DETERMINANTS OF ASP SUCCESS

by

WINSTON JOSEPH MACKENZIE

B.Sc.d., The University College of Cape Breton, 1997

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE (BUSINESS ADMINISTRATION)

in

THE FACULTY OF GRADUATE STUDIES

(Management Information Systems Division,
Faculty of Commerce and Business Administration)

We accept this thesis as conforming
to the required standard.

THE UNIVERSITY OF BRITISH COLUMBIA

April 2001

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Commerce and Business Administration

The University of British Columbia
Vancouver, Canada

Date April 26th, 2001
Abstract

Application Service Providers (ASP) are a recent phenomenon in the sphere of information technology. Leveraging the advantages of the universally accessible and ubiquitous nature of the Internet, application service providers claim to lower the cost of ownership of applications. Generally operating from a central and remote data centre, ASPs 'rent' their applications to clients in a 'pay as you go' fashion. This has several advantages to the client besides cost savings. This thesis introduces the reader to ASPs and examines the reasons companies hire the services of an ASP; furthermore it suggests an analytic framework based upon economic concepts in an attempt to predict what business models and strategies will be successful in the ASP market and what shape that market might take. The thesis also discusses the possible effect of some emerging technologies.
# Table Of Contents

Abstract ...................................................................................................................................................... ii

Table Of Contents ........................................................................................................................................ iii

List Of Tables ................................................................................................................................................ v

List Of Figures ............................................................................................................................................... vi

Acknowledgements ...................................................................................................................................... vii

Dedication .................................................................................................................................................... viii

Chapter 1. Introduction ................................................................................................................................. 1

Chapter 2. Defining The ASP Concept ......................................................................................................... 3

2.1. What Is An ASP? .................................................................................................................................. 3

2.2. What Types Of Asps Are Out There? ................................................................................................. 6

2.3. Who Are The Players In The ASP Ecosystem? .................................................................................. 8

2.4. Business Models ................................................................................................................................ 14

   ASP Market Internal Operations ........................................................................................................ 14

   A New Stratification Of ASPs ............................................................................................................. 18

2.5. Comparing The ASP Model To In House Solutions ........................................................................ 26

2.6. Technological Problems And Advances .......................................................................................... 28

2.7. Examples Of ASPs ............................................................................................................................... 33

   ASP Market Shares ............................................................................................................................. 33

Chapter 3. The ASP Market ......................................................................................................................... 39

3.1. How Big Is The Market For Asps? ...................................................................................................... 39

   How Much Revenue Is There For Asps? ............................................................................................... 39

   Who Is Investing And How Much? ...................................................................................................... 42
List Of Tables

Table 1. Categorization By Market And Span ............................................................ 18
Table 2. Subtypes Of Niche VSPs ............................................................................. 20
Table 3. Comparison Between ASP And Traditional IT ......................................... 27
Table 4. ASPs By Type And Target Customer Size ............................................... 38
Table 5. The Changing Projections Of ASP Revenue By Major Research Corporations. 40
Table 6. Applications By Percentage Intending To Buy From An ASP ....................... 45
Table 7. Codes For ASP Type Used In ASP Framework ........................................ 54
Table 8. Summary Of Variable Interrelationships .................................................. 68
Table 9. Client Acceptance And Lock-In By Criticality And Substitutability ............ 76
Table 10. Framework Predictions ........................................................................... 80
Table 11. Factors Increasing The Likelihood Of ASP Usage .................................... 98
Table 12. Factors Increasing The Likelihood Of Not Using An ASP .......................... 99
Table 13. Summary Of Successful Strategies ......................................................... 106
List Of Figures

Figure 1. Generic ASP Model ................................................................. 8
Figure 2. Customer, Client And ASP Interaction ................................ 11
Figure 3. Microsoft ASP Service Delivery Model ............................... 13
Figure 4. Comparison Of ASP Market With And Without Aggregators ... 16
Figure 5. Market Share By Number Of Customers [SCN, 2000] ............... 34
Figure 6. Market Share By Number Of Contracts [SCN, 2000] ............... 34
Figure 7. Communications Startups Venture Funding By Industry Segment ... 43
Figure 8. Attributes Affecting The Form Of An Organization .................. 52
Figure 9. ASP Type Matrix ................................................................. 54
Figure 10. Functionalities Combined Into An Application By Client .......... 56
Figure 11. Applications Delivered To Client ........................................ 57
Figure 12. Package Delivered To Client ............................................... 58
Figure 13. Client Choice Of ASP Type As Transactional Costs Increase .... 69
Figure 14. Client Choice Of ASP Type For P(i) As Asset Specificity Dominates ... 71
Figure 15. Client Choice Of ASP Type For P(i) As Asset Specificity Increases ... 72
Figure 16. Client Choice Of ASP Type For f(i) Or A(i) As Specificity Increases ... 73
Figure 17. Client Choice Of ASP Type For P(i) As Criticality Dominates .... 77
Figure 18. Client Choice Of ASP Type For P(i) As Substitutability Increases .... 79
I would like to acknowledge Yair Wand for his many contributions in shaping this thesis. My utmost thanks to Albert Dexter, for his advice and support along the way and at the end.

Many thanks to Martin Mroz for his editing, input and moral support. My gratitude to Kalpana Solanki and Sunil Chauhan for their late night proofreading.

My utmost gratitude to Julie Nichols, Jennifer Papke and Margot Fraser for providing information, direction and their ever-cheerful dispositions.

My thanks to Simon Fraser University Library for giving me access to their considerable, and up to date, resources.

And lastly, for my parents Anita and Winston MacKenzie, for their love and support.
Dedication

To Mom and Dad, without whom I would not have started. To Livia, without whom I would not have finished.
Chapter 1. Introduction

Application Service Providers (ASPs) are a growing phenomenon in the IT world. ASPs rent access to applications over the Internet. In the past much hype about their future success and current bad times have been in the news. Investors and customers have shied away from them after watching the failure of dot coms. What are ASPs? And how will they fare in the IT marketplace? Will customers really buy their applications 'over the wire'? And most importantly, what are the determinants of ASP success?

This thesis attempts to answer these questions by examining why potential customers favour this delivery method for software. It suggests an analytic framework based upon an extensive literature review. This attempts to predict what type of ASP business model will be successful and what strategies would be most appropriate for each ASP type. The framework accomplishes this by examining how the various types of ASPs will relate to one another and to their clients.

In Chapter 2, a definition of ASPs is set forth. ASPs are briefly compared to outsourcers and are then categorized by ownership of application. The focus then shifts to the various entities interacting with ASPs. Business models are set forth and a new classification scheme based on application market versus score is put forward. The chapter continues on with a comparison of hosted services against in-house solutions. Some upcoming technological solutions to performance problems (packet labelling), new connectivity standards such as UDDI/SOAP, and advances in software development techniques are explored and how they solve some of the problems with the ASP model are shown. The chapter concludes with some examples of ASPs and their market share.
The market for ASPs is discussed in Chapter 3. Total revenue projections are stated, as are investment figures and trends. Predictions with regards to ASPs in the year 2001 and beyond are presented. Changes in customer attitudes toward software as utilities instead of a product or service are discussed.

An analytic framework for examining the determinants of an ASP’s success is created in Chapter 4. Some economic concepts such as transaction costs and asset specificity are reviewed since they are used throughout the framework that is developed. In Chapter 4 the outcome variable is the decision not to use an ASP or to use some type of ASP. Several variables are used to determine the ASP type decided upon. This comes together to from a framework of ASP success. Predictions are made as to what types of ASPs will succeed in the future and the strategies they should follow. Strategies that an ASP should follow, according to the ASP type and what the model states, in order to draw customers to their business and keep them, are discussed.

Chapter 5 puts forward the forces driving clients toward ASP usage. Also, the possible reasons that clients are not using ASPs are examined in greater detail than they are in Chapter 4. These forces are analysed in light of the framework constructed in Chapter 4, and several of the variables used in Chapter 4 are applied to determine the client’s decision to use an ASP or not. This is done in hopes of furthering the predictive abilities of the model. Furthermore, Chapter 5 suggests the strategies an ASP might follow in light of the findings of Chapters 4 and 5 to maximize the likelihood of success. Mix of services, costs and relationship to other ASPs are discussed.

Chapter 6 concludes the thesis by summarizing our findings and presenting some possible research directions and extensions to this work.
Chapter 2. Defining The ASP Concept

In this chapter the ASP concept is defined and the types of ASPs that are operating is examined. The ASP ‘ecosystem’ discussed, as well as some ASP business models. A stratification of ASPs that examines them based on a market span versus functionality span basis is proposed. A comparison of hosted services against in-house solutions is made. Some upcoming technological advances are discussed that purport to eliminate of some QoS (Quality of Service) concerns with respect to the Internet new connectivity standards that will change the ways functionality is delivered over the Internet. Furthermore, new development methods and technologies that will create applications designed to be delivered over the Internet are also examined. The chapter then concludes with some examples of ASPs and their market share.

2.1. What Is An ASP?

International Data Corporation supposedly coined the term Application Service Provider, or ASP, in 1998 in a study on the future of outsourcing [Butler 2000]. In [Portera 2000] an ASP, or a Hosted Service Provider, is defined as: "...an independent, third-party provider of software-base services that are delivered to customers across a wide-area network, typically the Internet." Furthermore, Portera differentiates the ASP model from the traditional client/server approach to software: “Unlike a traditional client/server software vendor, an ASP typically installs and hosts applications exclusively within its own data centre.” It shall be shown that the data centre may not belong to the ASP itself, but to a third party, as long as it is not at the user’s site.
Explosive growth in network capacity has eliminated the traditional argument that a hosted application would be too slow. This combined with the dropping price of bandwidth, and upcoming improvements like packet labelling, has enabled the ASP industry to overcome some of the user concerns about speed. Since bandwidth is currently being priced in a commodity-like fashion, the economies that the ASP model can realize will be translated into savings for the customer, rather than buried under connection costs.

ASPs themselves are not really a new concept. One author opines the following about ASPs:

ASP, or application service provider, is one of the more common terms used to describe the companies working to free businesses from the burden of designing, implementing, and maintaining IT systems. Applications hosting is another popular term. I prefer outsourcing; a time-honoured term for what is really a time-honoured practice, despite efforts to the contrary to make this market look all new and shiny.

[Greenbaum, 2000]

Although this opinion may seem dismissive, it is accurate with regard to the ASP companies themselves.

However, the author does miss a few basic facts. For starters, ASPs today are offering major multi-user applications whereas back in the 70s these were mostly single user applications [Slavid, 2000]. Secondly, hosted applications represent a change from thinking of software as a ‘product’ to thinking of software as a ‘service’. This fundamental change in the way consumers view software has some important ramifications for the market. Lastly, and most importantly, ASPs are offered over the Internet.

Let us consider the implications of ASPs being offered over the Internet a little further. Because the Internet is ubiquitous, and there are few computer users without Internet access, they allow universal access to ASP systems by suppliers and vendors.
Many consider ASPs as essentially an outsourcing of the application maintenance, design and provisioning functions of the MIS or IT department [Greenbaum 2000, Butler 2000]. Furthermore, the ASP model bares a remarkable resemblance to another form of outsourcing: ‘time sharing’ of computers in the early 1970s. [Slavid 2000] “In those days, computers were expensive, and the expertise required to program and maintain them was scarce” [Butler 2000]. Some of these forces are still in effect today. Computers migrated from the time-sharing host to strict outsourcing companies in the early 1980s. Many factors played a role in the movement of companies as the usage of the time-shared computer approached 24 hours per day and the price of computers started to decline, thus making time-sharing more expensive than buying and running a data centre.

Although ASPs clearly act in an outsourcing role, this thesis shall not focus on issues regarding generic outsourcing but exclusively upon issues relating to ASPs. However, our discussion of ASPs does have strong parallels to the research in the field of outsourcing. Outsourcing in general is an area of intensive research and there exists a large body of knowledge on the subject. Those who are interested in more information regarding outsourcing are directed to Winning the Outsourcing Game by Janet Butler, Strategic Outsourcing Decisions written by W.R. King, or Outsourcing: A CIO’s Perspective written by Oakie Williams. Throughout this thesis we shall highlight relevant findings in the field of outsourcing.

For the purposes of this thesis, an ASP shall be defined as: an organization that offers a contractual service to deploy, host, manage and rent access (via the Internet) to an application from a remote facility. ASPs are responsible for either directly or indirectly providing all the specific activities and expertise aimed at managing a software application or set of applications [Douglass 2000].
2.2. What Types Of Asps Are Out There?

According to TeleChoice's Scott Heinlein there are four major categories of ASPs [Heinlein, 2000]. These groupings are based mainly on application ownership or company origins and are fairly representative of the classification of ASPs that most analysts are currently using in examining this market. It is useful to examine this at this juncture because it provides the reader with an insight into the categorization of ASPs that are currently in use. Let us look at each, and discuss the strengths and weaknesses of each.

**Resellers:** Package another ASP's product and resell it to end customers for a discounted cost. They typically will bill the end customer and provide the first level of support. Many resellers are not responsible for maintaining the application from either a technical or product perspective, beyond first level of support. They do not have much control over the application but do own the customer. This model can be good for companies wanting to offer applications but not having the experience to do so. This provides them with the ability to combine their own products with the resold applications in a bundle package offering. The downsides to this model are lack of product control and often small margins. The resellers Microage and Viasoft fall in this category.

**Aggregators:** Integrate multiple applications from different ASPs and offer them to customers. The applications are usually offered through a portal with a single sign-on. Utilizing multiple ASPs allows aggregators to offer vertical packages or many horizontal packages. This limits the number of ASPs that businesses must deal with as they add more applications. The disadvantage is that the aggregator model tends to be rather complex because of the integration requirements of multiple ASPs. Examples of aggregators include Jamcracker, Agiliti, and Bluetrain.
Pure-plays: Rely least on partnerships to deliver their applications. They ideally have their own data centres where they maintain the applications themselves. This gives them a high level of control over the service and customer experience. Some pure-plays are partnering with Internet service providers. This gives them total end-to-end accountability to the customer. The disadvantage to this model is that it requires expertise in many areas. Pure-plays must know the software side as well as the data centre side. It is also the most costly of the models. However, many customers will want their ASP to have end-to-end accountability if they are running critical business applications. Examples of pure-plays are USI, Interliant, and Corio.

Business Process Outsourcers: can be either a reseller, aggregator, or pure-play. The distinction is that they deliver business process outsourcing services in addition to application services to the customer. They may provide an outsourced HR organization, PR firm, shipping company, accounting group, etc. Business process outsourcers may partner with management consulting firms for these outsourcing services as well as companies like Kinkos, Office Depot, Federal Express, and JB Hunt. Business process outsourcers can be compared to a professional services organization. The disadvantage to this model is it relies on several partnerships similar to the aggregator model and demands creativity and simplicity in packaging.

For the client of an ASP, the distinction between a Business Process Outsourcer and any of the other three ASP types would be clear. In addition to the delivery of application functionality, Business Process Outsourcers would also deliver outsourcing capacity that outsourcers have always delivered. However the distinction between resellers, aggregators and pure-plays becomes clear when changes have to be made, or are made, to the application(s) which forms the basis of the service. Resellers and aggregators have little or
no control over the application since they are passing along the services of other ASPs. Pure-plays create their own applications so they have end-to-end control as mentioned previously.

2.3. Who Are The Players In The ASP Ecosystem?

The business space surrounding ASPs is often referred to as an ecosystem. The reason for this is the level of non-linear interactions and changing roles that the various players in this market undergo [Charley, 2000]. We shall begin with a look at those various players and where they are headed within this market.

Here we see the structure of a generic ASP interacting with one of its clients:

**Figure 1. Generic ASP Model**

There are four key players involved in delivering the application service to the client in this figure. The ISV (Independent Software Vendor), ASP (Application Service Provider), ISP
(Internet Service Provider) and the Data Centre. Note that these are not necessarily distinct organizations. For USi, the ASP and Data Centre are all in one organization, for MySAP.com the ISV and ASP are all within one organization. If a telco (telco is a common short form of telecommunications company) such as AT&T were to buy USi, only two companies would be represented in the generic ASP diagram above. Not depicted in this diagram are the client’s customers. ASPs impact the customer in certain ways. Let us examine each of these players in some detail.

