a system-independent application component. The purpose of the integrated component is to map system differences between Webcoin and the UMS, and between the UMS and NetPay. Because such integration only requires minimal upgrades on both systems and also allows their original functions to remain unchanged, the existing users of those systems would experience only minor changes of system usage, functionality, and the payment interface portals.

The demonstration of usage simulation based on the selected cases is presented in Section 7.2, which shows the process of a universal micropayment transaction between two MPSs through the UMS.
7.1.2 Implementation scenario

This UMS implementation scenario is based on the design of the protocol entities, functionalities and definition of protocol messages presented in Sections 6.2 and 6.3. The UMS is bridging the gap of cross-system payment interactions by interconnecting uni-Webcoin and uni-NetPay, to initiate a payment from the Webcoin system and to confirm a transaction to the NetPay system, respectively. As described in Section 6.3.7, the UMS database has a UMS table, universal transaction table and currency exchange table, where information about participating MPSs, the processed universal transactions, and the daily currency exchange rate, respectively, are stored.

7.1.3 Universal payment interaction scenario

In most cases, the Internet is the transmission channel for data exchange among two MPSs and the UMS. All the universal payment initiation and confirmation interactions are performed and implemented via the HTTPS protocol using request and response messages, as this is done by most existing MPSs in current practice. Some non-secured data might be transferred by HTTP messages, such as vendor’s name, content description, because there are, in most cases, less security concerns for those data. The IP address of the payment portals of the participating MPSs and the UMS are stored by the UMS and the MPSs, respectively.
7.2 Demonstration of universal micropayment via UMS

This section presents the typical process of a universal micropayment transaction in order to demonstrate how users experience the cross-system payment via the proposed UMS for purchasing online content. The following demonstration simulates the desired universal payment environment.

Within such environment, it is supposed that a buyer named Shaun found an online vendor – NewsOne.net, which sells low priced historical news. Assume Shaun is subscribed to use the Webcoin system and going to pay for a news archive through the Webcoin payment portal. The UMP application – uni-Webcoin, integrated with the Webcoin system, is responsible for incorporating with the UMS protocol to process universal payment transaction. Shaun will use the integrated Webcoin system to transfer cross-system micropayments to various vendors.

It’s assumed that the online vendor – NewsOne.net has signed up with the NetPay system, which is responsible for collecting online micropayments from online buyers and maintaining a commercial account for it. The UMP application – uni-NetPay, integrated with the NetPay system, is responsible for incorporating with the UMS protocol to process universal payment transactions. NewsOne.net will use the integrated NetPay system to receive cross-system micropayments from various buyers. Meanwhile, the vendor operates a web server which receives and processes HTTPS
request messages. It also maintains a database to store the information received in the payment confirmations from NetPay, so that the vendor knows which news content has been paid for.

Both the Webcoin and NetPay systems are integrated with the UMS payment portal applications built on top of them, so their users can pay each other regardless the different system used by the other side. Figure 7.1 simulates the interactions that occur between the buyer and vendor, and between the participating MPSs (Webcoin and NetPay) and the UMS. The index number of each interaction indicates their occurrence sequence, and the direction of the arrows shows the main process flow.

Figure 7.1 Demonstration of a universal micropayment
In the first interaction, Shaun searches the news database on NewsOne.net, and selects a title of historical news. Then, his browser sends a HTTPS request message to the web server of the news vendor.

The second interaction contains a HTTPS response message. NewsOne.net sends Shaun a web page that indicates 10 cents needs to be paid for that news. This page also contains the information, such as the vendor identifier of NewsOne.net, the unique product (news) identifier and an encrypted URL, etc. Those identifiers are needed for Shaun to initiate a universal payment. The encrypted URL provides access to the news, once the payment is completed. Afterwards, it starts the Webcoin payment application, which is the UMS payment portal imbedded into the original Webcoin payment interface. Shaun, at the same time, is advised to use the UMS payment portal to pay the vendor, who uses a different payment system other than Webcoin. Then, this interface will allow Shaun to enter his user name and password as usual, and also displays the related information from NewsOne.net, such as name of the vendor and value of the News, etc.

The third interaction refers to the process of the universal micropayment. After Shaun clicks on the ‘Pay’ button, the initiated payment information is submitted to the UMS in an HTTPS request message, and then the universal micropayment starts processing. The initiation of the cross-system payment is accepted by the UMS after the following four conditions are satisfied: (1) Shaun’s user name and password are correct and
authenticated by the Webcoin system, and he is authorised to use the UMS; (2) the vendor’s identifier is verified by NetPay, and the account of NewsOne.net exists and is currently in use; (3) both Webcoin and NetPay support the transfer of ten cents; and (4) Shaun has enough money in his account for this payment. After the successful payment interaction, Shaun will receive a decrypted link to the paid item (refer to interaction 5 and 6). If the payment initiation could not be accepted by the UMS, he will receive an error message instead.

In the fourth interaction, the web server of NewsOne.net receives a HTTPS response message with the payment confirmation. This message contains data, such as an identifier of the paid item, the paid amount, and an approval code, etc., all of which are sent from NetPay to the vendor, just in the usual way that NetPay confirms a payment to vendor. Then, NewsOne.net will save the confirmation data into its database.

In the fifth and sixth interactions, Shaun clicks on the decrypted URL link specified on the payment response page, downloads and views the news. As the payment has been confirmed, the request for accessing news from buyer and response to delivering news from vendor could be integrated into one interaction. It is noted that interaction 2 to 6, in some cases, depending on micropayment models used by MPSs, like ‘click & view’, could be integrated into one process without the buyer’s intervention, such that buyers could get the paid content immediately after a click.
7.3 Implementation of universal micropayment protocol

To realize the UMS, the implementation of its protocol is the first main task. Case studies are used to describe how the multiple MPSs incorporate with the UMS to process cross-system micropayments under the proposed protocol. As demonstrated in Section 7.2, two selected systems – Webcoin and NetPay, are built with UMP components as a payment portal, to provide a universal micropayment service to buyer and vendor.

These cases explain the operation of the proposed UMS; especially they describe how the protocol entities are functioning. As described in Chapter 6, the protocol behaviours could be decomposed into several functions (listed below). It is assumed that there are reliable and secure Internet transaction mechanisms for the universal payment service. Two integrated applications introduced earlier, uni-Webcoin and uni-NetPay, which are imbedded into Webcoin and NetPay, respectively, are used to perform the universal micropayment.

According to the protocol functions defined in Section 6.3, the implementation of the UMS transaction consists of the following steps:

- Authenticating buyer and identifying vendor;
- Allocating accounts;
- Verifying money;
- Initiating universal payment;
- Transferring universal payment; and
- Storing payment information.

Shown in Figure 7.1, the above protocol functions commence between interaction 3 and 4 inclusively. The implementation of other interactions, which are less important to the proposed UMS, will be excluded from discussion within this section. Figure 7.2 illustrates the interaction flow between participating MPSs and the UMS. The index number in arrows denotes the interaction sequence.

![Figure 7.2 Interaction flow between MPSs and UMS](image)

Note again, in the case of the protocol implementation, the Internet is the transferring channel for data exchange among the multiple systems. The detailed implementation of each protocol function will be described in the following sub-sections.
7.3.1 Authenticating buyer and identifying vendor

The buyer’s authentication is performed by the MPS-buyer rather than in the UMS, so that the UMS does not need to keep buyer’s account details. The purpose of this design is explained in Section 6.3.1. At this stage, Shaun’s user name and password are successfully authenticated, because he has already signed up with Webcoin and has sufficient credit in his account to initiate the payment.