**Independent Software Vendors (ISVs)**

Historically, ISVs maintained a direct relationship with the users of their software. However, under the ASP delivery model, the ISV is relegated to only a small component in the overall solution in which the ASP manages the entire customer relationship. For this reason, major software houses such as Oracle, SAP and IBM have already entered the ASP market as direct competitors. Microsoft has given mixed signals as to whether or not it will enter the market directly. However, Microsoft has signalled a significant backing of players already in this market with its recent .NET platform and through its ASP Partner Certification Program.

**Internet Service Providers (ISPs)**

Telcos and ISPs are the ‘connectivity’ component of the ASP model. Many of the larger telcos have the network and data centre capabilities, but typically lack the application expertise that an ASP calls for. As we shall see in later in this chapter, many analysts think the telcos should, and will, buy an ASP because of their depressed market capital values. Exaggerating the push for telcos to buy ASPs, broadband services are quickly becoming a commodity with low margins, and adding a hosted service would increase their return on
network investment [King, Sept 2000]. Qwest, AT&T and Sprint are all involved in joint ASP ventures.

Data Centre

Often referred to as Server Farms, data centres can be owned by the ASP (as with USi) or owned by someone other than the ASP (as with Corio). Often The data centres are often owned by telcos, however Exodus and Concentric are both data centre only services. Whether or not they will move into the ASP market is unknown at this time. There are two main categories of data centres: 1) co-location providers (co-lo’s) and 2) managed hosting providers. Co-lo’s offer clients a secure environment with reliably clean power in which to house servers. Managed hosting providers take the next step by adding IT support and hardware, basically offering clients turnkey hosting solutions. Forrester Research predicts managed hosting will account for 86 percent of an anticipated $12.5 billion data centre revenue in 2003 [Bernard, Feb. 2001]. Andy Hunn, a senior director of Business Development at MHP Digex has openly stated that “Co-location is already getting commoditized,” [Bernard, Feb. 2001] and the users of data centres desire the extras such as trouble-shooting services that managed hosts provide.

Furthermore, many ASPs are comparatively cash starved and investors are not as patient as they were with the dot coms when it comes to turning a profit [Wainewright, March 2001]. Fortunately for the ASPs that are not planning on building their own data centres, the market for data centre services is flooded at the moment. Data centre service providers were expecting the ASP market to be much larger than it currently is and they were also expected to be servicing the dot com market at this point in time [Bernard, Feb 2000]. Neither of these assumptions has held true and excess data centre capacity has therefore resulted.
**Customer**

ASP's affect the way companies deal with their customers. For instance, if a company outsourced its shipment tracking software to an ASP, it would make sense to leverage the fact that ASPs deliver their services over the net. One way to take advantage of this would be to have the customer contact the ASP directly to get information, or make their payments for goods. However this can be done in a transparent fashion. Consider the following diagram:

**Figure 2. Customer, Client And ASP Interaction**

How the System Appears to the Customer

How the System is really working
Here the customer has visited the client's website to obtain information about its order. Customers may believe they are interacting with the client's IT systems. However, transparent to the customers themselves, they are really interacting with the client's ASP. The ASP has all the data and functionality to relieve the client of the burden of supporting customer information requests. Note that a supplier, retailer or any other entity that is interacting with the client could replace the 'customer'.

*Application Service Providers (ASPs)*

The ASP generally does not own the software it leases out to clients. Companies like USi and Corio (both in the top five for number of clients and contract size [SCN, 2000]) do build some of the software and integrate the solutions they market. However they are in agreements with Microsoft to use Microsoft's development tools and if they are going to move toward aggregation or reselling, they will use one of Microsoft's ASP partners.
This diagram shows the many diverse activities involved in the delivery of an ASP service as put forward by Microsoft on its ASP partner website or as presented in [Cornfield, 2000]. Since ASPs interact with the software vendors and telecommunications companies (or their ISP divisions) and face integration, development and networking issues on a daily basis, a diverse set of skills are required to get an ASP running smoothly. For this reason, ASPs often strike deals with ISVs, telcos and consulting firms in order to leverage those firms' understanding of those areas.

Qwest Cyber.Solutions delivers infrastructure, software, application development/implementation, maintenance and enhancements, all in an end-to-end package. Cyber.solutions is a joint venture between Qwest communications and KPMG. Cyber.solutions has an obvious advantage by having such an intimate relationship with these companies: Qwest Cyber.solutions has access to Qwest's networking skills and
capacities at a preferential rate and KPMG brings in a “wealth of intellectual property”. Both have invested a great deal of cash into the company so even though it was in a negative cash flow position for a long time, it has out lasted most other start ups [King, Sept 2000]. Or even Pandesic, born as a joint venture between Intel and SAP in 1997, Pandesic designed its hosted e-business solution for small and midsize businesses. Pandesic leveraged its relationship with both parents and had what most in the industry considered a great service [McCabe, Nov. 2000]. However, Pandesic has gone out of business. The channel conflict on SAP’s part (between Pandesic and MySAP.com) is often blamed for this demise.

For ASPs, their relationships with the various industry players are extremely important for the survival of the company and adoption of its services within the industry. We shall continue our discussion of ASPs in section 2.4 where we shall consider the various types of ASP business models.

### 2.4. Business Models

This section examines the ways in which ASPs generate income and their internal operations. Service Level Agreements (SLA) and the role they play in the ASP/Client relationship shall also be discussed.

**ASP Market Internal Operations**

We can see in the Microsoft ASP Service Delivery Model (figure 3) that Microsoft subscribes to four specializations of labour under the ASP market. These shall be considered as the ‘business models’ for the remainder of this thesis.
Platform Enabler

Data centres and telcos fall under this category. Essentially we are talking about organizations that provide the hardware and infrastructure that make the ASP model possible. Platform enablers generally provide managed hosting to 3rd party ASPs to develop and deliver their solutions as network services. Generally they channel their services to end customer through ISPs, ASPs and developers. Qwest, Digex and Data Return are all players in this market. Platform enablers can make their revenue by charging for data centre usage and network services on the back end and then charging the customer directly for Internet access when they provide it.

Solution developer (or Provider)

Notice that this can be either an ISV (Such as Microsoft itself), a programmer designing web pages that drive the application, or a full-blown ASP. However under this scheme it does not matter. Solution providers are what first come to mind when one thinks of an ASP and they closely resemble the generic ASP at the beginning of this chapter. There is heavy overlap between this group and Application Aggregators. Therefore, Microsoft often uses the ‘developer’ moniker to make the distinction between these and aggregators because the developer has built the solution it is selling. In the earlier days of the market, the provider would own the customer relationship. However, providers are becoming displaced by the growth of aggregators in managing customer relationships. Solution developers can charge for their services directly or more often their charges are to aggregators. These charges are then passed on to the customers of the aggregator.
Application Aggregators

Corio, USi and Interliant all fall under this categorization. Although they do sell directly they can also use VARs (Value Added Resellers) or traditional system integration providers. The agent's most important roles are the management and maintenance of the software being provided as a hosted service. Aggregators allow customers to take the ASP model to a higher level of efficiency, by adding simplicity to the ASP customer relationship. Consider the following schematic:

Figure 4. Comparison Of ASP Market With And Without Aggregators

ISVs have been left out to simplify the diagram, however some or all of the entities in the diagram may be dealing with an ISV. Without an Aggregator involved, the client may have to outsource to several ASPs. However there are many shortcomings to this set-up. First of
all, most of the savings on the application management side are suddenly lost in contact management and integration if the ASPs are not coordinating this amongst themselves. Whereas with an aggregator involved, the client can provide a wider selection of services than any one ASP provider could. Furthermore, the client only has one contact to manage, and integration is not an issue the client has to deal with. Aggregators often simply bundle costs together and pass them on to the client. For the remainder of this thesis whenever aggregator is referred to, it is as in the Microsoft use of the word, not Scott Heinlein’s categorizations introduced at the beginning of section 2.2.

**ASP Agent**

Agents traditionally focused their business on integration and/or other professional services. To get a better idea of the role that agents play consider the relationship Andersen Consulting (now called Accenture) enjoyed with SAP. With the previous arrangement, Anderson contacted the company and customized the software to operate at its site. Instead under the ASP model will determine what pre-packaged applications are needed and communicate the customizations needed for the application to work at the client’s site.

The reason we introduce these new categorizations is that we will use these (with some slight modifications) as the basis of the framework we shall set forth in chapter 4. Although this model does not compartmentalize the ASP industry players as cleanly as we would like, we would be hard pressed to find a scheme that would since many players in this industry fall under many categorizations because they have different interactions with their clients.
A New Stratification Of ASPs

For the purpose of our analysis in chapter 4, we have created a stratification based on market (vertical or horizontal) and span of functionality provided (limited to enterprise wide). We define vertical to be relating to, involving, or integrating economic activity from basic production to point of sale. Horizontal is defined as business functions across several vertical markets. Consider the following two by two matrix, where 'market' is vertical or horizontal and 'span' is the number of business functions the services provided by that ASP support:

### Table 1. Categorization By Market And Span

<table>
<thead>
<tr>
<th>Span</th>
<th>Market</th>
<th>Horizontal</th>
<th>Vertical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Niche</td>
<td>Niche VSPs</td>
<td>Focused Solution</td>
<td></td>
</tr>
<tr>
<td>Enterprise</td>
<td>Enterprise Support Services (ESS)</td>
<td>Full Service VSPs</td>
<td></td>
</tr>
</tbody>
</table>

These four groupings are discussed in detail below.

**Focused Solutions**

Focused Solutions would solve one issue within one of the internal vertical departments (for example a demand prediction algorithm provider would be a Focused Solution within supply chain management). The service provided by a Focused Solution would be integrated into an in-house application or into the service of any of the other three ASP types. eBay has recently announced that it will be joining Microsoft's .NET framework. This means that any solution or service requiring an auction engine that eBay provides could be accessed and integrated into that service as described on the Microsoft web site. USInternetworking is another ASP who offers its services in this fashion. Because many companies are currently
adapting this framework, and few have announced it, it is hard to say which will be Focused Solutions and which will not.

**Full Vertical Industry Services**

Full Service Vertical Service Providers or VSPs are starting to emerge at a very rapid pace. The reason for this is simple: they intimately know both the industry they are in and the software they sell. This results in lower customization costs and better market acceptance (if one is in the restaurant business one would want the person providing software to understand the restaurant business). GovHost.com, (focuses on Government), AdWare Systems (advertising) and MIRUS (multi-unit restaurant chains) are all examples of Full Service VSPs.

What a vertical ASP (also called VSPs) brings to the table as its primary attribute is industry expertise. It offers customers more than just a new way to purchase software; it provides the specific industry knowledge needed to make a particular application solve a particular problem. “And, since every industry has its own set of unique needs, it is this knowledge that will be the differentiator at day’s end” [Bernard, Jan 2001]. The possible drawback is that this may eliminate a competitive advantage if adoption rate is high within an industry (Consider SAP and the petroleum industry). Although this would mean that the company offering this service would be very successful, its success could zenith and start to decline as SAP’s fortunes did when market saturation was reached.

Because these services are already specific to a particular industry, industry pre-customization lowers the transaction costs of contracting these services. And because the operating costs of the ASP are distributed over many clients within one industry, knowledge that is gained at other clients translates into savings for all the clients of that ASP.
**Niche VSPs**

Niche VSPs are the ASPs that typically offer email, accounting packages or supply chain software; services that are broadly applicable but would not satisfy the IT needs of any one organization. Niche VSPs are named VSPs since internal departments such as accounting or human resources are often considered vertical within the company. They allow the customer to focus on core business issues (Accounting and financials came in second in customer demand [Newcomb, Feb 2001]). NetLedger is an example of a Niche VSP offering accounting applications over the Internet and Virtual HR is a Niche VSP offering human resources over the Internet. One sub grouping of Niche VSPs would be as noted in the following table.

### Table 2. Subtypes Of Niche VSPs

<table>
<thead>
<tr>
<th>Enterprise support Services</th>
<th>Functionality that focuses on supporting the enterprise and integrating its processes. Traditional ERP software would naturally grow into this space.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis and Infrequent use (One off)</td>
<td>Any services that are designed to analyse a business problem, data warehousing and data mining, decision support systems, warehouse layout planning software and route planning software.</td>
</tr>
<tr>
<td>External collaboration support</td>
<td>This includes services that replace CRM, eCommerce and Procurement. Services that enable interaction between the client of the ASP and the external world.</td>
</tr>
<tr>
<td>Internal workflow support</td>
<td>This includes e-mail, groupware, knowledge management and document management.</td>
</tr>
<tr>
<td>Personal Productivity support</td>
<td>Office productivity services that would replace products such as MS Office or Corel WordPerfect (in their current format)</td>
</tr>
</tbody>
</table>

Let us explore each of these subtypes.

**Analysis and Infrequent use:** Why buy an application if it is only used once, or very infrequently? Why buy and support an application that is outside of the business's area of expertise or that requires hiring new talent? These questions are driving clients to analysis
and infrequent use software. Because the client only pays for the transactions it makes and not for a licence for the software each transaction has a lower cost associated with it. Many companies use warehouses but few specialize in optimizing their layout, hence buying floor layout software makes little sense for these companies.

ASP for infrequent use applications provide a ‘per use’ access to this application. Most companies today would like to have a data warehouse for reporting purposes, but few have the in-house expertise to build and maintain one (Cognos’ offerings are an example of such software). An ASP providing an online data warehouse allows these companies access to these services while sharing the cost of infrastructure across many customers. Currently, connectivity issues are holding customers back from using these services directly without an aggregator; however new technologies such as UDDI (Universal Discovery Description and Interface) and SOAP (Simple Object Access Protocol) could change this by reducing the costs of coordination and connection between new partners, as we shall see in section 2.6.

High specificity hinders profitability for this market since customization costs for short-term engagements easily eat into margin [Bushaus, Sept 2000]. It is notable that eCommerce and CRM were third and fourth in terms of number of customers planning on using an ASP to provide a particular service [Newcomb, Feb 2001])

Long-term analysis software faces a different challenge. It is because this software is usually very hard to install and customize to each client that ASPs in this area deliver cost savings. Because ASPs price their software post-customization [Trager, Feb. 2001] they reduce the uncertainty of the client with respect to costs of installation.

**External collaboration support:** External collaboration support, similar to infrequent use services, allows the outsourcing of services outside the company’s area of expertise. Because of the transparency of an ASP’s services to customers and other entities (see
customer section in Chapter 2), the ASP can alleviate the burden of supporting customers, and designing and maintaining a website or online store. Customization of services is costing this segment by increasing its operating costs. Companies such as Pandesic and Red Gorilla both offered eCommerce services, both tied their billing to transactions via the online stores they had built for customers, and both are now out of business because they over estimated online purchasing.

**Internal workflow support:** Internal workflow includes email and document management. This category includes email, the service that most buyers plan on getting from an ASP. Most of these services are work enablers, smoothing the flow of work within the company and hence lowering internal transaction costs, and their discontinuation would cause disruption of work, however the services are very low in specificity compared to an enterprise wide service. This type of service has a lot of upside for the customer and little potential down side.

**Personal Productivity support:** Office productivity services are services that would replace products such as MS Office or Corel WordPerfect. These services have been slow to grow because they tend to be user interface intensive, which in turn often results in numerous RPCs [ARIBA, 2000 (b)]. One solution has been to build these applications using DHTML (Dynamic HTML) because it utilizes the client computer's power. These services are fairly easy to maintain and are relatively easy to replace or substitute if need be. Getting these services online provides a cost savings (you only pay for what you use) and upgrades are someone else's concern.

Because the amount of specificity varies with the service offered we shall return to examine Niche VSPs, however some generalizations can be made at this time. Niche VSPs lower transaction costs in a way similar to Full VSPs, they specialize in one area (hence greater
domain knowledge) and enhance the connectivity of the product as much as possible. The application their service is based upon would generally be outside the client’s core area of expertise or business. They allow transaction cost reductions by allowing the client to focus on its key area of expertise.