For vendor’s identification, the UMS determines whether the NewsOne.net is known by the NetPay system. This task is executed at the UMP of the protocol layer. As designed in Section 6.3.2, the UMS requests NetPay to check the existence of vendor’s account. At this stage, the UMS sent NetPay the VendorID – N209. This identifier is recognised by NetPay because the vendor’s account already exists within NetPay. After that, a positive response is sent back to the UMS, which then authorises Webcoin to continue the next interaction; if the vendor’s account does not exist, then further transactions discontinue. This process is presented as a part of the Uni-Verif-Req and Uni-Verif-Resp interactions. In fact, this function is a joint process of allocating accounts that will be discussed next.

It is noted that the interaction between the UMS and NetPay could be also implemented via the HTTP web protocol, on which the request and response messages are exchanged.
Actually, the use of HTTPS is not necessary, because there are no security concerns for this identification.

7.3.2 Allocating accounts

As subscribing to use the Webcoin system, the buyer – Shaun is provided and forwarded the IP address of the UMS payment portal (132.234.119.11) when he is advised to use the UMS portal to pay the vendor - *NewsOne.net*, which has registered with the NetPay system other than Webcoin. This address has also been known by Webcoin and used when continuous universal transactions occur.

Webcoin sends the UMS an HTTPS request message *Uni-Verif-Req* that contains product and vendor information sent from NetPay to *Shaun* prior to his payment. Such information involves the local parameters created by NetPay, such as the vendor_ID, the value of the product, the item_ID and name of the MPS that processed this payment. The detailed value of each parameter is shown in Figure 7.3. The type of this message implies that the UMS needs to send back an HTTPS response message *Uni-Verif-Resp* (this response will be further described in Section 7.3.4). Meanwhile, the UMS has already recognized and stored the IP-address of the Webcoin payment portal. It is noted that both *Uni-Verif-Req* and *Uni-Verif-Resp* interactions are performed under the UMS protocol layer through the UMP that is connecting with Webcoin and NetPay. Figure 7.3 generally diagrams the HTTPs request and response messages with inclusive
Now the UMS is responsible for determining the payer and payee accounts for both the Webcoin and NetPay systems. As Shaun is a registered user of the Webcoin system, Webcoin could simply verify the system payer for Shaun’s account (the debtor account of MPS-buyer) by using Shaun’s username and password provided at system login. At this stage, the UMS needs to allocate the payee of Webcoin (the creditor account of MPS-buyer) for this payment. For this, the UMS looks up the system identifier in the MPS table of the UMS database (example given in Table 7.1) by executing an SQL look up command. The payer of NetPay (the debtor account of MPS-vendor) is also looked up in this database table; whilst the payee of NetPay (the creditor account of MPS-vendor) is already determined during the function of identifying vendor (Section 7.3.1) is performed by the UMS. The account identifiers for both systems are as follows:
Table 7.1 MPS table in UMS database

<table>
<thead>
<tr>
<th>MPS</th>
<th>System ID</th>
<th>Account ID</th>
<th>Min</th>
<th>Max</th>
<th>Default currency</th>
<th>Other currency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Webcoin</td>
<td>1019wc</td>
<td>um10A - 1019wc</td>
<td>0.01</td>
<td>10</td>
<td>AU$</td>
<td>n/a</td>
</tr>
<tr>
<td>NetPay</td>
<td>NP528</td>
<td>um13C - NP528</td>
<td>0.10</td>
<td>20</td>
<td>AU$</td>
<td>n/a</td>
</tr>
</tbody>
</table>

For the Webcoin system, the determined identifiers of payer and payee are:

- \( Payer_{ID} \): \(<Shaun, ae86guf#pwd : 1019wc>\), which is composed of the \( Buyer_{ID} \) (a set of username and password of the system login) and the MPS \( ID \) of Webcoin defined by the UMS.
- \( Payee_{ID} <N209 : NPS101NOAB1D03272005V17 : QKcq3Aas5R3yG4M2p6ZFYgf86Y94M2gK6f7N3f>\), which is comprised of three parts: the identifier of NewsOne.net (\( Vendor_{ID} \)), the certificate issued by NetPay and the vendor’s digital signature, respectively. It is accepted by Webcoin as a \( Payee_{uni} \).

For the NetPay system, the determined identifiers of payer and payee are:

- \( Payer_{ID} \): \(<um13C : NP528>\), which refers to a UMS account contracted with NetPay and accepted by NetPay as a \( Payer_{uni} \).
- \( Payee_{ID} <N209 : NewsOne.net>\) which is comprised by the \( Vendor_{ID} \) and the vendor’s name.
7.3.3 Verifying money

At this stage, the UMS determines the value of \textit{uniMicro} parameter. It runs a verification algorithm (as explained in Section 6.3.4) for Webcoin and NetPay, to check whether they can support both Australian currency and the amount ($0.10). The UMS then executes an SQL command in the currency table of the UMS database to look up the minimum and maximum (\textit{value}) of supported payment amounts and the set of currencies for both systems.

It is verified that the default currency of both systems is same (Australia dollar), so currency exchange is not needed. Because ten cents is between the transferable minimum amount $0.01 and maximum $10 for Webcoin, and between $0.10 and $20 inclusive for NetPay, it is concluded that the amount of this universal micropayment is supported by both MPSs. Therefore, the amounts of money to be transferred from Webcoin is AU$0.10 and that to be received by NetPay is AU$0.10.

7.3.4 Initiating universal payment service

Before starting universal payment transaction among the systems, the UMS needs to define a universal \textit{Content}_{ID}, which is related to the \textit{product}_{ID} transferred from the \textit{NewsOne.net} to \textit{Shaun}. With the \textit{Content}_{ID}, the UMS will know which universal payment is being processed and which universal payment needs to be initiated next. So
At this step, the UMS creates a unique Content\textsubscript{id}, which are mapped from the product\textsubscript{id}, and transferred from the UMS to uni-Webcoin and then to Webcoin. Also the same process is made for NetPay. The two content identifiers generated by the UMS for both systems are as follows:

- For uni-Webcoin: db515NsAr23051999 : NP528 (a combination of the value of the original product\textsubscript{id} and NetPay system ID); and
- For uni-NetPay: db515NsAr23051999: 0053 (a combination of the value of the original product\textsubscript{id} and the UMS index number);

Now, all payment initiation data collected for transferring an universal micropayment have been successfully determined and verified. They include the Payer\textsubscript{id} and Payee\textsubscript{id} parameters for both MPSs (defined in 7.3.2), the uniMicro (defined in 7.3.3) and the Content\textsubscript{id} (defined in 7.3.4). Afterwards, the UMS sends those data to Webcoin in a type of HTTPS response message shown in Figure 7.3. This response message is sent via the Internet to the 132.234.112.19 IP address of the Webcoin system. With those messages, sequent universal payment transactions could be performed.

7.3.5 Transferring universal payment - from MPS-buyer to UMS

After the Webcoin system receives the payment initiation data from the UMS, uni-Webcoin performs the transaction of a universal micropayment with the following
parameters shown in Table 7.2. The initiation of the universal transaction is accepted by uni-Webcoin, because all the following conditions are satisfied:

- *Shaun* is authorized to use the Webcoin system and successfully accesses to his account within Webcoin;
- *NewsOne.net* account is verified and recognized by NetPay;
- Both Webcoin and NetPay could transfer 10 cents and both support Australian currency for micropayment; and
- *Shaun* has sufficient credit in his account for this payment.

<table>
<thead>
<tr>
<th>Table 7.2 Parameters from Webcoin to uni-Webcoin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>PayerID</td>
</tr>
<tr>
<td>PayeeID</td>
</tr>
<tr>
<td>ContentID</td>
</tr>
<tr>
<td>uniMicro</td>
</tr>
</tbody>
</table>

After processing this transaction, uni-Webcoin delivers a *Uni-Trans-Resp* message to the UMS through the UMP, containing the following parameter values shown in Table 7.3.