**Enterprise Support Services (ESS)**

If the ASP also falls into the Aggregator or Pure-play subtype, it can offer a full suite of services for entire company using best-of-breed solutions (consider USi or Corio). If the ESS is an ISV (such as Oracle) the solution may not be as end-to-end because the ASP will attempt to only use its own products. ESSs behave the same as a Full VSP with respect to any given client; the difference is that their clients are in various industries whereas the VSPs clients are all in one industry. ESS, includes MySAP.com and Peoplesoft Online, however these can be industry-centric. For example, SAP’s software is used across many industries, however SAP can come in industry specific flavours (which may be considered vertical). These companies succeed based on the huge transaction cost savings factor they provide through simplification of client’s IT relationships (ideally one ESS would deliver all the client’s application needs by repackaging and integrating applications from numerous sources) giving the client one point of contact. Large and midsize companies are their most common customers. The largest drawback to their success is the vital role they would play in client operations [Bushaus, Jan 2001]. Customers realize these solutions have very high specificity and are vital to continued operations once they are installed, and customers may be wary of having such a dependence.

An extensive review of the literature reveals that customers tend to examine the ASP offering these services for financial stability to gauge whether or not they will fail. Often these solutions displace in-house IT staff as well as further increasing that lock-in may
already high due to specificity. Customer fears of lock-in to a particular service provider combined with customer concerns about the life span of the ASP has held this market back.

Here is what we have learned from recent ASP experiences: Full VSPs are succeeding in solving the needs of clients in their particular industries. The market is shaping up so that Niche VSPs (e.g. an accounting system provider) will often have their services resold or aggregated and then sold to clients [McCabe, Feb 2001]. Although there are a large number of VSPs (19% of ASPs were VSPs in 2000), few are in direct competition with one another [SCN, 2000].

The aggregator model that some Full VSPs and most ESS providers follow seems to be the most successful in the market today. While only 3.6% of companies planned on purchasing an online ERP system [Newcomb, Feb 2001], USi is the industry leader. Why is this? Because the aggregator model allows 'one-stop shopping' and flexibility that none of the other models provide, especially if the ASP is not also an ISV or is not too committed to one software vendor's products. Aggregators also allow a customer to 'ease' into ASP services 'step by step' rather than having a 'sink or swim' date. This is related back to specificity issues regarding lock-in and customization (which are discussed in Chapter 4 and 5).

**Service Level Agreement (SLA)**

Since ASPs, by their very name, are the providers of a service and play an outsourcing role, we would be remiss to not consider the role Service Level Agreements play in the relationship between client and ASP. Traditionally, an SLA would contain metrics by which an outsourcer (and sometimes the internal IT department) would be measured [Butler, 2000]. They would have goals based on those metrics. In an ASP environment, the metrics include, but are not limited to, system availability, response times, issue priority and quality standards [Butler, 2000]. For example, a call centre could promise to respond with a
solution to an IT problem within two hours for 65% of all calls. SLAs can provide the basis for measuring the outsourcer's performance by outlining the metrics by which they measured.

However, when dealing with applications, clear-cut metrics gauging functionality can be hard to come by. Issues regarding changing business needs can almost never be totally ironclad into the SLA. Furthermore, many companies that will be using ASPs will be outsourcing for the first time. SLAs have not eliminated problems in the traditional outsourcing world and they will not eliminate conflict in the ASP realm. That having been said, SLAs go a long way toward eliminating unstructured conflicts and providing a framework within which to resolve these conflicts. Even the very act of forcing the parties to sit down and lay out their expectations on paper make SLAs beneficial. We mention SLAs only to make the reader aware of their existence and possible impact on ASP/user relations.

One interesting start-up in the field of SLA management is the company Oblicore. Founded in January of 2000, Oblicore purports to provide ASPs and management service providers with the tools to draft measurable SLAs for all customers. The Oblicore software will measure the resources used to meet each SLA and help the ASP concentrate on fixing problem SLAs without affecting other customers' service [Dubie, 2000].

Oblicore says its software tells ASPs the application response time and the levels of network availability and latency the ASP can guarantee. Oblicore will also give service providers and their customers a view into the network and how its performing in terms of the SLA requirements throughout the billing cycle. If problems are encountered, the ASP can take action and possibly avoid penalties. Thus, SLA structuring is now offered as a service, to service providers. For further information regarding SLAs the reader is referred to Outsourcing: A CIO's Perspective by Oakie Williams.
2.5. Comparing The ASP Model To In House Solutions

At first glance, the ASP model bears many striking similarities to the mainframe model of old. It is analogous to the in-house network getting replaced with the Internet. However ASPs are much more than that. ASPs have the capability to deliver the high power GUI applications users have become accustomed to whereas mainframes have stayed distinctly on the number crunching side of the street [Slavid, Jan. 2001]. ASP delivery methods and Client/Server share a lot of the networking components, however processing and data are scattered all over the system. ASPs update software regularly and automatically (as far as the user is concerned) [King, Sept. 2000]. Clearly all three models have common traits, however they clearly have some very different characteristics as well. Let us compare the benefits and drawbacks of ASPs as compared with systems that have been designed and built 'in-house'. We start by examining the benefits the user of an ASP can expect and compare that to the benefits of an in house system.

The following table compares the ASP model to the traditional software delivery model based on the strength of each model.
Table 3. Comparison Between ASP And Traditional IT

<table>
<thead>
<tr>
<th>Benefits of the ASP model</th>
<th>Benefits of Traditional Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduced in-house IT demands</td>
<td>Control over server side architecture</td>
</tr>
<tr>
<td>Saved Money and Time</td>
<td>More custom reporting</td>
</tr>
<tr>
<td>No Additional Server software or hardware</td>
<td>Better performance of CPU intensive or heavyweight RPC programs</td>
</tr>
<tr>
<td>Few client applications to support</td>
<td>Customizability</td>
</tr>
<tr>
<td>Universal Access</td>
<td>Better offline performance</td>
</tr>
<tr>
<td>Shorter Application Cycles</td>
<td>Communication intensive applications</td>
</tr>
<tr>
<td>Ease of Integration</td>
<td></td>
</tr>
<tr>
<td>No obsolescence</td>
<td></td>
</tr>
<tr>
<td>Better distributed performance</td>
<td></td>
</tr>
<tr>
<td>Better customer service than in-house</td>
<td></td>
</tr>
<tr>
<td>Better for handling changes in demand levels</td>
<td></td>
</tr>
</tbody>
</table>

This table reveals that the ASP provides more benefits, however the drawbacks revealed by the strengths of the older system can be difficult to ignore. Custom reporting and customization are things that can eat into an ASPs margin because an ASPs profitability decreases with the customization each client requires [McCabe, Nov. 2000]. ASPs derive income from the 'repeatability' of their installations [McCabe, Nov. 2000]. Communication, or bus intensive applications such as graphics programs and CAD do not readily lend
themselves to the ASP models either. The ASP can provide file and data storage support to these applications, however it does not provide a viable replacement for these applications.

2.6. Technological Problems And Advances

ASPs have some technical considerations that still have to be addressed. Many of the technologies to maintain and support ASPs more efficiently are still being developed. Three of the top technological problems that have impact on customer acceptance and application performance and are currently holding back the growth of ASPs are as follows: 1) concerns over application performance because of networking issues 2) The lack of connectivity standards between applications often makes integration of services expensive and 3) Most software was not developed for the web and its online performance is lacking as a result. We discuss these issues because there are factors in some client's decision as to whether or not to purchase the services of an ASP. Let us examine each of these problems and discuss possible solutions.

Network Performance

Networking Issues include both response time uncertainty and reliability. Although the Internet is at a stage in its development where it is able to act as a delivery channel for software, it is by no means an ideal delivery platform. There are too many uncontrolled factors for any one ASP, or even any ISP to make a service level guarantee with regard to response times that it can really back up (other than to try to maximize the amount of bandwidth available to the user and hope for the best).

Needless to say, this is not an acceptable way of dealing with this issue if the applications that an ASP's customers are using are bandwidth intensive. Connection speed between any two points on the Internet is determined by the slowest point on the route chosen by the
routers between those two points. With current technology it is not possible to ‘prioritize’
data to insure customers requiring a high ‘Quality of Service’ (QoS). This means that
although other high priority has to get there immediately, data like videoconferencing has to
compete with lower priority traffic on the net. This problem is being dealt with by services
such as multi-label protocol switching (MLPS) and DiffServ. Both are aimed at allowing
voice and data to co-exist on the same network, and as author Mark Vail puts it:

The emergence of MPLS and DiffServ is enabling new packet-based access solutions
that can support and deliver toll-quality voice and data services. MPLS and DiffServ
together enable network operators to maintain a guaranteed QoS for IP, legacy voice
and data transmissions while providing tools for managing traffic, deploying and
monitoring SLAs and provisioning secure data VPNs across IP backbones. Ultimately, MPLS and DiffServ will enable service providers to provision QoS
across entire networks.

[Vail, Jan. 2000]

These services also allow for access quality to be dialled up or down depending on client
needs, thus telcos would be able to deliver various guaranteed levels of service and bill
each customer segment for the performance it demands, or even bill for high QoS time
periods in a fashion similar to the way telcos bill for long distance [Skemer, 2000].

When these services become available, ISPs can offer ASPs and their customers
guaranteed SLAs. This will alleviate most customer concerns with respect to response time
and reliability and thus more clients with see ASPs as a viable alternative to in-house
systems

**Lack Of Connectivity Standards Between Applications**

These drive up costs for any IT user or provider, whether the applications are online or not.
Easy integration would be of particular benefit to the aggregator or any client looking to
outsource only some of its IT application needs.
New enablers: UDDI and SOAP

With the emergence of business over the Internet, several business problems emerged for buyers and sellers of goods and services. New suppliers and buyers were on the Internet, however finding them, being sure of who they were and identifying what they are looking for or offering could be an onerous chore. New industry-wide panels came into existence to determine how to solve these problems. Consortiums such as W3C (W3 Consortium) and UDDI.org (Universal Description, Discovery and Integration) which is a joint panel formed by Ariba, IBM and Microsoft [Ariba, 2001(a)] took on these problems and came forward with the UDDI specification. UDDI offers businesses a standardized way to connect and interact over the Internet. Here is how UDDI defines itself:

To address this challenge, a group of technology and business leaders have come together to develop the Universal Description, Discovery and Integration [UDDI] specification - a sweeping initiative that creates a global, platform-independent, open framework to enable businesses to (1) discover each other, (2) define how they interact over the Internet, and (3) share information in a global registry that will more rapidly accelerate the global adoption of B2B eCommerce.

Each incremental advance in Web-enabled commerce has carried deep implications for business processes and organizational culture. UDDI is a major advance - the first cross-industry effort driven by platform providers, software developers, marketplace operators, and eCommerce and business leaders that comprehensively addresses the problems limiting the growth of B2B eCommerce, and that will benefit businesses of all sizes by creating this global, platform-independent, open framework.

UDDI is a building block to enable businesses to quickly, easily and dynamically find and transact with one another via their preferred applications.

[Ariba, 2001(a)]

The UDDI specification is based on the W3C's and Internet Engineering Task Force's (IETF) standards such as XML (eXtensible Mark-up Language) and SOAP (Simple Object Access Protocol), which we shall discuss momentarily. With the UDDI standard, ASPs have a formalized way to describe their interface, and services they provide.
This, combined with the advances in the software development arena, allows for the ASPs that were aggregated by providers such as USi and Corio, to now be 'discovered' by clients and incorporated into solutions that are built in-house with outsourced applications.

SOAP is an industry wide replacement for Remote Procedure Call (RPC) models such as CORBA and COM (or DCOM). SOAP uses no new technology, and is currently supported by both Apache and IIS web servers. SOAP uses an http (hypertext transport protocol) as a transport and XML as a data descriptor [Microsoft, 2001]. Traditionally, when one would connect to a 'server farm' over the net, http would be used as the transport and then COM or COBRA would take over the transport inside the farm. SOAP, overcomes the 'cultural' reasons for users to stay with one RPC or the other. This allows for a new level of interconnectivity between applications over an http connection (note that this is not just limited to the Internet itself). This combined with the UDDI framework allows for companies offering applications over the Internet to dynamically link and operate together [Ariba, 2000(b)]. This is a boon for companies offering low criticality, commodity or so called 'one off' applications.

**Most Software Was Not Developed For The Web**

Most of the offerings that ASPs are providing are simply older applications with a web enablement layer built onto them [Dering, Jan. 2001]. Many of the economies of scale that ASPs have been promising were lost in trying to deliver and interact with the software in a way for which it was not designed. Software vendors have started retooling their offerings to perform in this new environment. Citrix is the chief proponent of this approach and its software ports existing software for the net.

Microsoft is heading the other way with its .NET initiative. .NET allows the building of web native applications that utilize UDDI and SOAP to their fullest [Microsoft, Nov. 2000]. The
.NET development environment is designed so that online services can be used in a fashion similar to the way class libraries are currently used in the C++ environment. Many claim that .NET and UDDI represent the model for future software development [Kerstetter, Feb. 2001; Fonseca, Nov. 2000; Kucharvy, May 2000]. Microsoft is facing stiff competition from Sun and Oracle as those vendors retool their offerings as well [Kerstetter, Feb. 2001].

Many analysts are predicting that the major players will have to do more than just repackage their offerings for web sales. Gartner's Ben Pring, lead ASP researcher at Gartner, has stated the following:

2001 will be the year of net-native ASP solutions. The year in which the realization becomes apparent that ASP is not really an ERP play at all but just a stepping-stone from an ERP world to an Internet world. Net native applications fly on the current (let alone future) bandwidth in place in a way that Citrixised ERP applications will not be able to match.

[Pring, Aug. 2000]

This new connectivity is something that ASPs ignore at their own risk either way. Oracle is 'going it alone' in the ASP world, offering its software services directly to the customer. The problem? Quoting Art Williams, a director at Giga Information Group:

Some analysts say Oracle should get rid of Business Online altogether and simply use ASP partners. "They need to realize the real ASP opportunity lies in accelerating the uptake of their applications via the use of partnerships," Williams says. "Oracle will never satisfy customers, because they only offer their own applications."

[Maselli, Oct. 2000]

Oracle is ignoring two market trends: 1) In a interconnected world the users will search for best-of-breed [Dering, 2000] and 2) Users are not going to be as interested in brand names as they are interested in functionality and the consistency of its delivery [Kerstetter, Mar 2001; Sperling, Jan. 2000].
Aggregators ignore this threat at their own peril as well. UDDI/SOAP and .NET type development methodologies allow users to find and plug into ASPs directly. Although this is now possible, it is the belief of the author that UDDI/SOAP will allow users that are hung up on sunk costs to 'ease' into using ASPs piece by piece rather than having to immediately make the jump. This would satisfy the 'sunk cost effect', which shall be defined in Chapter 3, because existing investments would be phased out due to their age.

These changes in technology and market are coming about at the right time to enable ASPs to become accepted as relevant players in the software delivery market. The change that is occurring in customer attitude toward ASPs is an interesting one and it is unclear what stature the ISVs will hold in this new market.

### 2.7. Examples Of ASPs

This section is dedicated to looking at companies out in the market today or in days gone by. We start by looking at the market share of the larger ASPs.