<table>
<thead>
<tr>
<th>Table 7.3 Parameters from uni-Webcoin to UMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>ContentID</td>
</tr>
<tr>
<td>uniMicro</td>
</tr>
<tr>
<td>uniTransID</td>
</tr>
</tbody>
</table>
Finally, to complete the processing of the universal transaction, *uni-Webcoin* generates the unique *uniTransID* <C09980-51500 : 1019wc>, and then transfers to the UMS.

### 7.3.6 Transferring universal payment - from UMS to MPS-vendor

After receiving the *Uni-Trans-Resp* message from Webcoin, the UMS initiates the subsequent universal transaction for *uni-NetPay* through the UMP with the following parameters shown in Table 7.4. The initiation of this transaction is accepted by *uni-NetPay*, because all the following conditions are satisfied:

- The UMS account exists within NetPay, and the transaction is authorized;
- The *NewsOne.net* account is recognized by NetPay; and
- Both the currency and the amount of money are supported by NetPay;

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PayerID</td>
<td>um13C : NP528</td>
</tr>
<tr>
<td>PayeeID</td>
<td>N209 : NewsOne.net</td>
</tr>
<tr>
<td>ContentID</td>
<td>db515NsAr23051999 : 0053</td>
</tr>
<tr>
<td>uniMicro</td>
<td>0.10 : AU</td>
</tr>
</tbody>
</table>

\After processing the *Uni-Trans-Req* interaction, *uni-NetPay* delivers a *Uni-Trans-Resp* message to the UMS with the following parameters shown in Table 7.5.
Finally, to complete the processing of the universal payment, *uni-NetPay* generates the unique *uniTrans ID* "V160330N209, NP528", and then transfers to the UMS. At the final stage, NetPay confirms the completion of the payment to *Newsone.net* using the first part value of *uniTrans ID*.

The universal payment interactions made by the UMS from Webcoin to NetPay are via the UMP, which interconnects Webcoin and NetPay to enable the data exchange cross multiple systems. Figure 7.4 diagrams those interactions with associated messages and parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content ID</td>
<td>db515NsAr23051999 : 0053</td>
</tr>
<tr>
<td>uniMicro</td>
<td>0.10 : AU</td>
</tr>
<tr>
<td>uniTrans ID</td>
<td>V160330N209 - NP528</td>
</tr>
</tbody>
</table>
Figure 7.4 Universal micropayment interactions from Webcoin to NetPay via UMP

7.3.7 Storing payment information

After the completion of each universal transaction, the UMS stores the payment information in the Universal Payments table (described in Section 6.3.7) for the purpose of financial auditing and cross-system payment tracing. Not all messages need to be recorded. This information, such as Shaun’s account identifier and confirmation message to Shaun, are usually recorded and maintained by the local system – Webcoin, and are less important to the UMS. Table 7.6 lists the processed payment data. The SQL commands of the UMS server are running for the data storage as follows:
• The confirmation information received from *uni-Webcoin*; and

• The payment initiation information provided to *uni-NetPay*.

### Table 7.6 The processed payment data recorded by UMS database

<table>
<thead>
<tr>
<th>from Webcoin</th>
<th>Payer&lt;sub&gt;ID&lt;/sub&gt;</th>
<th>Content&lt;sub&gt;ID&lt;/sub&gt;</th>
<th>uniMicro</th>
<th>uniTrans&lt;sub&gt;ID&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>um10A - 1019wc</td>
<td>db515N53Ar23051999 : NP528</td>
<td>AU : 0.10</td>
<td>C09980-51500 : 1019wc</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>from NetPay</th>
<th>Payer&lt;sub&gt;ID&lt;/sub&gt;</th>
<th>Payee&lt;sub&gt;ID&lt;/sub&gt;</th>
<th>uniMicro</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>um13C - NP528</td>
<td>N209 : NewsOne.net</td>
<td>AU : 0.10</td>
</tr>
</tbody>
</table>

#### 7.3.8 Summary

Section 7.3 demonstrated the implementation of a universal payment transaction under the proposed UMS protocol cross multiple MPSs – Webcoin and NetPay. Because only common Internet techniques, such as HTTP and HTTPS protocols for secure communication are used for the current commercial micropayment transaction and the UMS universal transaction, it’s proved by the above case study that the implementation of the UMS is achievable.

Figure 7.5 summarises the behaviours of the UMS protocol, in terms of an executed universal payment service and related functions. These protocol functions occur between the interactions step 3 and 4 of Figure 7.1. The index number of function indicates the occurrence sequence and the direction of the arrows indicates the direction of the message flow. The internal actions of the various components are not shown in
the following figure.

![Diagram of universal micropayment protocol](image)

**Figure 7.5 Function sequence of universal micropayment protocol**

### 7.4 Implementation of system mapping

Implementing the system mapping is the second main task to realize the UMS. Successful system mapping enables the processing of cross-system micropayments in a smooth and straightforward way. Therefore, it’s important to the overall implementation of the UMS. Webcoin and NetPay are incorporating with the UMS by an integration of application components (*uni-Webcoin* and *uni-NetPay*) into both systems. They are also responsible to perform system mapping between the MPSs and the UMS.
7.4.1 Mapping Webcoin

The *uni-Webcoin* component imbedded in the Webcoin system interconnects with the UMP to provide the universal payment transactions. In the above case, *uni-Webcoin* interacts with the Webcoin system and the UMS to make cross-system micropayments and receive confirmations of processed payments, respectively. It also performs the system parameter mapping for universal payment services. Figure 7.6 shows the parameter mapping within the Webcoin system.

![Parameter mapping diagram](image)

**Figure 7.6 Parameters mapped in Webcoin**

Two boxes A and B represent the mapping processes that occur between *uni-Webcoin* and Webcoin, and the direction of the arrows represents the mapping flow. The mapped parameters are the result of the *Uni-Pay-Req* and *Uni-Trans-Resp* interactions within Webcoin. The right directed arrow indicates the *Uni-Pay-Req* that is generated and provided from *uni-Webcoin* to Webcoin. It refers to the initiation of a universal
payment from the UMS. The left directed arrow indicates the *Uni-Trans-Resp* that occurs between Webcoin and *uni-Webcoin*; i.e. it refers to the confirmation of a processed payment to the UMS.

### A. To map the parameters for payment initiation

When *uni-Webcoin* receives the parameters of the *Uni-Pay-Req*, it will perform the system mapping for Webcoin; four primary parameters of the payment request are mapped onto the parameters for Webcoin as the following processes:

- The *Payer* parameter is decomposed and assigned to the Username (*custID*) and Password (*custPwd*) parameters for the access to Webcoin account;
- The *Payee* parameter is decomposed and assigned to the shop ID (*shopID*), Certificate ID (*certfID*) and Digital Signature (*digSign*) parameters required by Webcoin;
- The *Content* parameter is assigned to the Product ID (*prodID*) parameter; and
- The *uniMicro* parameters are combined together with “$” marker in front and assigned to the product value (*prodVal*) parameter.

In addition, *uni-Webcoin* generates the Initiation Time (*iniTime*) and lag time (*lagTime*) which are required by original Webcoin system. Then *uni-Webcoin* parameter assigns the current date and time and 30 minutes, to parameter *transTime* and *lagTime*, respectively, so that the universal payment is required to be completed within 30
minutes after initiation. Meanwhile, uni-Webcoin also passes the parameters that were sent by the shop prior to the payment and assigns them to the shopName and shopSite parameters for Webcoin, referring to the shop name and shop site, respectively.