**ASP Market Shares**

It is very difficult to accurately track the revenues generated in the ASP space since there are few ASP only companies and ASP revenues are blurred as they are often bundled with other services the company offers [SCN, 2000]. So in order to gauge market share we present two charts extracted from *ASP: Application Service Providing* by SCN Education. The first represents the market share by the number of customers and the second shows market share by number of contracts (to account for customers buying more than one service). Here we see market share by number of customers
Interpath, Applicast, Qwest and more are included in the other categories. Many analysts attribute the success of USInternetworking (USi is an application aggregator reseller) to its having started an ASP model with a clean slate and having built the company on a pure
USInternetworking’s chief advantage is big name clients “USi has been better at penetrating Fortune 50 companies” says Burney of Cahners In-stat [King, Sept 2000]. USi’s iMAP services have gone the opposite route of Breakaway and Intacct because it has bought its data centres. USi offers enterprise applications from the big names such as Microsoft, Oracle and Peoplesoft. USi feels that in order to get those big names the company has to own its data centres [King, Sept 2000]. Michele Perry, senior VP of marketing at USi, claims this success is based on the following three promises USi makes to its clients: “We guarantee what it will cost, when you’ll get it and that we’ll keep it running for the next three years,” she said in a September 2000 interview. In November of 2000 USi struck a deal with Microsoft where in exchange for $50 million in funding USi agreed to give Microsoft’s .NET managed applications services platform [Fonseca, 2000]. USi’s fourth quarter loss in 2000 was $12 million, down a third from Q3’s $18 million loss [Wainewright, Feb 2001].

Breakaway solutions became profitable in a hurry, well before most of its competitors. Breakaway began life as a systems integrator and hopes to continue on as an Aggregator ASP. Hosting revenues comprised only 15% of Breakaway’s $32.5 million revenue in the third quarter of 2000. “We’d like to get revenue up to 30% to 40% of total revenue base,” Dev Ittycheria, senior VP of the ASP group at Breakaway [King, Sept 2000]. The company has a gross operating margin of 55% and has done this by keeping its cost structure low. To do this, Breakaway has decided not to build its own data centres and instead leases space [King, Sept 2000]. The company has grown from 120 people to 900 employees in little over a year. Breakaway builds solutions tailored to each customer based on software already developed. Breakaway claims each new engagement usually requires a 20% customization of the software, however it is unclear as to how this metric is gauged.
Unfortunately, Breakaway has not remained profitable. Part of its funding agreement with Internet Capital Group forced Breakaway to take on dot coms that were also funded by ICG. This in turn squeezed out more viable potential customers that Breakaway was considering. With the dot coms die-out, many of Breakaways customers turned out to be insolvent, dragging Breakaway back to the red side of the ledger [Bushaus, Feb. 2001]. This is a problem to which many ASPs attribute poor year 2000 fourth quarter performance.

Oracle attempted the move from ISV to ASP relativity early [Greenbaum, Feb. 2000] and has wavered between maintaining its ASP stance or moving to the background and taking a more traditional ISV stance [Maselli, Oct. 2000]. Many analysts were claiming that Oracle should drop its Business Online division and allow ASPs such as USi and Corio to sell Oracle's applications, however Oracle has made several large steps in shoring up its ASP arm by restructuring the SLAs accompanying its contracts with clients [Maselli, Oct. 2000]. The new agreement offers very generous service guarantees to clients that are backed by a 25% rebate on monthly fees. In late February 2001, Oracle launched online tools for maintaining online stores just weeks after launching an entirely web based customer support application. These tools move Oracle closer to providing users with an entire suite of applications accessed only via the Oracle web site, rather than be hosted by an ASP.

Corio is an Aggregator ASP with a consulting background. Corio offers ERP applications from Peoplesoft and SAP. Because Corio customers were mostly dot coms Corio has yet to see a profit from its ASP services. However, despite the losses, Mark Verbeck, an analyst with Epoch Partners, says Corio has stayed on track by shifting toward recurring application-management revenue while partnering with companies such as Cap Gemini Ernst & Young for implementation services. The partnership lets Corio win larger customers, helping boost
its average new-customer contract to approximately $100,000 per month [Maselli, Feb. 2001].

FutureLink delivers software from ERP software from Citrix (in fact it is a top reseller) and has partnerships with Microsoft and Great Plains - recently Microsoft announced plans to buy Great Plains [Alexander, Jan. 2001]. However, FutureLink’s future is unclear. In the fall of 2000, the Lake Forest California, company announced minor layoffs, most of which were to eliminate redundancy caused by an aggressive acquisition strategy during the previous 18 months. But it has yet to integrate its companies effectively into one cohesive organization, and it must further expand its presence outside the United States to stay up to speed [Willis, Feb. 2001].

Other interesting players include: Qwest Cyber.solutions, a joint venture between Qwest communications and KPMG as described previously. Intacct focuses on specific vertical markets or specific horizontal applications. Intacct main product is focusing on Web-based accounting for fast-growing companies. Intacct is not building data centres; instead it is focusing its efforts on customer service. Pricing starts at a relativity low cost of $50 per user month. Even at this price, Intacct makes an 80% gross margin. Interliant has been an ASP since before the term ASP was coined by IDC – in 1994: Interliant was hosting applications. Interliant is currently EBITDA negative at negative 19.1 million mainly attributed to a great deal of recent capital expenditures. However, Interliant has garnered a wealth of knowledge about hosting and has a wide client base. [King, Sept. 2000].

We round out this list of examples by looking at where the major players in the market fall under the categorization scheme created earlier in this chapter however this list is by no means exhaustive. Furthermore, let us segment the customer base by the size that the ASP is targeting:
### Table 4. ASPs By Type And Target Client Size

<table>
<thead>
<tr>
<th>Client Size</th>
<th>SOHO</th>
<th>Small (less than 100 employees)</th>
<th>Medium (Between 100 and 1000 employees)</th>
<th>Large (More than 1000 employees)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASP Type</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Focused Solution</td>
<td>eBay, USi</td>
<td>eBay, USi</td>
<td>eBay, USi</td>
<td>eBay, USi</td>
</tr>
<tr>
<td>Full Vertical Service Provider</td>
<td>Javelinx, atyourbusiness.com</td>
<td>Javelinx, MIRUS, RetailAspect, atyourbusiness.com</td>
<td>Javelinx, govHost.com</td>
<td></td>
</tr>
<tr>
<td>Enterprise Support Services</td>
<td>FutureLink, Corio, Peoplesoft</td>
<td>FutureLink, Usinternetworking, mySAP.com, Corio, Applicast, eOnline, Peoplesoft</td>
<td>FutureLink, Usinternetworking, mySAP.com, Corio, Applicast, eOnline, Peoplesoft</td>
<td>Usinternetworking, mySAP.com Corio, AristaSoft, Oracle BO, Cyber.Solutions, Peoplesoft</td>
</tr>
</tbody>
</table>

Note: information from various sources throughout bibliography and company websites
Chapter 3. The ASP Market

This chapter provides the motivation for examining the ASP phenomenon by looking at the market in terms of revenue and investment. As the reader shall see, the market size projections are very volatile. However, all of them are fairly large, making ASPs of interest to investors and academics alike. Also highlighted is the apparent confusion amongst the analysts of this market. The chapter concludes with an analysis of current market trends and short-term projections and predictions for the market.

3.1. How Big Is The Market For Asps?

This section examines the size of the ASP market in terms of revenue and investment. It shall become clear that none of the ‘experts’ know what the market is going to do in the short-run.

How Much Revenue Is There For Asps?

Author Dawn Bushaus sums up the confusion among industry analysts with the following statement: “When it comes to ASPs, the only thing analysts can agree on is that the market will continue to grow” [Bushaus, Sept 2000]. The following table was complied from various sources throughout the bibliography and provides a snapshot of projections made in the past with regard to how big the ASP market would become, in terms of revenue.
Table 5. The Changing Projections Of ASP Revenue By Major Research Corporations

<table>
<thead>
<tr>
<th>Year</th>
<th>Pre-November 2000 (in billions)</th>
<th>Post-November 2000 (in billions)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001</td>
<td>Forrester: $23 B (old projection)</td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td>Giga: $2-$6 B</td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td>Gartner: $22.7 B</td>
<td>$23 Billion by 2003 over by several billion –Ben Pring, Gartner Lead ASP Researcher</td>
</tr>
<tr>
<td></td>
<td>IDC: $2 B</td>
<td>Detoitte: $48.5 B</td>
</tr>
<tr>
<td></td>
<td>Forrester: $10 B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yankee Group: $19.2 B</td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>Gartner: $25 B</td>
<td>Frost &amp; Sullivan: $12 - $17 B</td>
</tr>
<tr>
<td></td>
<td>IDC: $7.75 B</td>
<td>Gartner: $25.3 B</td>
</tr>
<tr>
<td>2006</td>
<td>Frost &amp; Sullivan: $25 B</td>
<td></td>
</tr>
</tbody>
</table>

Projections made prior to the spring 2000 market correction tended to be much higher then predictions made afterward. However, even after the correction, many analysts stood by their predictions. Three factors came into play in this behaviour: 1) Analysts felt that dot coms were overvalued and the depression this caused in the market was temporary [Trager, Dec. 2000] 2) ASPs looked like solid business models when compared to businesses like Socks.com that sold socks online 3) They did not realize all IT businesses would be painted with the same brush as dot coms by customers and investors [Trager, Dec. 2000]. In December of 2000, the ASP market projections started to contract drastically because of the fact that the chill on IT investment and purchasing did not lift and because talk of a recession in the US made analysts think the set backs that first showed up over the summer would be around longer then they had anticipated [Trager, Dec 2000].
Since most ASPs are fairly young, they need a cash rich environment in order to survive the negative cash flow start up period. As a result, many good applications will, and currently are, being sold at well below what their market value would be otherwise. This low price exists not because of the shortcomings of those applications, but because the company that created it did not have the needed capital to survive this contraction [McCabe, Nov. 2000].

Consider the following statements

Frost & Sullivan’s Stan Prescott is forthright enough to wish he could get back his hanging curve of a projection. The company is on record predicting $26 billion in 2006 revenue for an industry broadly defined as including hosting and managed services...’Now that we’re all seeing a slowdown in the economy, I might want to pull that back a little bit.’ Prescott says. ‘I’m seeing numbers tossed around for 2004. The high numbers that are realistic are $17 billion, and the low numbers are around $12 billion.’

[Trager, Dec. 2000]

There is apparent confusion even at Gartner: “Pring was quoted last March [of 2000] as saying disappointing demand made his prediction of $22.7 billion in ASP revenues by 2003 overblown by several billion. But in August [of 2000], Dataquest came right back with a $25.3 billion target for 2004” [Trager, Dec. 2000]

For the majority of 2000, Gartner Group repeatedly made the following projections: Of the current 480 ASPs, 60% (288) will be out of business by the end of 2001, 20% will survive as providers of ASP products or vertical services in 2004, 16% Will be bought out by another ASP or be out of business by 2004, 4% (20) will still be in business as full-service retail ASPs in 2004. “Industry watchers generally agree that the ASP business model is worth the hype, but they caution that there is going to be a shake-out soon that will leave only a few service providers standing.” [Bushaus, Nov. 2000]
This is combined with a currently lukewarm customer response to the industry. “An electronic audience poll during the State of the Industry panel [in September of 2000] indicated that 60 percent of the several hundred solution providers in attendance were actively developing an ASP strategy. However, only 31 percent said they viewed the ASP model as ‘extremely important’ to their businesses currently.” [Longwell, 2000] Or consider the damning with faint praise language Goldman Sachs analyst Gregory Gould uses in a note to clients on newly public ASP Corio, whose stock trades around $11, three dollars below its late July offering price. “Although the ASP opportunity makes sense, we believe it is necessary to exercise caution when assessing the specific business models, many of which are young and likely flawed.” [Lashinksy, 2000]

Many potential customers are waiting on the sidelines watching to see who will survive this stage of the market. However Mark Chestnut, who is Microsoft’s director business development efforts with ASPs and ISPs, had this to say in a recent article in The Net Economy: “Customers are delaying purchase decisions because they don’t know whether the ASPs will survive...ASP's aren’t getting enough customers to survive” [Trager, Feb 2001]. However, the telcos, analysts, authors and ISVs all maintain that this is the software delivery model of the future.

Who Is Investing And How Much?

Figure 7 shows VC (Venture Capital) investment in communications startups:
These charts represent VC investment in communications startups. Even though the chart on the right represents four times the amount on of capital of that on the left, as a percentage ASPs have lost ground. Equipment and infrastructure include network and broadband delivery equipment. ESPs (Emerging Service Providers) are essentially Competitive Local Exchange Companies (CLECs), providing multiservice access services and integrated providers such as in-Building service providers.

Eastern Management Group, which monitors telecom industry VC investment as part of its Venture Compass service reports VC investment in ASPs actually declined from $1.54 billion in the first quarter to just under $1.21 billion in the second quarter. Meanwhile, investment in competitive service providers increased nearly fourfold over the same period, from $446 million to more than $1.81 billion. From an interview with Robert Saunders, an analyst at Eastern Management, Jonathan Blum reported the following:
‘There is no question that the ASP sector has taken a hit,’ says Saunders. ‘Though ASP deals are still getting done, the VC community clearly feels that the loss of equity valuation for ASPs and the fewer IPOs coming to market have eroded the value of these businesses. The spring [2000] correction was the clear cause...whether the trend away from ASPs holds over the long run remains to be seen. I would not count ASPs out,’ Saunders says. ‘But clearly the market is cycling away from those businesses as of now.’

[Blum, Sept 2000]

Many industry watchers are speculating that the falling share prices of ASPs may allow big telecoms like AT&T, WorldCom and Sprint (all of whom have gone on record stating they would not become ASPs) to buy ASPs at wholesale prices [Bushaus, Dec 2000]. The main push for buying ASPs is that the basic hosting offered by most telcos is quickly becoming a commodity, with rapidly shrinking margins and alternative, new sources of revenue are likely being considered.

Because AT&T and Sprint are weak in web hosting services, they would find ASPs such as USi attractive because it owns its own Data Centres. WorldCom, which is already established as a top hosting provider through its UUNet subsidiary and Digex would likely prefer ASPs such as Breakaway or Corio because it does not need to buy more data centres, and both of these ASPs lease third party data centres [Bushaus, Dec 2000].

Although rarely making any direct cash investments, Independent Software Vendors (ISVs) are making a significant investment in product re-engineering and training (see mysap.com, Microsoft’s .NET, Sun’s Suntone). They are also lending their names, and hence their credibility, to startup ASPs that have passed their certification programs.

3.2. Market Trends

We start by examining what applications clients are seeking. Communications, financials and e-commerce are the most common types of applications being contracted from ASPs
according to an ASP Industry Consortium customer tracking survey released on February 1st 2001 [Newcomb, Feb 2001]. This survey was of 137 senior and executive level managers and IT professionals in the U.S. that have purchasing authority for or involvement with general office productivity or software. All had indicated that they were currently using or had plans to use the services of an ASP within the next year. Here is a table outlining the results (note that percentages do not add up to one hundred since multiple answers were allowed).

Table 6. Applications By Percentage Intending To Buy From An ASP

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communications (such as email)</td>
<td>33.6%</td>
</tr>
<tr>
<td>Financials and Accounting</td>
<td>24.8%</td>
</tr>
<tr>
<td>E-commerce (catalogs and transactions)</td>
<td>21.2%</td>
</tr>
<tr>
<td>CRM</td>
<td>19%</td>
</tr>
<tr>
<td>Education and Training</td>
<td>18.2%</td>
</tr>
<tr>
<td>Human Resources</td>
<td>13.1%</td>
</tr>
<tr>
<td>Project Management</td>
<td>9.5%</td>
</tr>
<tr>
<td>Sales Force Automation</td>
<td>8.8%</td>
</tr>
<tr>
<td>Personal Productivity (Office, word)</td>
<td>8.8%</td>
</tr>
<tr>
<td>Enterprise Resource Planning</td>
<td>3.6%</td>
</tr>
<tr>
<td>Virtual trading communities (E-Hubs)</td>
<td>3.6%</td>
</tr>
<tr>
<td>Supply Chain Management</td>
<td>&lt; 1%</td>
</tr>
</tbody>
</table>

These numbers are given here to provide the reader with an appreciation of the applications that clients would likely seek from an ASP. In this section we shall consider what impact the particulars of the application will have on customer adoption. We shall look at the services
from the following aspects because they seem to be the critical factors shaping client interaction with the software and client purchasing decisions.

**What Is The Economic Outlook For Asps?**

ASPs learned a lot in 2000 about how this market will operate. In response, many ASPs have changed their billing structures. Both USi and Corio have started asking customers to pay for part of the cost of software licenses up front [Kerstetter, March 2001]. USi has finished its three data centres and is projected to be profitable by Q4 2001. Corio is projected to be profitable by Q4 2002. Of the top ASPs, only Corio and Oracle have avoided lay-offs [Bushaus, Feb. 2001]. However, most analysts believe 2001 will be the year ASPs round the bend and start to grow in terms of market penetration, revenue and profitability [Bernard, Feb 2001; Kolbasuk McGee, Jan 2001]. Why is this the case?

Firstly, the problems that dot coms caused ASPs are starting to pass and ASPs are no longer relying on dot coms as potential customers [Bushaus, Feb. 2000]. As mentioned, some ASPs got roped into having to take on dot coms as part of their financing agreements - Breakaway is a prime example of this - and not only did they lose payments when the dot coms went out of business, they also lost customers that had to be turned away [Bushaus, Feb. 2000]. Hence, ASPs are turning to traditional businesses, i.e. ‘brick and mortar’, as new customers and are activity jettisoning dot coms as a drain on resources [Bushaus, Feb. 2001].

Secondly, ASPs are one of the few areas in IT that stand to benefit from the uncertainty that has hung over the North American economy at the end of 2000 and the start of 2001. The reasons? “With executives nervous about committing large investments and large investments and many capital spending budgets frozen, enterprises will favour the drip-feed,
pay-as-you-go, ASP subscription model." [Wainewright, March 2001]. As if to back this analyst up, Qwest Cyber.Solutions landed a $22 million dollar deal with Expanets. This tops the previous record-breaking deal for the ASP industry of $18 million (also by Qwest Cyber.Solutions) with Redback Networks. Other factors that are bolstering the long term viability of the market is that 92% of the customers of ASPs have reported that they are satisfied with the service they are being provided with according to an ASP Industry Consortium survey [Newcomb, Feb 2001].

The main point being made is that the future for ASPs is not as gloomy as the present. Sanford Brown, vice president and general manager of AT&T’s hosting services, framed it nicely when he say “There’s been no denying the trough of disillusionment, but despite the tough times, the future really does look quite promising. The business issues that companies are facing – they’re not going to go away” [Koblentz, Feb. 2001].

**Changing Customer Attitudes**

Customers are starting to change the ways in which they interact with ASPs. As mentioned in Chapter 2, ASPs represent a fundamental change in viewing software as a service and not a product. This has some rather interesting consequences. Firstly, software is starting to follow a utility model [Gruhn, June 2000]. This commoditization of software has been noted by other authors as well [Sperling 2000, Kerstetter 2001]. Customers are following a progression from product to service and then service to utility. With a service, method of delivery is an important factor, however under a utility model, the only concern is delivery [Gruhn, June 2000]. Today, customers perceive the hosting model as a new type of applications delivery model. Just as in the past they are still activity engaged in selecting the specific application they want to use. In a utility model the customer’s perceptions, expectations and investment style changes. Customers do not take ownership of the
software or the infrastructure the applications the applications runs on. Instead, they buy the functionality and services it provides and have little concern about how that functionality is delivered [Gruhn, June 2000].

Consider the following:

USi, one of the pioneers of hosted applications, reports a marked change in the way customers are inquiring about their offerings. Last year, customer inquiries were always couched in terms of the branded application (e.g., “I want to talk to you about hosting Siebel software”). Today, only 25% of inquiries mention the software brand. More than half (52%) simply asks about USi’s financial, sales-force automation or e-commerce services. USi sees this customer behavior change as an indication that customers are accepting the hosted paradigm and are seeking out the company based on its expertise rather than the brand of application; but this also suggests a longer-term change in customer behavior. Customers are disconnecting from the application brand and looking at software as a service that provides them with desired functionality. From this point, it’s only a small to disregarding the application’s underlying hardware, operating systems and middleware. And it’s yet another small step for customers to begin looking at these solutions as a utility that delivers desired functionality.

[Gruhn, June 2000]

Data mining and OLAP followed this same route and are now considered standard functionality requirements. Microsoft ships SQL Server 2000 with OLAP and data mining features as part of the DBMS. One author worries that ASPs may move to being commodity so quickly that they may never be able to recover their initial investments [Sperling, 2001]. Unlike other utilities, ASPs would not have a stable commodity to deliver to the customer.

This concludes Chapter 3. In chapter 4 an analytic framework to determine the factors in ASP success will be created.
Chapter 4. ASP Framework: Deciding ASP Type

This chapter sets forth an analytic framework for explaining current trends in the ASP market space and for making predictions as to the success of various players in that market (success of course being defined in terms of continued business operation and growth of both revenue and profit). Since we are attempting to predict outcomes that have not yet occurred, our forecasts are based on a simple conceptual analysis rather than on a systematic empirical study. A conclusive testing of this model and the predictions made will therefore require further empirical and analytical work. Nevertheless, some examples that would seem to support the predictions made are given wherever possible.

The full analysis of the model is spread over Chapters 4 and 5. Chapter 4 examines the decision as to whether or not to buy the services of an ASP. Furthermore, if the services of an ASP are going to be purchased, from what type of ASP would they be purchased. However Chapter 4 focuses more on the interplay of factors in producing the type decision than it does the decision as to whether or not to hire an ASP. In Chapter 5 we focus more on the forces pushing a client toward, or away from, buying services from an ASP more than we have in this chapter by a review of the literature and then we use that literature review to refine the findings of this chapter. Once this is completed we shall have modeled the entire decision process behind the client's choices. We can then focus upon the strategies that ASPs should adopt in light of this decision process as we do in Chapter 6.

4.1. Economic Concepts Used In ASP Framework

This section introduces the economic theories upon which the analytical framework presented in the next section is partially based. One of the major economic concepts
forming the basis for this framework is transaction cost theory as explored by Oliver Williamson [Williamson, 1975; Williamson, 1979]. Market and hierarchy theories are presented. We provide the reader with a brief introduction to these theories so that the ASP framework model can be more easily understood.

**Oliver Williamson’s Transaction Cost Economics**

Transaction costs are considered by economist Oliver Williamson to be the “economic equivalent of friction in physical systems” [Williamson, 1985]. Englander quotes him as stating:

A Transaction occurs when a good or service is transferred across a technologically separable interface. One stage of processing or assembly activity terminates, and another begins. A well-working interface, like a well-working machine, is one where these transfers occur smoothly. In mechanical systems, we look for frictions: Do the gears mesh? Are the parts lubricated? Is there needless slippage or other loss of energy? The economic counterpart of friction is transaction cost: Do the parties to the exchange operate harmoniously, or are there frequent misunderstandings and conflicts that lead to delays, breakdowns, and other malfunctions?

[Englander, 1988]

For our purposes transaction costs are the costs associated with 1) determining the availability of services 2) determining relevant prices of services and 3) the costs of negotiating a contract. Environmental factors such as complexity of a good or service and uncertainty about costs increase transaction costs [Williamson, 1975] since more informational baggage must be included with each transaction. Transaction costs have been used in the analysis of hierarchical versus market structures in intra and cross firm interactions [Williamson, 1975, Englander, 1988; Malone 1987]. In the following section, factors determining the success of ASPs will be analysed in light of transactional costs, which are closely related to Williamson’s concept of the same name. It is our argument that
transaction costs are a fundamental force in the buyer's decisions as to whether or not to contract the services of an ASP(s) and play a profound role in the choice of ASPs.

Arguably, ASPs lower transaction costs for their clients in at least two ways. One, they have specific knowledge of the services they are offering. Two, because limited human resources are being spread across many users, the cost of those assets attributed to each transaction is lowered.

In [Malone, 1988] the authors put forth an analytic framework in an attempt to predict whether new information technologies will cause the coordination of economic activities, which are adjacent steps in the value chain. They state that economies have two basic mechanisms for coordination of the flow of goods and services through these adjacent steps: markets or hierarchies. Markets coordinate the flow of goods and services through supply and demand forces. When the buyer of goods and services compares its many possible sources and makes the decision based on their own criterion, a market structure is realized. Hierarchies, on the other hand, coordinate the flow of materials through adjacent steps by controlling and directing the flow of goods or services through a set of predetermined suppliers. Managerial decisions, not the interaction of market forces, determine the nature of hierarchical interactions. In [Malone, 1988] a framework stating that internal organizational structures favour markets when product-description complexity and asset specificity are both low; hierarchical structures are favoured when these factors are high (both factors are explained in greater detail further on in this section). The organizational form likely for the other two possible combinations of these factors depending upon which force is dominant. This is presented graphically in the following figure
In the framework put forward in this thesis, the concept of asset specificity is also used (and its definition is presented in the following section) however we are applying a similar logic as [Malone, 1988] to the ASP market space. However 'complexity of product description' is replaced with the broader concepts of transaction costs and switching costs. We shall reference this framework and use logic similar to this in order to explain some occurrences in the ASP domain today. We refine the concepts of Hierarchy and Market in the next section. Rather than applying them to the structure of an organization, we shall be applying similar concepts to the structure of the ASP market.

4.2. ASP Framework

In this section we apply and adapt the prior theory to determine the major forces in determining the success of an ASP. We shall then attempt to build a framework around these forces and based on this we shall try to determine when an ASP will be used, and if yes, which ASP model is most appropriate.
Our model is based on four input variables: transactional costs, specificity, criticality and substitutability. These variables are used to justify our possible outcomes, which are defined in the following section.

**Outcome Variable**

Our equivalent of a 'Market' structure as described in Figure 8, in the ASP domain would be a space where client relationships with ASPs would be similar to a "spot market." This is where clients could readily switch between ASPs and readily combine and integrate their services. It is our theory that transaction costs and specificity would be low for this to be the case. Under these conditions it is our argument that clients will buy services from various sources, as suits their needs, and link them together if the cost of doing this is low.

The choice of ASP type (or not to use an ASP) on the part of the client is the dependent variable in our framework, and is what the client will decide by measuring their needs against our input variables defined in the next section. We represent our stratification graphically in the following figure:
In this figure each quadrant represents an ASP type as created in Table 1. Not represented in Figure 9 is the decision not to use an ASP. We shall use the following codes to act as shorthand for each quadrant/ASP type:

Table 7. Codes For ASP Type Used In ASP Framework

<table>
<thead>
<tr>
<th>ASP Type or Decided not to use an ASP</th>
<th>Quadrant</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focused Solution</td>
<td>Q1</td>
<td>NV</td>
</tr>
<tr>
<td>Niche VSP</td>
<td>Q2</td>
<td>NH</td>
</tr>
<tr>
<td>ESS</td>
<td>Q3</td>
<td>EH</td>
</tr>
<tr>
<td>Full VSP</td>
<td>Q4</td>
<td>EV</td>
</tr>
<tr>
<td>Will not use an ASP</td>
<td>N/A</td>
<td>XX</td>
</tr>
</tbody>
</table>
Let us examine the impact the clients choice as to whether or not to use an ASP or use an ASP of type X further before we continue with the construction of our model.

In this model, if most clients favour Focused Solutions in the long run, then Focused Solutions would be considered the “successful” model. ASPs in NH, EH and EV would not be successful in this situation, because the client’s choice would effectively cut them out of the value chain. If clients favour Niche VSPs, then ESS and Full VSPs are still unsuccessful models. However Focused Solutions could still exist by selling their services to the NH ASPs. Similarly for long run choices of EH and EV, because EV is the full service VSP model, it is unclear whether or not they could dominate all vertical industries. However, if most clients favour option XX, then NV, NH, EH and EV are all considered unsuccessful.

As the cost of finding, connecting, coordinating and service customizations of a new ASP decreases (i.e. lower transactional costs and specificity), clients will go from a mode where they do not use an ASP to interacting with the minimum number of ASPs (i.e. use a full service VSP or ESS in quadrants 3 or 4) toward interacting with a number of Niche VSPs (NH). At the extreme they would combine the services of a number of Focused Solutions (NV or NV and NH). Some ASPs (particularly USi which has adopted the .NET framework [Fonseca, Nov. 2000]) would seem to be hedging the risks of uncertainty by offering both ESS and Focused Solutions, hence spanning NV, NH and EH.

NV, and to a somewhat lesser extent NH, correspond to the “market” in the analysis by [Malone, 1988] and would be characterized by free interaction between the client and numerous ASPs, with NV being a pure market where clients directly combine the services of many Focused Solutions. Thinking of an application as a collection of functionalities, with \( f(i) \) representing a unique piece of functionality and \( A(i) \) representing an application (in this representation ‘i’ is an indexing value), this would be represented graphically as follows:
Connections to the functionalities provided by various Focused Solutions and are brought together by the client to form application A(1). This implies that the client is responsible for the integration of various software functionalities (for various sources). This is not an easy task, and shall the implications of this shall discussed further in this chapter and Chapter 5. Note that the client is not limited to just creating one application and furthermore any f(i) could exist within the client if an ASP is not providing that functionality. In fact all f(i) would exist within the client if no ASPs are being used.

This arrangement would be the result of very low transaction costs and low specificity of services if the most f(i) are being provided by NV ASPs. The next step toward a hierarchy as we have defined it would be the use of Niche VSPs, which would appear in the following figure. Also note the client has created what we define as the package denoted P(i). Again the reader is reminded that 'i' is an indexing value due to the fact that more than one package maybe required. A package is a collection of an underdetermined number of applications:
Here the Client has contracted the services of a number of Niche VSPs (similar to Figure 10, any A(i) could be an in-house application). The connections between the layers in this figure are not as clear as in the previous figure. The connections between the functionality layer and the application layer are not necessarily across a SOAP connection to a Focused Solution and f(i) could be provided by an ISV. They could exist solely within an application hosted by the Niche VSP (in the case of an ISV providing f(i)). In this mode, the client is contracting by the application and may have to interact with more ASPs in order to acquire more applications. Travelling further up this hierarchy we arrive at the level of using an ESS or Full VSP. This choice would be equivalent to the hierarchies to which [Malone, 1988] refers because the client would have a long-term relationship with the ASP providing it with the desired package of applications. Regardless of market forces such as pricing, once an ESS or Full VSP is being used the client is locked into that relationship.
Note that the client may need more than one package (this is a shortcoming of Oracle Business Online as was discussed in Chapter 2) but ideally the client would need only one package. Any A(i) could simply be part of a collection of applications within the ESS or Full Service VSP (if A(i) is provided by an ISV) or they could be within Niche VSPs as would be the case if the top level ASP is an aggregator. Furthermore, in a situation where the client has made decision XX (remember this is the notation for not using an ASP) package P(i) would be provided by an ISV such as SAP or Peoplesoft. The advantage is that the client is
not concerned about what is occurring at the lower levels, just that the services they want in P(1) are delivered, they have little concern how the ASP brought them together.

However it is our theory that the complexity of product description and the asset specificity of most applications are so high that hierarchical industry structures (i.e. the client will prefer to deal with EH or EV ASPs), with the players locked into long-term relationships, will exist between most clients and ASPs for the foreseeable future. Furthermore, in order to minimise transactional costs many clients will opt for an ESS aggregator or Full VSP that can offer applications to support the highest number of business functions possible because this reduces their coordination costs to a minimum.

Because of the difficulty associated with software integration, the would be one of the primary forces driving the client toward using an ESS or Full VSP. This is accounted for through the transactional cost variable defined in the next section. It is clear from these figures that interacting with ASPs at the functionality or application level that the client has more relationships to maintain with ASPs, or that if only a few components of their IT infrastructure is provided by one or more ASPs, these services must be integrated with their existing in-house applications. This means that the client is supporting more relationships and integrating more software. As the progression is made toward EH or EV, the client is directly supporting fewer and fewer of these relationships and interconnections. However, if the decision is made to conduct all operations in-house, even though no relationships to ASPs must be maintained, the cost of this are higher than using an ASP (or so the proponents of this model argue).