Furthermore, some parameters that are less important to the UMS payment but specific to the Webcoin system, such as Product Description (prodDesp) parameter, and returned URL (rtnURL) parameter which describes the location where the customers will be re-directed if their payment is successful or failed, will be stored and assigned to Webcoin, even if sometime they contain null values. Table 7.7 summarises the parameters and their values that uni-Webcoin maps and provides to Webcoin after a successful payment initiation.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter value</th>
<th>Parameter description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custID</td>
<td><a href="mailto:Shaun01@Webcoin.net.au">Shaun01@Webcoin.net.au</a></td>
<td>customer ID located in Webcoin MPS</td>
</tr>
<tr>
<td>custPwd</td>
<td>ae86guf</td>
<td>customer password to Webcoin payment</td>
</tr>
<tr>
<td>prodID</td>
<td>db515NsAr23051999</td>
<td>product ID issued and sent to customer by shop</td>
</tr>
<tr>
<td>prodDesp</td>
<td>News archive: history news access</td>
<td>product type: news archive data retrieve from shop</td>
</tr>
<tr>
<td>prodVal</td>
<td>$0.10</td>
<td>value of news archive</td>
</tr>
<tr>
<td>shopID</td>
<td>N209</td>
<td>shop ID issued and sent to customer by shop</td>
</tr>
<tr>
<td>shopName</td>
<td>NewsOne</td>
<td>name of the shop site</td>
</tr>
<tr>
<td>certfID</td>
<td>NPS101NOAB1D03272005V17</td>
<td>shop certificate issued by NetPay</td>
</tr>
<tr>
<td>digSign</td>
<td>QKcq3Aas5R3yG4M2p6zFYgf86Y94M2gK6f7N3f</td>
<td>digital signature signed by shop</td>
</tr>
<tr>
<td>shopSite</td>
<td><a href="http://www.newsone.net">www.newsone.net</a></td>
<td>address of shop site</td>
</tr>
<tr>
<td>rtnURL</td>
<td><a href="http://www.newsone.net/transinfo.jsp?res=1">www.newsone.net/transinfo.jsp?res=1</a></td>
<td>returned URL to customer</td>
</tr>
<tr>
<td>iniTime</td>
<td>1830T-29062007D</td>
<td>time and date</td>
</tr>
<tr>
<td>lagTime</td>
<td>30 mins</td>
<td>waiting time for a payment</td>
</tr>
</tbody>
</table>
B. To map the parameters for payment confirmation

After Webcoin performs the interaction of payment confirmation, the parameters related to the confirmation of the processed payment are generated by Webcoin and sent to uni-Webcoin, which will then perform the primary parameter mapping as the following processes:

- The prodID parameter along with the NetPay system ID issued by the UMS is assigned to the Content ID parameter;
- The prodVal parameter is decomposed and assigned to uniMicro parameters parameter with the “$” mark eliminated and “AU” currency sign added after the amount; and
- The payment reference (payRef) and trace number (traceNo) parameters combined with the Webcoin system ID are assigned to the Universal Transaction ID (uniTrans ID) parameter.

Table 7.8 summarises the payment confirmation parameters that uni-Webcoin receives from Webcoin, in which other parameters from Webcoin, such as the shopID and system version (sysVers) parameters, are not relevant to the universal payment, are ignored by uni-Webcoin. And if the value of errNo parameter differs from true, neither mapping nor confirmation of the universal payment will be performed. After the mapping, uni-Webcoin then confirms the universal payment to the UMS by sending the UMS the payment confirmation message containing those parameters (see tables in
Section 7.3.5). At this stage, the processing of the universal payment at customer system side completes.

**Table 7.8 Parameters for payment confirmation of Webcoin**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter value</th>
<th>Parameter description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prodID</td>
<td>db515NsAr23051999</td>
<td>product identifier issued and sent to customer by shop</td>
</tr>
<tr>
<td>prodVal</td>
<td>$0.10</td>
<td>value of news archive</td>
</tr>
<tr>
<td>payRef</td>
<td>C09980</td>
<td>payment reference no. created by Webcoin</td>
</tr>
<tr>
<td>traceNo</td>
<td>51500</td>
<td>product trace number created to users by Webcoin</td>
</tr>
<tr>
<td>transTime</td>
<td>1833T-29062007D</td>
<td>time and date of the transaction completion</td>
</tr>
<tr>
<td>errNo</td>
<td>true</td>
<td>no transaction errors found</td>
</tr>
<tr>
<td>shopID</td>
<td>N209</td>
<td>shop identifier issued and sent to customer by shop</td>
</tr>
<tr>
<td>sysVers</td>
<td>Webcoin 1.3v</td>
<td>version of MPS current used</td>
</tr>
</tbody>
</table>

7.4.2 Mapping NetPay

The *uni-NetPay* component imbedded in the NetPay system interconnects with the UMP to provide the universal payment transactions. In the above case, *uni-NetPay* interacts with the UMS and the NetPay system to receive cross-system micropayments and generate confirmations of processed payments, respectively. It also performs the system parameter mapping for universal payment services. Figure 7.7 shows the parameter mapping within the NetPay system.
Two boxes C and D represent the mapping processes that occur between \textit{uni-NetPay} and NetPay, and the direction of the arrows represents the mapping flow. The mapped parameters are the result of the \textit{Uni-Pay-Req} and \textit{Uni-Trans-Resp} interactions within NetPay. The right directed arrow indicates the \textit{Uni-Pay-Req} that is generated and provided from \textit{uni-NetPay} to NetPay. It refers to the initiation of a universal payment from the UMS. The left directed arrow indicates the \textit{Uni-Trans-Resp} that occurs between NetPay and \textit{uni-NetPay}. It refers to the confirmation of a processed payment to the UMS.

**C. To map the parameters for payment initiation from UMS to \textit{uni-NetPay}**

When \textit{uni-NetPay} receives the parameters of the \textit{Uni-Pay-Req}, it will perform the system mapping for NetPay, four primary parameters of the payment request are mapped onto the parameters for NetPay as the following processes:
• The $Payer_{ID}$ parameter is decomposed and assigned to the user ID ($user\_ID$) and access code ($access\_code$) parameters for the access to the UMS account at NetPay;

• The $Payee_{ID}$ parameter is decomposed and assigned to the vendor ID ($vendor\_ID$) parameter required by NetPay;

• The $Content_{ID}$ parameter is assigned to the item ID ($item\_ID$) parameter; and

• The $uniMicro$ parameter is decomposed and assigned to the payment value ($value$) and currency code ($curr\_code$) parameters.

In addition, some parameters which are less important to the UMS payment but specific to the NetPay system, such as the Item Name ($item\_name$) parameter, which is a short description of the product, the Session Time ($session\_time$) parameter so that the paid item is available for downloading till session expires, and the Successful URL ($succ\_url$) and Failed URL ($fail\_url$), which describes the two locations where the customer will be redirected if the payment is successful or failed, respectively.