We caution the reader that, although there are very strong parallels, we are not using these concepts of 'Market' and 'Hierarchy' in the same way as [Malone, 1988]. If the client decides to use a collection of Focused Solutions, this would most closely resemble the
‘Market’ structures defined in [Malone, 1988] as for the purposes of this thesis we re-use the term market but reapply it to the situation depicted in Figure 10. If the client decides to use a ESS or VSP then this most closely resembles the ‘Hierarchy’ structures as defined by [Malone, 1988], however we are reapplying this term to the situation depicted in Figure 12. Figure 11 represents a cross, or compromise between the two structures. Also, note that when we have a situation as set forward in Figure 12, the entities at the bottom (the Niche VSPs and Focused Solutions) may behave in a spot market like fashion, however from the client’s perspective this is a hierarchy situation.

We shall now examine why the client would make the decision to use one model or another or not to use the ASP model at all.

**Input Variables**

In this section we define and present what we consider to be the main factors in the client’s decision to use one ASP model or another (implicitly the assumption has been made that the client has decided to use an ASP, Chapter 5 presents the reasons why a client would use an ASP). It is our theory that the decision rests on these four variables: transactional costs, specificity, criticality and substitutability. We borrow heavily from the ideas of Oliver Williamson and [Malone, 1988], however we slightly redefine their concepts for the purposes of this thesis.

**Transactional Costs:** For the purpose of this thesis, these are the costs (whether opportunity or realized) of finding, agreeing on an SLA, connecting to, the costs of interacting with an ASP and the switching costs of changing from an in-house solution to a hosted service or switching between ASPs. Note that this would include the costs of customization. The main argument behind the ASP model is that these costs are lower over
the lifetime of the application then the client buying, installing, integrating and maintaining
the application for themselves (this is discussed further in Chapter 5). The cost of
interacting with an ASP can include the subscription fee (or whatever other pricing
agreement is arranged), as well as integrating the services of that ASP with other ASPs or
with any in-house applications that will remain in use (i.e. a coordination cost). All things
being equal, the number of ASPs and/or in-house solutions the client is dealing with, the
higher the transactional costs. The reason for using numerous ASPs could only be justified
(from a purely transactional cost point of view) if using those ASPs as opposed to using one
ASP lowers the costs of finding, connecting and interaction so the gains in customizability
would offset the increased coordination costs. This is not the traditional definition of
transaction costs as defined by [Williamson, 1979] but a variant of it we have created for the
purposes of this thesis. Furthermore we caution the reader that, although there are very
strong parallels, we are not using this concept in the same fashion as [Williamson, 1979;
Malone, 1988].

**Asset Specificity:** In [Malone, 1988] an input used by a firm (or individual consumer) is
considered to be highly asset specific (according to Oliver Williamson’s definition) if it cannot
readily be used by other firms because of site specificity, physical asset specificity, and/or
human asset specificity. Some examples of these types of specificity would be: a natural
resource such as an oil well is site specific because it cannot be moved without great cost, a
specialized machine tool or a complex computer program designed for a single purpose
would be considered physically specific. Human asset specificity would include learning by
doing, or skills possessed by a limited percentage of the population. We reuse this
definition of specificity in the model created in this section.
Criticality: Criticality is defined as the extent to which a good or service is necessary for the continued operation of the firm without extraordinary cost or inconvenience. For example an ERP service provided by an ASP would be considered highly critical since disruption of that service would disrupt almost all business functions. However a service such as email can be readily replaced so it is not considered to be highly critical. Note that if email service could not be reinstated in a timely fashion it would result in a disruption of business functions (this is due to the time component implicit in our definition of criticality). Criticality and specificity are logically independent, yet highly correlated in a very interesting fashion. The reason for this is as follows: if a critical IT asset is highly specific then it cannot be readily replaced and that lag in replacement is what disrupts business function. If an IT asset is low specificity then it can be readily replaced; and regardless of its criticality that quick replacement means it will not have the opportunity to disrupt the function of that business unless the continuous operation of that asset is crucial. It is difficult to imagine a situation in which an IT good or service would be have low-criticality, yet be highly specific, so we feel justified simply considering this to be a highly specific asset.

Substitutability: Substitutability is generally inversely related to specificity, however the two are logically independent. Substitutability is defined as the level of ease with which one set of functionalities can be replaced with another set that perform the same task. Email is an example of a highly substitutable service. The providers of this service are numerous and the specificity of this service is low so providers can be readily changed (unless there are contractual obligations preventing this). Substitutability is sometimes referred to by the inverse concept of ‘lock-in’.

We have chosen these four variables because they seem to include all of the questions the client would want answered with respect to the use of an ASP for providing a particular
application. The transactional cost variable accounts for price and cost concerns. The amount of specificity relates to the client's concerns about customization. Any IT decision's possible impact upon business functions must be accounted for therefore we include criticality. Also, as with all IT purchases, the client would have concerns about how readily that IT asset could be replaced and this is accounted for by the substitutability variable. We do not consider reliability at this stage because it would be more of a concern in the choice to use an ASP or not, therefore we leave it for Chapter 5. Note that performance and security would also play a crucial role in the client's choice to purchase an IT asset. However, as with reliability it plays less of a role in the choice of ASP type, so we also leave the discussion of performance and security to Chapter 5, Section 5.1.

We will usually limit the granularity of our variables to a simple discreet space of \{Low, High\} for the purposes of simplicity when making predictions. However when analysing them in the next section we shall vary them over a continuous space of \([\text{Low}, \text{High}]\). Having defined our independent variables, let us now consider their interactions to one another before we consider their impact upon the outcome variables.

**Interrelationships Of Input Variables**

As discovered in a review of the current literature, many of the proponents of the ASP model argue ASPs lower transactional cost in several ways. Many of their reasons for this assertion shall be discussed in Chapter 5. We assert that clients will opt to use only one ASP. If the customer currently only has need to outsource part of its current IT infrastructure, whether or not it plans on outsourcing more in the future plays a part in their choice of ASP type. For example, a company that plans having its email service hosted today and having its CRM hosted in the future would have two choices:
Option One: contract the services of a provider with email services only, and then in the future, contract the services of an ASP providing CRM and integrate the two or

Option Two: contract the services of an ASP that provides both services, purchase the email hosting services today and then purchase the CRM services once they are needed.

With the first option, a new ASP would have to be contacted and integrated with the first. With the second option, one ASP would suffice and would, as the analysis of transactional cost variable shall show, be a better situation. Any clients who chose the first option would likely switch to the single ASP option, and knowing this they would have incentive to pick an ASP that provides a wide breath of services, whether or not they currently need them, to avoid this situation. It is to capture this type of dynamic that we have separated switching costs out from transactional costs.

The applications themselves are very high in site specificity, as moving them from platform to platform on the server side can be a difficult task. That creates immense pressure to leave them on the site where they were first created. However this contrasts against the situation for the hosted services that are based upon these applications are very low in site specificity because they can be accessed from almost anywhere in the world. We are not as concerned with the site specificity of the application as we are the service that is provided, part of the reason for raising this point is as follows. Applications that operate between organizations (such as procurement or CRM) lend themselves to the ASP model because of the universal accessibility of a web-hosted service. This is where the low site specificity of the service is of advantage. Furthermore, transaction costs are reduced because of the easy access to the Internet both parties would have for connecting to the ASP. The ASP can then act as a buffer, protecting the client from the coordination costs of connecting to a new partner.
Hosted services can be high or low in physical specificity, for example email services require relatively little customization and hence are not very specific whereas an ERP system requiring a great deal of customization and client knowledge would be very specific. Human assets that support the hosted service are very specific and scarce.

As noted by [Malone, 1988] the specificity and complexity of goods or services are often correlated; however in most general discussions the fact that they are logically independent means that their have to be treated as separate factors. Site-specific assets can go against this trend. Consider, for example, an artesian well, the well is very low in complexity, yet it is highly site specific. Another example would be that of a complex good or service that is low in specificity such as an automobile. Most information technology goods and services are very complex when compared to other good and services; however, relative to one another, they follow a clear trend. Low specificity goods are usually those that are established and well understood by users and providers, require little customization and hence they are low in specificity when compared against other IT goods and services. High specificity goods and services are those that require a great deal of customization and are generally the more complex items in the IT realm.

Specificity is a double-edged sword. The buyer becomes vulnerable because as the asset becomes more and more specific via customization, it becomes harder and harder to find a replacement supplier. Conversely, by customizing their good or service, the seller has limited the number of buyers who would be interested in that good or service and is reliant on the continued patronage of the buyer to recoup that investment in customization. High specificity of assets tends to increase the lock-in of the customer [Williamson, 1979; Malone, 1988; Englander 1988].
Note that we are not talking about 'mass customization' but referring to further specificity required by each new client. Customization eats into an ASP's margin on a contract [King, Sept. 2000]. ASPs generally try to build their applications balancing functionality and required customization for each customer. This is where specialization is key for non-aggregators. Consider the following quote: "'What's different about us is that other ASPs have 10 to 12 applications [for rent],’ says CoreHarbor CEO Jay Chaudhry. 'If you do that, you become a generalist. We’re the heart surgeons when it comes to e-procurement .... We do Ariba day and night, so we really know how to support it.'" [Aun, 2001]. This means that for the maximum efficiency non-aggregators should minimise the costs of customization by focusing on the application they are renting. ASPs are selling a service. That service includes the customization of the software. ASPs make money when they do as little customization as needed and do what customization is needed as efficiently as possible. In the aggregator model, money is made purely on integration and customization cost reduction abilities.

However, for the current players in the market that are not adapting to the changing environment. New entrants have looked at what is working and built their own new applications with the web in mind (unlike many of the current offerings). Consider the following:

No matter how well they tune their engines, though, the first generation ASPs will have a tough time outperforming the newer entrants. The newbies establish one super-reliable Web site that all their customers hook into—plugging their information into simple templates. If the Young Turks get it right, the economies of scale could produce gross profit margins topping 90%—dramatically better than the 70% average among traditional software makers and the 15% to 20% margins that early ASPs have achieved, say analysts.

[Kerstetter, 2001]
This is the mass customization we referred to previously. This is important because these 'newbies' are lowering the switching costs between ASPs and hence reducing specificity. This force is a push toward a market style structure in the ASP industry.

Because clients are not going to be concerned with brand name if software becomes a utility as projected by many analysts, customer loyalty gets placed on the ASP and the functionality, loyalty to the underlying ISV erodes. Exacerbating the erosion of ISV importance, many ASPs are starting to create their own solutions, hence increasing margin by cutting out the ISV. This would be inline with our transaction cost theory since removing the ISV would further reduce the cost per transaction.

The lower the amount of customization required, the better for the ASP if customers are usually short term. For an ASP, lower customization means higher margin in the short run, which can be translated into further reducing transaction costs if the relationship is a short-term relationship. However from a specificity point of view long term relationships would be created with the lock-in of a high-specificity service; and high up-front customization costs could be considered an investment to gain a long stream of monthly payments generated by a long-term relationships. Services that are not easily customizable (i.e. are highly specific before the client purchases them) face a serious problem because of the high cost associated with customizing them: businesses don’t want to change their processes to suit the application. SAP experienced this problem once customers started to wonder about loss of competitive advantage. Many potential clients will not buy if the software is perceived to require business process changes. This makes further customization costly since the application is already highly physically specific. However, the customer often adapts to the software, and is effectively creating specificity by making himself or herself specific to the application, hence creating lock-in.
The net result of these interrelations is that many of these variables are not orthogonal (i.e. if the axis they would be represented by would not meet at a 90 degree angle if drawn in two dimensions). This means that a change in one variable often may result in a change of another (or others). Because it is outside the scope of this thesis we shall not delve into this in a much greater extent. We summarize the interactions of the four variables in the following table:

Table 8. Summary Of Variable Interrelationships

<table>
<thead>
<tr>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Impact of increasing Factor 1 upon Factor 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactional Cost</td>
<td>Specificity</td>
<td>No Effect</td>
</tr>
<tr>
<td>Transactional Cost</td>
<td>Criticality</td>
<td>No Effect</td>
</tr>
<tr>
<td>Transactional Cost</td>
<td>Substitutability</td>
<td>No Effect</td>
</tr>
<tr>
<td>Specificity</td>
<td>Transactional Cost</td>
<td>Increases</td>
</tr>
<tr>
<td>Specificity</td>
<td>Criticality</td>
<td>Increases</td>
</tr>
<tr>
<td>Specificity</td>
<td>Substitutability</td>
<td>Decreases</td>
</tr>
<tr>
<td>Criticality</td>
<td>Transactional Cost</td>
<td>Increases</td>
</tr>
<tr>
<td>Criticality</td>
<td>Specificity</td>
<td>No Effect</td>
</tr>
<tr>
<td>Criticality</td>
<td>Substitutability</td>
<td>Decreases</td>
</tr>
<tr>
<td>Substitutability</td>
<td>Transactional Cost</td>
<td>No Effect</td>
</tr>
<tr>
<td>Substitutability</td>
<td>Specificity</td>
<td>Decreases</td>
</tr>
<tr>
<td>Substitutability</td>
<td>Criticality</td>
<td>Decreases</td>
</tr>
</tbody>
</table>

We now consider the impact each of these variables would have upon the output decision variable.

4.3. Framework Predictions

In this section we examine the impact each input variable would have upon the output decision variable. Each variable is considered alone, with the orthogonal variables being
held equal and the resulting changes predicted by our model is put forward. Note that in each of the diagrams in this section the client is not necessarily moving from quadrant to quadrant, their choice is at the outset of deciding whether to use an ASP and if so which ASPs to contract. Furthermore we assume that the client is not currently using an ASP.

**Transactional Costs**

As mentioned, transactional costs (denoted TC in this section) and switching costs are interrelated, however it is our opinion that they are related weakly enough that the impact of varying TC over the space of [Low, High] can be ignored in the following analysis.

It is our argument that as TC is increased, specifically as the costs of finding, agreeing on an SLA, connecting to, and interacting with an ASP (or ASPs) increases clients will move toward an ASP that supports the widest number of business functions. Graphically this can be represented as the following:

**Figure 13. Client Choice Of ASP Type As Transactional Costs Increase**

![Graph of Client Choice Of ASP Type As Transactional Costs Increase](image)
In this diagram the arrow represents the choice of the client, and how it changes as transactional costs are increased. In this analysis, transactional costs are indifferent to whether EV or EH are used. An argument could be made that the client would favour EV before EH (remember EH is an ESS whereas EV is a Full VSP). However, it is our belief that this would heavily dependant upon what vertical industry the client is in since any VSP only operates within one vertical industry, and hence we do not include this at this juncture. Also, in this diagram and in Figures 14 to 18, once the arrow ends the client’s decision becomes not to use an ASP.

In [Malone, 1988] a cross-comparison of complexity of product description against asset specificity is used to determine the structure of intra and inter firm interactions. Even though the thesis of their work was that as information technology lowered transaction costs, firms will use markets structures more and more. UDDI and SOAP are technologies that lower the cost of finding and connecting to new ASP partners in NV and NH. Hence technologies such as these would lower the TC of connecting to various ASPs and push clients toward these quadrants. We feel that the reason for this is the control the client gains in the lower levels of our functionality hierarchy would then offset the costs of maintaining more relationships with ASPs.

UDDI and SOAP could topple aggregators in EH and EV since these technologies enable easy linkage between clients and Niche VSPs (NH) or Focused Solutions (NV). However, it would seem that clients buying into the ASP model would benefit from totally outsourcing rather then mixing ASP services with in-house client/server or mainframe systems. Full service ASPs deliver the most benefit when they are in control of all the applications the client is using, because this minimises the number of firms and services that must be
coordinated by the client. Once the TC of using an ASP becomes higher then using in-house solutions the client will make decision XX.