Table 7.9 summarises the parameters and their values that $uni$-$NetPay$ maps and provides to NetPay after a successful payment initiation.
Table 7.9 Parameters for payment initiation of NetPay

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter value</th>
<th>Parameter description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_ID</td>
<td>um13C</td>
<td>user ID located in NetPay, this value referring to the UMS account in NetPay</td>
</tr>
<tr>
<td>access_code</td>
<td>NP528</td>
<td>access code to NetPay payment, this value referring to NetPay ID</td>
</tr>
<tr>
<td>item_ID</td>
<td>db515NsAr23051999</td>
<td>item identifier issued by vendor</td>
</tr>
<tr>
<td>value</td>
<td>0.10</td>
<td>Item value</td>
</tr>
<tr>
<td>curr_code</td>
<td>Au$</td>
<td>the currency of the item value</td>
</tr>
<tr>
<td>vendor_ID</td>
<td>N209</td>
<td>vendor identifier issued by vendor’s MPS</td>
</tr>
<tr>
<td>vendor_name</td>
<td>NewsOne.net</td>
<td>name of the vendor</td>
</tr>
<tr>
<td>succ_url</td>
<td><a href="http://www.newsone.net/">www.newsone.net/</a> transinfo.jsp?res=1</td>
<td>returned successful URL to customer</td>
</tr>
<tr>
<td>fail_url</td>
<td>n/a</td>
<td>returned failure URL to customer</td>
</tr>
<tr>
<td>session_time</td>
<td>120</td>
<td>minutes available for item download</td>
</tr>
</tbody>
</table>

D. To map the parameters for payment confirmation from uni-NetPay

After NetPay performs the interaction of payment confirmation, the parameters related to the confirmation of the processed payment are generated by NetPay and sent to uni-NetPay, which will then perform the primary parameter mapping as the following processes:

- The paid item (paid_item) parameter is assigned to the ContentID parameter;
- The item total value (item_total) parameter is decomposed into values “0.10” and “AU” assigned to the uniMicro parameter; and
- The approval code (app_code) parameter combined with the NetPay ID are assigned to the Universal Transaction ID (uniTransID) parameter.
Table 7.10 summarises the payment confirmation parameters that uni-NetPay receives from NetPay, in which other parameters from NetPay such as the item description (item_desp) and VendorID parameters are not relevant to the universal payment, are ignored by uni-NetPay. After the mapping, uni-NetPay then confirms the universal payment to the UMS by sending the UMS the payment confirmation message containing those parameters (see tables in Section 7.3.6). At this stage, the processing of the universal payment at vendor system side completes.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter value</th>
<th>Parameter description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paid_item</td>
<td>db515NsAr23051999</td>
<td>product identifier issued and sent to customer by shop</td>
</tr>
<tr>
<td>item_desp</td>
<td>News archive:</td>
<td>item type: news archive data retrieve from NewsOne</td>
</tr>
<tr>
<td></td>
<td>history news access</td>
<td></td>
</tr>
<tr>
<td>item_total</td>
<td>AU$0.10</td>
<td>value of item, currency and amount</td>
</tr>
<tr>
<td>app_code</td>
<td>003105_2007629</td>
<td>payment approval code created by NetPay and sent to NewsOne</td>
</tr>
<tr>
<td>vendor_ID</td>
<td>N209</td>
<td>vendor identifier issued by vendor’s MPS</td>
</tr>
</tbody>
</table>

7.4.3 Summary

Section 7.4 demonstrated the implementation of system parameter mapping for bridging the interaction gaps between the Webcoin and NetPay systems. The UMP performs the mapping task under the proposed UMS protocol. The processes from A to D demonstrate the system mapping sequence for the major UMS interactions. Those processes occur on the interactions between the built-in UMP components and
integrated MPSs. From the successful result of the mapped parameters, it’s proved by
the above case study that the implementation of mapping systems is achievable.

7.5 Measurement of UMS

To measure the performance of the proposed UMS, we use the criteria derived from the
requirements defined in Chapter 4. The following sub-sections discuss whether the
UMS design meets those requirements.

7.5.1 Unifying universal payment platform

This is actually a primary requirement (defined in Section 4.1.1) from both the
customer and merchant sides. First, the current micropayment services comply with the
UMS protocol, and then turn into a part of the universal micropayment service, such
that the UMS is able to support most of the “pay-per-click” micro-transaction models
used by current MPSs so far. Second, the proposed UMS incorporates and integrates
current MPSs to allow current customers and merchants to continue to use the MPSs of
their choice to pay or be paid, regardless of the MPSs used by the other party. Such
system cooperation and service integration are achieved by the use of specific system
components built-in and system-independent interface. As a consequence, existing
micropayments are unified and a universal payment platform is formed.
In addition, the proposed UMS supports the conversion of multi-currencies defaulted by different MPSs, such that global micropayments are made possible, and also new developed systems could be easily incorporated with it. Furthermore, the UMS enables the transfer of small value payments online, because it inherits and works with the primary payment functions of the MPSs that support micro-transactions.

As a consequence, this requirement is satisfied, current users are able to make a universal micropayment through their existing systems that have incorporated and integrated with the UMS.

7.5.2 Providing a user-friendly system

The requirement (defined in Section 4.1.2) is one the top concerns throughout the entire design of the UMS architecture. The proposed UMS provides a unified payment platform that resembles the current interfaces of the existing MPSs and extends the cross-system payment service. It means the extended payment portal is only provided to a buyer when a universal payment interaction is going to occur between two different MPSs. Actually, the UMS component is a lightweight, system-specific and system-independent software application that will be built into the MPSs, so that it would be automatically linked to or redirected by the portal interfaces of existing MPSs. Meanwhile, the buyer would be advised that cross-system payments will be processed. In addition, the UMS payment portal inherits the authentication method and
process of existing MPSs, such that users are able to keep using their original username and password at the universal payment portal to access their accounts retained in their MPSs. Then the rest of the interactions are totally hidden from the users and are performed by the UMS without any user-involved interaction.

As a consequence, this requirement is satisfied because the user’s payment habits and experiences do not really change, and the user friendliness and convenience of the UMS are comparable with that of their MPSs. Furthermore, for the new users, they just need to sign up with one MPS (already integrated by the UMS) of their choice, then they are able to use the UMS payment portal to make universal payments to the vendors who use other MPSs that have been integrated with the UMS.

7.5.3 Providing a secure system

The security requirement (defined in Section 4.1.3) is always regarded as a critical constraint for e-payment measurement. As the UMS is a web-based system requiring the payment interaction via an open network, it is likely to attract attacks from malicious users in order to commit fraud, forge e-money or mishandle the system, especially those fraud attempts on the UMS, as so-called "man-in-middle" attacks. Therefore the UMS needs to provide sufficient security mechanism to prevent and detect such attacks.
The above mentioned potential attacks may occur during the interactions between universal payment initiation and confirmation at local MPSs, and interactions between the UMS and the participating MPSs. The implementation process has already demonstrated the security methods, listed as follows, to meet this requirement.

- To prevent or detect the attacks in the former interactions, current MPSs are using Secure Socket Layer (SSL) or the Transport Layer Security (TLS) to secure the communication channel, because they are able to detect any modification during their transmission. The SSL, for instance, provides the authentication of the end points and communication privacy over the Internet using cryptographic algorithms like RSA, DES, MD5 and SHA, and prevents midway intercepting, altering or forgery of payment information. Because the authentication processing is also performed at the MPS side rather than at the UMS side, the users will always know that the UMS is really the party it claims to be. Furthermore, the fraud attempt is difficult to realize since the account balances are stored in a local MPSs under heavy security mechanisms.

- To prevent or detect the attacks in the latter interactions, the case study showed that the secure HTTP communication protocol over the earlier mentioned SSL or TLS security protocols were applied. Actually, this common solution is also used by the existing cases without the involvement of the UMS in order to authenticate their users and secure online payment transactions. Therefore, it is considered that the security of the UMS is similar to the security mechanism
used by the existing systems, which are resistant to malicious attacks. This means that the UMS does not represent a higher security risk than existing MPSs.

The case studies showed that commonly used security mechanisms can be used for the UMS. The security of the UMS is comparable with that of existing MPSs, because the functionalities of the built-in universal portal components are quite similar to the existing ones; the implementation would not create a higher security risk and the chance of being attacked will not be higher.

As a consequence, this requirement is satisfied because all major security issues, such as confidentiality, authentication, data integrity and non-repudiation have been addressed throughout the UMS payment service and protocol designs (see Section 5.1.1, 5.1.2 and 6.3 for detailed discussions). The case studies has also demonstrated that the potential attacks could be prevented and detected by the use of the common security techniques. Since the main payment interactions are relying on the existing security mechanisms of the participating MPSs, the security threats on the UMS are considerably less than that on existing MPSs.
7.5.4 Providing a trustworthy system

The trust requirements are relevant to both the UMS users and the Micropayment System Brokers (MSBs). As described in Section 4.1.4 and 4.2.4, the trust requirements have different meanings to users and MSBs.

The users’ trust in a new system could be very hard to achieve. Instead of convincing users to switch from their trusty payment systems to a new system, we use a new solution to enforce the trustworthiness for the proposed system. The approach is to inherit the users' trusts in the existing MPSs they have used so far, and provide more usability and control on the UMS so as to enhance the system trustworthiness. The case studies also realize this trust reinforcement through shifting trust in the new system from the user side to the MSB side. The case studies demonstrate that the system users are able to continue using the existing MPSs of their choice, with the same authentication method and payment experience that they are familiar with. Users’ trust, therefore, is extended. This simple deduction is based on the assumption that system users already trust MPSs they prefer and use so far. If this assumption comes into existence, the trust between users and the UMS becomes a non-issue because the component of the UMS was integrated with the MPSs they are currently trusting and using. Furthermore, besides the inheritance of the existing user’s trust in the MPS, the trustiness of the UMB, from user’s point of view, is also strengthened by providing simple and easy-of-use control mechanism (analysed in Section 7.5.2), and using
proper security techniques (analysed in Section 7.5.3), as well as protecting the user’s privacy and providing anonymity (further analysed in Section 7.5.5).

For MPSs, however, the UMS has transaction agreements and financial contracts with the existing MPSs in order to perform service integration, so that the UMS can be supervised under such contracts by the MSBs of these participating MPSs. The possibility that the UMS commits seems quite rare because all attempts are detectable and traceable. The probability of fraud attempts from the UMS is far lower than that from the user’s side; this fact was proven in Foley and Quillinan’s study (2002). If the existing MPSs agree with the system cooperation and integration, it is assumed that trust between the UMS and MPSs has already been built.

As a consequence, this requirement is satisfied because users will remain and extend their trust in their MPSs to the integrated UMS with equal degree, and the solution to enforce trust between the MPS and the UMS is also achieved by the pre-settled contracts and transaction agreements.

7.5.5 Protecting privacy and user’s anonymity

This requirement (defined in Section 4.1.5) is specific to the buyers in the current practice. The case studies show that with the introduction of the UMS, the protection of buyer’s privacy and anonymity of buyers has not been down graded. First, the
authentication process was processed at their original MPSs; second, the buyers were not revealed to the UMS and vendors during their purchases, though their accounts remained at their MPSs and were disclosed at the integrated payment portal, but payer information was not transferred and stored by the UMS after the purchases are confirmed. In this way, buyers remained fully anonymous to both the UMS and the vendors. Also, the MPSs don’t need to provide their buyer information even if a payment trace request for universal transaction occurs.

As a consequence, this requirement is satisfied. This conclusion has been drawn from the case study that when the buyer – Shaun accessed his account at Webcoin and then initiated a cross-system payment to NewsOne.net, he remained fully anonymous to the UMS, and the purchasing data was also not disclosed to NetPay and vendor – NewsOne.net, as a result his privacy was fully protected.

7.5.6 Keeping maximal functions unchanged and minimal system modifications

The requirements were defined in Section 4.2.1 and 4.2.2. As demonstrated in the case studies, the existing services of incorporated Webcoin and NetPay systems were functioning as usual with the integration of the UMS (such as user login process, confirmation generation, user account crediting and debiting, etc). Their local payment services complied with the UMS services by building the UMP component on top of Webcoin and NetPay. This modification could be acceptable by most of other MPSs
because it is considerably smaller compared with the system modifications for directly integrating services of two different systems (such integrations result in large amount of effort time in the definition of system mapping rules, which was mentioned in Section 5.1.2).

As a consequence, this requirement is satisfied. The case study has also demonstrated and proved that the proposed approaches for universal micropayment interactions (both between the UMS and its users, and between the UMS and multiple MPSs) are practical and feasible.

7.5.7 Enhancing compatibility, scalability and efficiency

The requirements were defined in Section 4.2.3 and relevant to MSBs. The case studies demonstrate that the proposed UMS is compatible with the existing MPSs in terms of user authentication methods, security mechanism, transaction interactions and billing methods, etc. Because of applying the suitable interaction approaches and use of integrated payment portal, the UMS is able to scale with regard to a growing user base and increasing payment volumes.

Regarding the system efficiency, the UMS adopted the optimized payment solution, which happens when both buyer and vendor are using the same MPSs to perform transactions. In that case, both accounts remain in one system, and then the use of UMP
for processing of a universal transaction would be unnecessary. The judgement of involving the UMS or not will be made at the buyer’s MPS when the buyer initiates a payment. Because the vendor’s MPS ID or system name will be transferred to the built-in portal at the buyer’s MPS along with the product and vendor information, then it could be identified if the vendor is using the same MPS. If the case is true, the portal would not perform the further protocol tasks such as identifying vendor’s account at vendor’s MPS, generating a unique Content ID, or converting currency, etc. Such that no extra interactions would be added into the original payment service performed by their MPSs, the transaction efficiency is optimized in this manner. It is noted that those further protocol tasks do not create efficiency problems as the UMS inherits these characteristics from the existing MPS services.

As a consequence, this requirement is satisfied because the proposed UMS provides a high degree of compatibility, scalability as well as optimizing efficiency.

7.5.8 Summary

According to the results from the demonstration of the case studies and the analysis of requirement fulfilment, it is concluded that the requirements from both the users and the MSB sides are satisfied, the proposed interaction approaches and the UMS architecture and its protocol are economically and practically implementable.
7.6 Conclusion of case studies

The case studies selected for the simulation of the system implementation are theoretically based at this stage due to the time restraint and operational limitation of commercialized systems in use. Although other system components implemented have not been shown in this Chapter, the implementation of universal payment protocol and system mapping are successfully carried out at a theoretical level. Concluded from the presented case studies, it has been proved that the practical implementation of the universal micropayment across multiple MPSs is realistically achievable.

7.6.1 Case studies’ demonstrations

The two case studies demonstrated an implementation of the universal payment protocol and system mapping for cross-system transactions. The universal payment transaction from customer to vendor via different MPSs is achieved through the system interconnections under the proposed payment protocol. The universal transaction is also realized by accurate system parameter mapping from the payer side to the payee side of the MPSs, and from the payment initiation to payment confirmation. Indeed, the mapping occurs between the existing MPSs and the integrated UMS applications, rather than directly mapping two systems. The inbuilt UMS applications and cross-system mapping is working at the back of the existing systems without modifying their original functionalities.
By the current micropayment experience, the subsequent transaction needs to be processed from the same customer to vendor during a short period. It could be achieved in the implementation, because it’s assumed by the design of the UMS protocol that if the initialisation information for all universal payments is determined and the first payment is completed then the second universal payment will also be completed and confirmed to the vendor. Furthermore, the confirmation data transferred from the payer system to the customer and from the payee system to the vendor are required to complete a universal payment, so that the information needed to be transferred via the existing systems is possible in both cases.

Noted again, the interactions associated with both universal payment initiation and confirmation could be implemented via the HTTPS protocol, such that the exchanged messages that carry sensitive payment information are secured, “man-in-middle” attacks are prevented or detected.

### 7.6.2 Limitation and possible solution

These case studies have shown that the functionality of Webcoin and NetPay are very similar, the only difference is the system parameters they use for payment transaction. However, the real operation process will vary from system to system, especially when dealing with cross-system mapping from a pre-paid system to a post-paid system. The
UMS involvement will require additional data exchange among participating systems, which might result in some time lag during transactions. Additionally, securing the participating MPSs and the UMS components is not a difficult task, but the internal data exchange at local MPSs is relying on the local security mechanism, which might risk a security breach in some degree.