Asset Specificity

Like switching costs, we cannot consider asset specificity to be orthogonal to the other four variables. Increasing specificity invariably means increasing customization, hence impacting transactional costs and by definition increasing specificity increases switching costs. Increases in specificity imply a decrease in substitutability. Criticality can be held constant in this analysis. However, letting specificity dominate where package level services are required produces the following:

Figure 14. Client Choice Of ASP Type For P(i) As Asset Specificity Dominates

This is ignoring the other factors, however we shall view what we feel is the more realistic outcome when the other factors are taken into account. Our rationale behind the following is this: If specificity requirements are low, an enterprise wide service makes the most sense as
it keeps TC at a minimum and presumably an ESS would could provide the most generic solution that would satisfy those needs. As specificity becomes a greater concern a Full VSP would be somewhat “pre-customized” for the industry that the client is in, hence satisfying the increased need for specificity. Once specificity requirements increase further, the ability to ‘mix and match’ applications (and at the extreme functionalities) becomes the most important factor. If the client only requires an application A(i) this process starts in NH and ends in NV, if the client only requires a functionality f(i) the process would start and end in NV. However, this analysis ignores the impact that specificity has on TC, switching costs and substitutability. We feel that a more realistic outcome would be the following:

Figure 15. Client Choice Of ASP Type For P(i) As Asset Specificity Increases

Because the client is searching for a package P(i) it would make little sense based on our previous TC and switching cost analysis to consider NV or NH services. So to simplify the analysis we assume the client is seeking a package P(i) and then extend this analysis to A(i)
and f(i). As the required specificity increases the client would decide to use a VSP if one is within their vertical market, since these services would presumably require less customization at the outset. However the case where the client is searching for an ASP to provide A(i) or f(i) is rather interesting when the other factors are considered:

Figure 16. Client Choice Of ASP Type For f(i) Or A(i) As Specificity Increases

Note that this is close to being the complete opposite of the prediction made in Figure 15. The reason for this is as follows: assuming f(i) is required and specificity increases so would switching costs, to avoid the situation of coordinating with numerous Focused Solutions, or ASPs in general (i.e. in an attempt to minimise TC), the client would decide upon an ASP in NH (from this point onward the argument for A(i) or f(i) is identical so we shall only refer to A(i)). As specificity of the application increases further the switching cost and transactional
cost arguments again push the client toward an ASP supporting a wider number of business processes and from this point onward or argument parallels that for package P(i).

In Figures 15 and 16 the client's decision will become XX as specificity becomes extremely high. This argument is strongly paralleled by research in the field of general outsourcing [Butler, 2000]. However the client's decision in the case is strongly related to criticality and substitutability so before we analyze criticality and substitutability in the fashion we did the previous three variables we must first step back, and consider the choices the client is making before they have decided to enter into a contract with an ASP or build the desired package, application or functionality themselves. This is explained in the following, rather lengthy, aside.

The perceived level of criticality (to the client) would seem to be inversely related to the willingness of clients to buy into an ASP's services (i.e. high criticality implies the client will favour decision XX). Clients are aware that if the provider of a high criticality, highly specific service were to go out of business would halt their operations for an undetermined amount of time. Clients are also aware that this is a new market and Gartner is giving any particular ASP a 2 in 5 chance of surviving this year [Bushaus, Sept 2000, Pring Feb. 2001].

This has lead to an interesting set of paradoxes within the industry. It would seem clients would flock to the first ASP that was profitable, and in response many ASPs did not build data centres (consider Breakaway and Corio in Chapter 2). However the ASP with the largest client base is the only ASP that built data centres: USi. USi even goes so far as to use its ownership of its data centres as a marketing tool. This may be because clients view this as a more “real” company because it has real, tangible assets whereas the others may seem too much like the dot coms, where once they go out of business nothing remains.
Clients are willing to let ASPs handle 'headache' applications such as email and messaging. This due of the fact it is not hard to switch ASPs if need be with these applications since the specificity is low and they are highly substitutable. Because critical applications are usually not easily substituted, clients have two reasons not to buy in. Firstly, substitutability is low and specificity is often high. The client cannot easily buy these services somewhere else once they have moved to an ASP solution that has been highly customized. Compounding this lock-in is that ASP solutions often make in-house IT staff redundant, hence the IT department is often emptied and a new staff could be hard to get again. Secondly the dependence on the ASP is extremely high as well. Even if the ASP gave a copy of the software to each client when they went out of business, it is unclear if many of the companies would be able to maintain or run it on their own and most would surely lose the transaction cost benefits that drove them to ASPs to begin with.

A high criticality and a high substitutability situation would be dominated by the fact that new providers are easy to come by. But this logic only holds to a limit. If the system is critical in the extreme and virtually zero downtime is allowed, confidence in the provider may dominate.

If a service is of low criticality and not easily substituted, then the client has little concern if the service is offered or not since it is not critical, however it is unclear if this situation would ever exist. If a service is low criticality and easily substituted then the company has no downside for going with an ASP solution, and lock-in is low.

So, from the clients point of view, the following conclusions are made:
### Table 9. Client Acceptance And Lock-In By Criticality And Substitutability

<table>
<thead>
<tr>
<th>Criticality</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>In this case the amount of lock-in is high and criticality is low. It is unknown how the client would accept these services.</td>
<td>In this case lock-in is high and the criticality is high as well. Hence, the client would be less willing to purchase these services.</td>
</tr>
<tr>
<td>High</td>
<td>In this case lock-in is low. And since the service is not critical, the client would be very willing to purchase these services.</td>
<td>Although lock-in is low, the criticality is high. Hence the client's acceptance would depend upon which factor dominates.</td>
</tr>
</tbody>
</table>

Intuitively this table would seem correct since lock-in decreases with substitutability and acceptance increases with criticality. In terms of our outcome variable and specificity we arrive at the following conclusions: high criticality and low substitutability drive clients toward an XX decision. High substitutability and low criticality make clients choose from the set \{NH, NV, EH, EV\}, however in this case specificity is low. The outcomes of the mixed cases are unclear.

As we shall see momentarily, criticality and substitutability are weak variables with regard to the choice of ASP type in our model when compared to the other three input variables. However, they are powerful variables for explaining the decision on the client's part to use an ASP or not. With this in mind we continue in a similar fashion as before and assume the decision to use an ASP has been made.

**Criticality**

As the astute reader has probably deduced, our arguments regarding criticality has strong parallels with the arguments regarding specificity. Criticality's relationship to the other variables (that is to say: transactional costs, specificity and substitutability), is identical to
that of specificity, except with respect to substitutability where the inverse relationship is not as powerful. However criticality, as we shall see in the next chapter, is one of the major forces stopping clients from purchasing the services of an ASP in the market today. Regardless of this, we shall assume that the decision to contract an ASP has been made. Similar to specificity, let us first consider the case where criticality is allowed to dominate:

**Figure 17. Client Choice Of ASP Type For P(i) As Criticality Dominates**

Note that we could have used a figure similar to Figures 13 or 14 to indicate that criticality is indifferent to EH or EV ASP, however we have used this figure to emphasize the parallels between criticality and specificity. In this case, because the client is looking for a package P(i) we would assume that the other factors would drive them to contract an ESS or Full VSP. However as criticality increases, ignoring other variables, it would make sense for the client to decide upon a collection of ASPs from NH because they are no longer as reliant upon the continue operation of one company (so they are, in a fashion, diversifying their
portfolio in the ASP market by diluting the dependence on any one ASP) and at the extreme
the highest number of ASPs would be involved in delivering a package if only Focused
Solutions were used. Again, as with specificity, this argument starts with a Niche VSP for an
application $A(i)$ and starts a Focused Solution for a functionality $f(i)$. This is highly unrealistic
once the other variables are considered.

However we cannot make as clear a set of predictions with regard to criticality. It is our
belief that the decision path as criticality increases would mimic the decision path of
specificity in Figure 17; however there is no driving reason from a criticality point of view to
pick a ASP of the ESS or Full VSP type or vice versa. Ultimately, client choice with regard
to this variable would be based on an empirical study of ASPs in the market that seem to be
the most financially healthy and the reputations of the ASPs in the market for reliability of
services as discussed in Chapter 5. So, unfortunately, our analytical model falls short in
fully capturing the client behaviour in regard to criticality and further research would be
required to make any predictions on this variable when the other variables are taken into
account.

**Substitutability**

As explored earlier substitutability has a significant bearing on the choice of ASP type by the
customer. Substitutability is inversely related to switching costs and specificity and is
orthogonal to neither. As we have done with the previous two variables, let us vary the level
of substitutability while ignoring the other variables. No figure is required to describe what
occurs in this case, because nothing happens as substitutability is increased or decreased if
the other variables are ignored. Whatever forces cause the client to decide upon one ASP
or another are still dominant because substitutability, in our opinion, has no direct logical
impact on ASP type choice when considered alone. Similar to criticality, substitutability
based decisions would likely be the result of business analysis of the current ASPs and plays an important role in the clients decision as to whether or not to use an ASP to begin with. However when specificity is allowed to vary as our past analysis has stated they would the following decision path is followed:

Figure 18. Client Choice Of ASP Type For P(i) As Substitutability Increases

Not surprisingly this is the exact opposite of Figure 15 because substitutability is inversely related to specificity. And with regard to our output variable, the argument for this variable is the exact opposite of our analysis of specificity.

However, as mentioned, criticality and substitutability play a profound role in the decision as to whether or not to use an ASP.
Summary Of Framework Predictions

We summarize the predictions made in this section and in the next section we apply them in order to determine the strategies ASPs should follow depending upon what quadrant they fall into. In the following table we vary each variable over the range [Low, High] and allow the non-orthogonal variables to change in value as we have in Figures 13, 14, 17 and 19.

Table 10. Framework Predictions

<table>
<thead>
<tr>
<th>Variable</th>
<th>Decision Path as variable increases from low to high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactional Costs</td>
<td>NV, NH, {EH or EV}, XX</td>
</tr>
<tr>
<td>Specificity</td>
<td>NV, NH, EH, EV, XX</td>
</tr>
<tr>
<td>Criticality</td>
<td>{NV, NH, EH, EV}, XX</td>
</tr>
<tr>
<td>Substitutability</td>
<td>XX, EV, EH, NH, NV</td>
</tr>
</tbody>
</table>

These predictions are made in light of the non-orthogonal relationships between the input variables as we have described them. Even with as low a granularity as Q = {Low, High} these four variables would represent a vector space of 32 values, so our analysis is far from exhaustive on this space. Also, note that each of these predictions form a testable hypotheses with regard to the choices of the client. Empirical testing would be required to confirm or dispel these hypotheses.

This concludes Chapter 4. In Chapter 5 we focus more on the forces pushing a client toward, or away from, buying services from an ASP then we have in this chapter. Once this is completed we shall have modeled the full decision process on the part of the client.
Chapter 5. ASP Framework: Deciding Whether To Use An ASP

As seen in the previous chapters, the ASP model is one that is backed by many players in the IT world, including the current big names in software. Will this software delivery model succeed? In this chapter we set forth to examine the forces driving clients toward adopting the ASP model and the forces impeding the growth of the ASP model. These forces are the result of an extensive literature review or have been created by the author.

We reuse the variables of Chapter 4 in our analysis of these forces wherever possible, however as the reader shall see shortly, they are not enough to fully explain the decision to use an ASP. Hence we shall add variables to this part of the model which are strong in predicting if a XX decision will be made. Thus we hope to capture the entire decision process of the client in its decision to use an ASP or not, and if yes, what type of ASP. Once we have modeled this we then attempt to determine the strategies that the ASP should follow to maximize the likelihood of long-term success in the next chapter. For the majority of this chapter we shall use XX to indicate a choice not to use an ASP or ZZ to indicate that the client has chosen to use an ASP. Most times we can only state that a choice of ZZ only implies that ZZ ∈ {NH, NV, EV, EH}, without being able to say which element ZZ is equal to in most cases. So our outcome variable remains the same, except for the fact that we are unable to state clearly what the ASP type is in the following analysis. Also unlike our previous analysis, we have many more variables, which we define in the following section.
5.1. Forces That Are Causing Or Hindering ASP Growth

This section examines the forces that are driving the growth and demand for ASP services. As we have seen in Chapter 3, although declining, the investment in ASPs is still fairly large. What is driving that investment? Also, why are the research companies, despite their obvious confusion, projecting such large revenue for ASPs? The IT industry has several problems that ASPs purport to solve.

A thorough review of existing literature, and analysis of the various factors the author has found and created as a result of this review, are presented in this section. Furthermore the author aggregates the factors a client would use in its decision to use an ASP into the following categories:

- Pure information technology concerns
- Cost concerns
- Management concerns
- Inertia to stay with in-house systems

'Pure information technology concerns' include issues with regard to access, security and integration of applications. An ASP is supposed to bring along all the knowledge required to set up and maintain an application. 'Cost concerns' are simply the answers to the questions "Will an ASP save my business money, and how will an ASP save me money?" Although the expression "time is money" does hold true, most time concerns are included in 'management concerns'. In 'management concerns', we have included issues that generally are associated with IT, other than technical considerations and cost. These issues include: management distraction from running the business, troubles in accounting for IT costs, project management concerns and predicting cost. We find that it is in this area that ASPs
gain the most traction, and these are what clients focus on when considering a hosted application. Let us examine each of these categories in detail and look at their underlying factors. Theories about each individual factor are put forward below. The reader is cautioned and reminded of the following piece of symbol logic. That if the statement “If A then B” is true, it is not assumed that “not A then not B” is also true unless it is explicitly stated.

**Pure information technology concerns**

Pure information technology concerns include issues with regard to access, security and integration of applications. Some of the issues that end-users are concerned about in this regard are:

1. Geographically Distributed and/or Mobile Workforce
2. Security Issues
3. Performance Issues
4. Need for Internal and External Collaboration
5. Integration Issues

Let us examine each of these in detail and look at how these factors would impact the decision to use an ASP.

*Geographically Distributed and/or Mobile Workforce*

Portera frames the need of modern companies with employees who are widely distributed or highly mobile, to access company applications:

Distributed organizations and those with highly mobile workforces are prime candidates for a hosted applications solution. Good communication is critical to any
organization, but especially in environments where people don’t have the opportunity
to personally interact with all of their peers on a regular basis. Hosted applications
enable better communication within distributed firms by providing full-time universal
access to information regardless of location – bypassing the barriers and occasional
outages often associated with remote access procedures. Firms with centralized
employees who all work together in the same building will not enjoy many of the
benefits of a hosted software solution.

[Portera, 2000]

Clearly this is tied to the concept of low site specificity of the service provided by the ASP as
mentioned in Chapter 4. So if a company has a geographically distributed or mobile
workforce they have an incentive to take advantage of this property of the services of an
ASP. This argument could also be framed in terms of transactional costs, where the cost
per transaction of supporting this wide spread or mobile workforce is more expensive if it is
done in-house than by an ASP. Either way the following theory is put forward:

If the workforce of a client is highly mobile or geographically spread out then the client is
inclined to use an ASP, otherwise they would decide not to use an ASP.

Security Issues

Since clients often become more security and data protection aware, standard protections
like backups become more strictly enforced. Ironically, the security problems created by
dealing with an ASP are problems that already exist in most organizations today. Most non-
IT personnel and management believe that the transmission of sensitive data over the
Internet is unsafe, however consider the following:

"The risk of being intercepted is not even in the top 1,000 concerns for companies
today. Web sites make a point of using encryption to guard against a problem we do
not have....A case in point: ICSA.net has verified with all major credit card
companies, security firms, numerous banks, and law-enforcement agencies the
number of cases in which credit card information was intercepted over the Internet.
The answer: none. Ever. The real security issue when dealing with an ASP is much
closer to home. It involves locking down security inside your own company and
ensuring that your ASP does the same. When problems occur, it's at either end of the data transmission, and it's clear that on that score, both companies and the ASPs they use have a lot to think about."

[Gilster, 2000]

As with their own organizations, clients must insure the ASP they've decided to use is taking necessary security precautions. This is the crux of the statement that ASPs improve security: end users often overlook security gaps in their own enterprise, however once an ASP is involved they become hypersensitive to security issues. Furthermore, ASPs can hire dedicated security experts that none of their clients alone could afford. Although security is something holding back some customers, preliminary analysis of the model shows that security actually improves. So we propose the following:

If the client perceives that ASPs improve security then they are inclined to choose to use an ASP, otherwise they would decide not to use an ASP.

Note that this is based upon what the client believes to be true about security issues with regard to ASPs. As time goes on and the ASP model becomes more familiar that perception should become more accurate.