The possible solution for the above limitation might be achieved through the following approaches. First, the time-out period used by the system components to avoid the blockage of the UMS could be pre-set so that the latency of the additional communication and processing is taken into account. Second, both the UMS and participating MPSs might need to maintain a session and store related information until their involvement to the processing of a universal payment completes. Third, the system security evaluation could be performed to check if the participating MPS meets the defined security requirements for the secured cross-system payment.
Chapter 8  Conclusions

This chapter concludes the research presented in this thesis. An overview of the study is presented first in Section 8.1, followed by the main conclusion of the study in Section 8.2. Then, Section 8.3 highlights the possible contributions of the study. Lastly, the limitations of the study are considered and a number of directions for future research are suggested.

8.1 Overview of the study

Internet commerce has been researched and developed over a decade. The continuously expanding online marketplace is now becoming well regulated and tends to be more specifically segmented and user driven. This study has been conducted under such circumstances.

The literature shows that the huge market for online content will soon emerge, especially for those low-value content, like music, video, game downloads, and is expected to grow substantially. In order to support “pay-per-use” of such type of online content, Micropayment System (MPS) is playing an important role for such market.
The research on MPS started along with the Internet booming a decade ago, when many trial systems appeared but none succeeded. The main reasons for the failure were not the limitation of technology, but user resistance as people were used to getting online content for free and also the transaction costs were high. The second wave of the MPS development arrived during recent years when the charged online content with small value becomes quite popular on the Internet. Many systems dedicated to processing micropayment transactions on the Web emerged, but no one system has yet gained wide acceptance among online merchants and consumers due to the fact that most systems are running locally with limited user base. As a result, many current research interests on MPS studies have been shifted towards social and human issues rather than technique focused.

The current micropayment situation also shows that many competing MPSs already exist on the Internet market. Most of them in operation are serving local users within regional boundaries, customers and merchants who want to perform extended online trades are forced to use multiple systems to serve all their needs. Using multiple MPSs also creates new problems and inconvenience for users, for instance, customers and merchants have to install multiple software packages and hardware devices, manage multiple accounts and e-wallets, remember multiple passwords, trust different payment system brokers and so on. Therefore, users expect a global micropayment solution where the customer and merchant could have a uniform payment platform to integrate cross-system payments.
In this research, we identified a new approach that may have a higher chance of achieving the desired situation. This approach is originally derived from the investigation of existing MPSs in terms of their characteristics and functionalities. After the common features were identified from the system reviews, we propose that the existing micropayment services could possibly be cooperated with and integrated into a universal payment service in a harmonious manner. To realize this integration, the Universal Micropayment System (UMS) was introduced and proposed. The core component of the proposed system is a Universal Micropayment Portal (UMP), a part of which is required to build on top of existing MPSs to map existing system parameters into the UMS parameters. The interactions between the MPSs and the UMS are performed under a proposed UMS protocol, such that a chain of cross-system payments could be integrated into a universal micropayment.

8.2 Conclusion of the study

The main research questions of the study were formulated in Section 1.4. In order to address those questions, a series of research tasks were designed and undertaken in a logical way during this study. The conclusions for those accomplished tasks are as follows.
8.2.1 Extensive system review and investigation

The starting point is a broad study on existing e-payment systems. The characterization model was developed, which is used to describe and classify the e-payment system types from business and functional viewpoints. Great focus was placed on the MPS type, where the previously proposed and current operational MPSs were extensively examined.

We identified that the main functional characteristics of existing MPSs are the payment initiation and confirmation interactions between systems and their users. System parameters for data exchange associated with those interactions were also discovered. A distinct finding was that payment functionalities of investigated MPSs are quite similar behind their different interfaces, though some functions have local system significance. This finding highlights the research hypothesis, that is, whether the existing payment services could be integrated into a uniformed one through incorporating their common functionalities.

8.2.2 Methodology refinement and Approach determination

To test this hypothesis, we defined and discussed two possible approaches, the market-driven approach and the service-oriented approach. We then determined the service-oriented approach one is the more suitable to achieve the desired solution.
because it has practical advantages over the other.

We also refined the design methodology by the use of Enterprise Architecture (EA) Approach and selected appropriate approaches for system integration and implementation. The system integration could be realized by mapping functional interactions and setting an agreed payment protocol on top of the existing portal of MPSs. This approach may minimize current systems’ modifications to a great extend such that it seems to be much realistic. The system implementation is realized by a case study approach.

**8.2.3 Requirement detection**

Before proposing a new system architecture and protocol, a task to identify the requirements for the UMS was undertaken. Based on the results concluded from the literature review, the requirements for the UMS were determined from three aspects: user, Micropayment System Broker (MSB), and functional requirements. From user’s (both customer and merchant) points of view, they require a trusted system that provides a secured payment service with anonymity and privacy protection. Users also expect the universal payment platform could extend their favorite system interface and keep their payment habits and experiences unchanged. From the MSBs’ points of view, they require a minimal change of their original system functions, and are also concerned with availability and scalability of the proposed UMS. Both user and MSB
requirements determined the UMS functional requirements in terms of payment imitation and confirmation, as well as the payment methods.

8.2.4 UMS architecture and protocol design

According to the requirements, we proposed the UMS architecture. For the process of a universal payment transaction, we decided suitable interaction approaches. The interactions between the UMS and its’ users were realized by standardising the payment portal that could be integrated system-independently. The interactions between the UMS and multiple MPSs were realized by standardising the system mapping procedures for uniformed payment transactions. To allow practical and efficient system mapping, a system-specific application component is required to be built on top of participating MPSs. Upon those interaction approaches, the UMS architecture was distinctly divided into access layer and transaction layer, which refer to UAL and UTL, respectively. At UAL, the UMS inherits the original services (access methods) of incorporated MPSs, but extends them into a unified layer, whilst at UTL the UMS bridges multiple MPS for a universal transaction by mapping system parameters.

The UMS protocol was then developed. Based on the main functionalities of studied MPSs (concluded in Section 8.2.1), we also determined the primary system parameters for payment initiation and confirmation. Those parameters were exchanged and
converted within the UMS under such protocol. The developed protocol functions also specify how the system cooperation and service integration are performed and how the payment transactions are processed.

8.2.5 System implementation and measurement

We used case studies to demonstrate that the implementation of the UMS and its’ UMP is feasible. The cases we selected are Webcoin and NetPay systems, then the UMP components that were built upon them are so-called uni-Webcoin and uni-NetPay, respectively, which carry out the parameter mapping during the cross system payment interactions. The case studies demonstrated how the implementation of the various components was realized. Though the experiment was limited to the theoretical level, it proves that approaches for such system cooperation and integration are practicable and achievable.

The measurement of the system implementation was performed to examine whether the defined requirements were fulfilled. The case studies demonstrated that the UMS offers a system-independent solution such that both the Webcoin and NetPay systems could be continuously functioning. Only a minimal modification was made to them, and most of their local functions remained unchanged. Additionally, with the system integrations, buyers and vendors can pay each other in a transparent way and universal payments are transferred in a harmonious manner. They also don’t need to change their
payment habits as they do not have to deal directly with the UMS. As an additional result, buyer’s privacy and anonymity were protected. The UMS inherited the existing security mechanisms used by Webcoin and NetPay, and applied common security technologies and secure channels for online interactions, such that no significant security problem was found. Because a universal micropayment processed by the UMS is actually a chain of payments processed by the existing MPSs, the trust issues were addressed. Furthermore, the UMS provides a high degree of comparability, scalability as well as optimizing efficiency. In conclusion, the requirements formulated from both user and MSB sides were satisfied.

8.3 Contributions to knowledge

This thesis describes how the proposed UMS could be incorporated with and integrated into multiple existing MPSs to make a universal cross-system transaction achievable, such that customers and merchants can always use their preferred MPS to pay each other. This research has made several contributions to academic study and industrial practice with respect to micropayment system’s modeling and development, and system integration in terms of payment functionality and transaction protocol.
The main contributions of this research can be summarized as follows.

1. The extensive review of existing electronic payment systems contributes to knowledge by offering a better understanding of e-commerce payment systems though an innovative scheme of comparison and classification based on the nature and functional characteristics of these systems.

2. The detailed investigation and comparison of currently in-use and previous operational MPSs, highlighting their main characteristics and technical specifications, not only contributes to the literature in the micropayment area, but also provides practical implications to new MPS design with respect to functionality and security mechanisms.

3. The use of Enterprise Architecture (EA) approach in this study for system design and life cycle contributes to current system modelling techniques, and also extends the scalability of EA framework to be applied in e-commerce related services.

4. The development of the UMS architecture and a generic and systematic interconnection approach for existing MPSs enables cross-system interactions. The design of the Universal Micropayment Portal (UMP) also allows a high scalability of the proposed UMS, system compatibility and global acceptability.

5. The lightweight design of the universal micropayment protocol allows commonly used security techniques for protecting data exchange at universal transactions, and minimises the threats for both users and system brokers.
without an overhead computational burden and significant time lags for cross-system payments.

6. The service-oriented design of a universal payment service, which can be provided on top of the participating MPSs, allows them to comply with this service without changing their local functionalities and user’s payment habits.

7. The implementation of the proposed UMS and its protocol demonstrates the feasibility of a universal micropayment system, such that it could become a guideline for future MPS design and a de facto standard for system integration.

8. Furthermore, the conceptual design of the UMS and its payment service has the potential to help further promote the development of e-commerce involving micropayments by means of simple, low cost, secure and efficient universal payment portal and protocol.

Some research work related to this PhD study has been published in volumes of conference proceedings, including Shao, Nguyen and Muthukkumarasamy (2004) and Shao and Nguyen (2005).

8.4 Limitations of the study

Before recommendations for any future study can be formulated, there are some limitations of the present study that need to be addressed.
The limitations of the investigation of current micropayment systems include:

- Firstly, in this study, the selection of currently available MPSs on the market was restricted by the difficulty of accessing the technical specifications of commercial payment systems in operation. The trade off made for obtaining those system details (especially those information about what security mechanisms are applied and how the transaction messages are delivered) is mainly based on their company’s whitepapers, newsletters and related articles. Some detailed system measurements that may relate to commercial secrets are limited for academic purpose. Due to the restriction of the available system specifications from their brokers, the process of literature review and examination of some MPSs could not be updated with the most recent information.

- Secondly, the fast growth of the micropayment market and rising numbers of local MPSs with new technologies and advanced security methods make the literature of this research difficult to remain up to date. In addition, more and more existing e-payment systems intend to shift to micropayments and some are now supporting micro-transaction for particular businesses, such as music downloads. This situation results the definition of micropayment systems in this research ambiguous and formless.
The limitations of the implementation of the proposed system include:

- Both payment infrastructure and system development are expensive and time consuming. The system protocol and functional implementation in this research are simulated and performed principally at the theoretical level since serious experiments could hardly be undertaken due to the limitations of time and funds.

- The proposed architecture and universal payment protocol may not be suitable for some of newly developed MPSs because they may use different billing methods, such as charging to their ISPs’ accounts or mobile bills rather than the commonly recognized e-wallet approach.

- Most MSBs are focusing on their local market; some of them may not have intention to globalize their markets, such that the persuasion of an involvement to the UMS protocol might be more like a management concern rather than a technical or trust issue.

In spite of these limitations, this study has extended research on micropayment systems with respect to a new system model, payment protocol and transaction method. The study can also be replicated in the context of system integrations and payment networks.
8.5 Directions for future research

As mentioned in the first Chapter, low-value online content and service with micro-transactions have been generating huge revenues over the past few years, and micropayments continue to expand and occupy the electronic payment marketplace with an incredible speed. Undoubtedly, it will attract continuous research in this area. This research extends the knowledge in the micropayment area with a conceptual system having been proposed. However, the limitations of this research (due mainly to the need to confine the research scale to the time frame and resources available) suggest several potential areas of interest that may become directions for future research.

These include:

- further quantitative research to collect users’ feedback after using the UMS. As explained in Chapter 3, a key contributor to the failure of first-generation micropayment systems was the lack of users’ acceptance.
- more case studies, covering additional existing MPSs (other than Webcoin and NetPay), to confirm or enhance the reliability and scalability of the UMS integration.
- the development of an UMS with multiple micropayment portals, which could incorporate most systems in a global coverage.
As the number of mobile phone users rises rapidly and as mobile networks mature, mobile micropayments are becoming an important business trend (Anyasi & Otubu 2009, Xin 2009). The unification of mobile payment services at a universal level would further expand markets for e-payments. Consequently, the incorporation of a mobile payment network by introduction of universal payment portals for the mobile world would be of great interest. Similarly, the rise of social networks, such as Twitter and Facebook, may transform the micro-transaction market in the near future through micropayment-involved activities such as micro-blogging, donations and even instant feedback for online shops (Max 2009). Shifting the research focus onto payment platforms based on such social networks would be highly appropriate.

Besides the above research directions on technical and user aspects, other subjects related to the analysis from both managerial and economic points of view could be considered, for instance:

- A study on the management issue of potential system brokers would be one subject for further research. Potential brokers could be local banks, telephone companies, ISPs, or other independent financial institutes, etc. Such a study could involve an evaluation of the payment services they provide, a comparison of their billing models, and a selection of an appropriate universal portal for an integrated micropayment service, as well as some stakeholder’s rights, law and regulation issues.
• A study on the economic analysis of systems implementation cost and investment return raises another area of research interest. Due to the lessons learned from the failures of previous micropayment approaches and developed systems, the concerns on how to balance the capital funding for a system’s sustainable development and market occupancy for being profitable become more critical than technical issues of a new system.

Furthermore, economics, sociology, and psychology related obstacles are likely to keep micropayment systems restricted to a minor economical role in e-commerce. Those obstacles against micropayments found in (Shirky 2000 and Odlyzko 2003) seem not to have drawn great attention by system developers, but it’s definitely worthwhile to raise research interests in those issues.

8.6 Chapter summary

At the time of this thesis being completed, it is still hard to predict which system will have an evolutorial break through in technique. Also it is too early to conclude that micropayment systems will have a global acceptance like credit card system (some of which have already started micropayment services). But the micropayment has already start penetrating into our daily life, not only for music or video downloads, but also parking, movie tickets, ring tones, IP phone, virtual gaming, and even donations. Given all the obstacles that micropayments face, micropayment systems may unlikely to
achieve widespread acceptance unless they overlay on top of existing services or systems which are already widely used, which has leaves many issues for researchers to probe into.
References


Barry, DK 2003, Web services and service-oriented architecture, Morgan Kaufmann publisher, San Francisco.


Centeno, C 2002a, Securing Internet payments - the potential of public key cryptography, public key infrastructure and digital signatures, Background paper no. 6 of the Electronic Payment Systems Observatory (ePSO) project, Institute for Prospective Technological Studies (IPTS), Joint Research Centre (JRC), European Commission.

Centeno, C 2002b, Building security and consumer trust in Internet payment - The potential of soft measures, Background paper no. 7 of the Electronic Payment Systems Observatory (ePSO) project, Institute for Prospective Technological Studies (IPTS), Joint Research Centre (JRC), European Commission.