Performance Issues

It is almost impossible to guarantee response times over the Internet because there are too many factors outside the ASP's control. However, for some potential clients, a guaranteed response time is a must and may prevent some clients from using an ASP. Packet labelling will enable priority assignment, and thus allow ISPs to provide some level of service guarantees, which could be passed on to customers via the ASP.
Also, as mentioned, UDDI/SOAP and packet labelling will speed the market forward. UDDI will allow potential clients to link up with ASPs quickly and easily further reduce coordination costs by lowering up front investment. As stated in Chapter 2, packet labelling will allow a guaranteed quality of service level. We assert that as UDDI/SOAP mature, and packet labelling becomes available, applications that are specifically built for the online market come into being; hosted services will gain stature as their performance and ease of implementation approaches that of in-house systems. Furthermore, barriers to ASP growth, such as network response times, will be eliminated with packet labelling thus reducing uncertainty about performance.

Essentially, we are stating the following: As performance differences between the application as provided in-house or by an ASP become smaller the client is inclined to decide to use an ASP, otherwise they would decide not to use an ASP.

Need for Internal and External Collaboration

Consider the following:

Cross-organizational collaboration has become a necessity in today’s business environment. However, the overhead and security issues associated with enabling collaboration via the Internet has created a dilemma in many organizations between the need for network security and the need to share information with partners, clients and other third parties. Hosted applications resolve this issue by providing access to all customer-authorized parties, with higher security levels than most individual firms can attain.

[Portera, 2000]

Consider Figure 2 on page 11; in addition to the customer of the client, the ASP could also interact with any party that wished to contact the client electronically. Traditional connection methods such as EDI do not allow the same freedom of readily connecting to new partners.
or third parties in the way that the Internet (and additionally using SOAP and/or UDDI). This is one of the primary advantages of ASPs over traditional outsourcing.

For these reasons we make the following claim: As ASPs and technologies supporting ASPs enable the quick and cheaper creation of electronic relationships combined with the client's increasing need for these relationships, the client will be inclined to use an ASP.

Integration Issues

We consider software integration problems to be the primary reason that clients would move toward using an ESS or Full VSP because they are very difficult to manage, as we have mentioned in Chapter 4. For example, if a client needed an accounting package and a HR package and the client uses the services of two Niche VSPs to provide that, then the client would have to integrate these packages. If an ESS or Full VSP is used then the ASP integrates those packages. By using an ESS or Full VSP the client is benefiting those ASP's expertise in integration.

Although it could be argued that this is an 'ease of management' issue, it has been categorized with IT problems because although integration problems are usually difficult to manage, they are a problem that originates with the IT itself. Since the ASP and the client will have an SLA, if the client orders multiple services form the ASP it is the ASP’s problem to get those applications to work together. The only case were this would not be by default is when the client takes a mixed approach where a hosted application is to be tied in with a local non-hosted application. In this case it is not immediately clear who would be responsible for integration, however this would be an issue worked out in the SLA.
For these reasons we make the following claim: As ASPs increase the clients ability to integration applications easily, the client will be inclined to use an ASP.

**Cost concerns**

In this section the cost savings of ASPs are analysed. These savings can be significant, especially for larger companies and companies in the start-up phase. Of the many potential savings, following are some of the most common found in a review of the literature:

1. Lower up front costs
2. Cost of application can be expensed outright similar to a lease payment
3. Cost of applications are spread over many users
4. Lower Total Cost of Ownership
5. Applications that are needed infrequently can be used with out having to buy the application outright

Many of these benefits are obvious consequences of the ASP model and billing system. Hypotheses are not made on the first four points in the list above. This is because they are inherent properties of the ASP model and individual ASPs would not create strategies explicitly upon these factors. However they are part of the client's decision process so they are included for the sake of completeness. Let us explore each in greater detail.

*Lower up front costs*

Lower up front costs occur for the users of a hosted service since costs are deferred over the course of years. This is an obvious advantage over buying software. Often companies make large upfront investments in software packages only to find the package does not meet their IT requirements. This is of benefit to companies that do not have access to large
sources of capital and would not have been able to afford it otherwise. The slowing economy would seem helpful to ASPs. Uncertainty is driving many managers to avoid long-term commitments [Wainewright, March 2001; Koblentz, Feb. 2001]. Further reducing the costs of using an ASP is the fact that since ASPs have tremendous experience customizing their software, hosted applications are usually up and running much faster than in-house IT projects. Issues regarding hardware and networking are also greatly simplified under the ASP model, driving the cost of creating this relationship lower.

However, the price of starting a relationship, while still lower, is not as inexpensive as it once was. Up until recently ASPs followed a pure “pay per play” model. However, Corio and USi (two of the industry leaders) switched to a down-payment model [Kerstetter, March 2001]. What impact this will have on client acceptance, and whether other ASPs will follow this lead, is yet to be seen. Regardless, the economics behind the rent versus buy argument still holds true.

We feel that this factor is not a driving force toward the use of an ASP until a new functionality, application or package is needed because clients that are satisfied with their current IT infrastructure would not buy the services of an ASP simply because it is inexpensive to start. However, once there is a need for a new set of functionalities, this becomes a powerful motivator for purchasing the services of an ASP.

*Cost of application can be expensed outright similar to a lease payment*

Cost of application can be expensed outright (like a lease payment) without having to depreciate the cost of software. Under the traditional software as a product model, the huge upfront cost had benefit over many reporting periods (whether those periods are months or years). For this reason, the Canada Customs and Revenue Agency does not allow the
company to expense that outlay of cash, but treats it as the purchase cost of capital. This capital investment could never be fully expensed, therefore the tax benefits are only on the amount the product decreases in value according to capital cost allowances as set forth by the government. This is a primary advantage of using an ASP over purchasing software. This is a cost factor that would drive clients to decide to purchase an ASP when new functionalities are required.

Cost of applications are spread over many users

It has been estimated that the cost of maintaining a dedicated server compared to shared server resources can be as much as three times higher in a 15 month time frame [SCN, 2000]. This does not take into account maintaining the software that resides on those servers. So as a result the customer of an ASP saves on infrastructure costs.

More importantly, the customer saves because they do not have to pay for the entire application; they only pay for the part they are using (if per seat or per transaction billing is used). This has the side effect of allowing small users to access heavy-duty applications that were previously unaffordable, the effect of which cannot be understated. Furthermore this is one of the primary advantages of using an ASP over purchasing software.

Lower Total Cost of Ownership

Lower Total Cost of Ownership (TOC) exist since maintenance of the application and availability are the concern of the ASP. Furthermore, cost savings are realized because the cost of maintaining a desktop client, under the client/server model, is estimated at $10,000 dollars per year in some organizations [SCN, 2000]. Under a full ASP system, low-end desktops with a web browser can replace most of these. The savings on installation and
maintenance of desktop software could be in the millions for larger corporations. Lower transaction costs also play a role because the cost of the IT infrastructure to support the application is spread over many users. Also computing power, especially on the server side, is better utilized.

*Applications that are needed infrequently can be used without having to buy the application outright*

Warehouse layout planning and Warehouse locating software would be a good example of such software. Many companies use warehouses but few specialize in optimizing their layout, hence buying floor layout software makes little sense for these companies. These 'one off' applications would have to be purchased, or access would be gained through consulting companies that would be using the software. However in the ASP model clients pay on a per use basis, so there is potential for significant cost savings if the application is a one time use or infrequently used piece of software.

Hence the following claim is made: If a client intends to use an application rarely they are more inclined to have an ASP provide that application (in other words make the decision to use an ASP).

**Management concerns**

Management concerns include but are not limited to the following:

1. Need to quickly scale up or down based on workloads and growth rates
2. Yearly costs of application can be predicted more accurately
3. ASPs can be held more accountable for shortcomings than an internal IT department.
4. Upgrading issues

Let us explore these in detail.

Need to Quickly Scale Up or Down Based on Workloads and Growth Rates

Portera makes the following suggestion with regard to this issue

Fast-growing companies and those with highly variable workloads may be well-advised not to make the capital outlay associated with a client/server application with limited future capacity, and instead opt to adopt the “pay as you go” model offered by the ASP model – with virtually unlimited scalability. If, however, a company’s workload and growth rate is stable and projected to remain so, then a client/server application may make more sense. [Portera, 2000]

Obviously what Portera has said about client/server can be extended to mainframe systems or any in-house systems as well. This is because in-house systems are built to handle a particular range of workload. If the workforce or workload is curtailed the investment in the resultant excess capacity is wasted since in-house systems are often very site specific or physically specific. If the workload or workforce is rapidly increasing, an in-house system could be overwhelmed with demand levels it was not designed to handle. There is not as bad an excess capacity situation for the client if an ASP is used since the client is only paying for the part of the application they are using. Conversely, ASPs tend to scale much better than in-house solutions [Portera, 2000].

This leads to the following hypothesis: If a client is expecting a volatile growth pattern the client will tend to choose to use an ASP.
ASPs can be held more accountable for shortcomings than an internal IT department.

Service Level Agreements (SLAs) structure is very important in this regard. If an in-house project goes over budget in terms of cost or time, the company has little recourse. This is often the case with consultancies that make no guarantees (or often break promises) with regard to cost and deadlines. However SLAs form a legal and enforceable document between two independent entities. It is reassuring to the client to have some protection against these all too common events in the IT world. This concept is discussed extensively in the literature on outsourcing.

This leads to the following theory: the requirement of a set of functionalities that traditionally involve cost overruns, delays or extensive consultant activity will drive client choice toward using an ASP.

Upgrading Issues

ASPs have an inherent advantage with regard to traditional ERP and client/server packages. Many companies that bought into ERPs are not satisfied and are probably hesitant to upgrade their ERP packages. Large IT projects tend to take years, and once they start running well, an upgrade to the package is on the horizon (this is often referred to as ‘technology indigestion’ in the literature on ASPs). This is not going to change, in fact: “the rapid pace at which technology is changing is only going to increase even more” [Bushaus, Sept 2000]. The problem of software becoming obsolete is no longer a problem that the client has to worry about if an ASP is used because upgrading the applications is now the concern of the ASP.
This leads to the following hypothesis: If the client requires a set of functionalities then ASPs lower the costs of using that application by eliminating or reducing the costs of upgrading thus increasing the likelihood that the use of an ASP will be chosen.

**Inertia to stay with current In-House systems**

If a client decided to use an ASP they may have to walk away form existing IT investments, IT human resources and traditional methods of dealing with IT issues. What problems is this causing? Let us break down this issue into four major components.

1. Resistance from internal IT staff
2. Sunk costs of existing systems
3. The knowledge that is lost with IT staff is very hard to regain
4. Client uncertainty about the future for ASPs

These are factors that, as they increase, encourage the retention of in-house IT systems. Let us examine each.

**Resistance from internal IT staff**

Resistance from internal IT staff can be significant. As in [Phair, 2000] the previous 'IT champion' may not be willing to let go of current systems they had advocated and worked hard to get. This IT champion would also feel somewhat foolish admitting that systems they had advocated are already obsolete and may not endorse an ASP solution for that reason. Furthermore, IT staff would be aware of the fact that ASP solutions would make many of their jobs redundant, and that "knowledge empires" they have built up around existing systems would fall by the wayside [Phair, 2000].
This leads us to the following hypothesis: Internal IT staff resistance can reduce the likelihood of the client choosing to use an ASP.

_Sunk costs of existing systems_

Sunk costs of existing ERP, Client/Server or Mainframe system are hard to walk away from for obvious reasons. [Charley, 2000] refers to this as the sunk cost effect, when the bias is towards staying with the incumbent IT technology, even when the profit-maximizing decision for a firm would be to choose the ASP. This is of great concern to ASPs that replace the most expensive systems since potential clients may use this logic to justify not using those ASP's solutions.

This leads to the following theory: Large or recent sunk costs reduce the likelihood of the client choosing to use an ASP.

_The knowledge lost with IT staff is very hard to regain it if need be_

Once an ASP is adopted and internal IT staff is lost, the knowledge that is lost with them is very hard to regain and is discussed widely in the literature on outsourcing [Butler, 2000]. An interesting account of the consequences of this effect is found in [King, 1992]. This is closely related to the fear of ASP failure. IT staffing issues represent a double-edged sword for ASPs, such that it is a reason to buy and not to buy ASP services. This is closely related to the substitutability variable in the previous chapter.

This implies that the lock-in that results because the client has reduced staff is a deterrent to purchasing the services of an ASP. We shall not base a hypothesis about how the substitutability of the service impacts the choice of the client. However this idea is expanded further in the next factor.

95
Uncertainty About the prospects for ASPs

We have named the following cycle the “Catch 22 syndrome” it consists of:

a) Clients are not sure which ASP models will fail so they do not buy in.

b) ASPs cannot get clients because they are all waiting to see which ASPs will succeed.

c) ASPs fail without clients.

d) See a.

This is where some of the pall of the dot coms hangs over the heads of ASPs; customers have little faith in many IT companies and ideas. However this fear may not be unfounded, here’s why:

The Gartner Group predicts that 60% of them will fail in the next year [2001]. And by the way, that is a lot more serious than an online pet store suddenly running out of cat chow. After all, the demise of a company that runs your critical software would be a little like the power company’s cutting off your service for good.

[Lashinksy, 2000]

Again, Mark Chestnut’s quote comes back to us: “Customers are delaying purchase decisions because they don’t know whether the ASPs will survive... ASPs aren’t getting enough customers to survive.”

Mission-critical applications such as ERP and CRM were among the first to be offered by ASPs, but they can be a tough sell to customers concerned about turning over control and security of their data to unproven service providers. Even a successful sale results in a small number of seat sales because the majority of a customer’s employees do not need access to those applications, analysts say.

[Hagendorf, 2000].

If an ASP who provides mission critical application fails, results could be devastating, and it is for this reason many analysts project ‘commodity applications’ may be the earliest growth market space. “While analysts say the ASP model has been slow to catch on, they believe
a ubiquitous application such as messaging could be the ticket to mainstream success." [Hagendorf, 2000]. Furthermore "Customers already think of e-mail as a utility," says Laurie McCabe, vice president and service director at consulting firm Summit Strategies. "In every survey I've seen, messaging and collaboration were in the top half of those applications customers would consider hosting." [Hagendorf, 2000] Considering the ASP Industry Survey quoted earlier in Chapter 3 and the analysis of Chapter 4, this would seem to hold true.

The literature about ASPs focuses strongly upon this problem. Customers are not moving to the model and that is causing ASPs to fail. This is closely related to the analysis in Chapter 4 that lead to Table 8. This is the case with any new market and can only be overcome in time. We now state the previous theories explicitly in the terminology of this chapter:

As substitutability of an ASP's service decreases the likelihood of the client choosing to use an ASP decreases. Furthermore, as the substitutability increases so does the likelihood of a decision to use an ASP. The same set of statements hold true for criticality of the services in the reverse. As criticality of an ASP's service decreases the likelihood of the client choosing to use an ASP decreases. As the criticality increases so does the likelihood of a decision to use an ASP.

Low criticality, high substitutability and low specificity applications could be considered the most 'ASP friendly' as these applications are the easiest to sell, and cheapest to customize. High criticality, low substitutability and highly specific applications are the least 'ASP friendly' since they are the hardest to sell and most expensive to customize.

The next section summarizes the findings of this chapter.
5.2. Summary Of Predictions And Findings Of ASP Framework

Throughout the previous section we made hypothesis and theorized about the affects of several factors in the clients choice of whether to use an ASP or not. We bring each of these hypotheses together to form the second part of our ASP framework, which is summarized in the following set of tables. Note that some of these findings are ASP specific and some can be applied to outsourcing in general, we indicate those factor that are ASP specific.

Table 11. Factors Increasing The Likelihood Of ASP Usage

<table>
<thead>
<tr>
<th>Highly mobile or geographically diverse workforce (ASP specific)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client perceives that ASP usage improves security</td>
</tr>
<tr>
<td>Performance of applications same online as in-house</td>
</tr>
<tr>
<td>Increased need for external electronic relationships (ASP specific)</td>
</tr>
<tr>
<td>Client desire to integrate applications (Advantage over purchasing software)</td>
</tr>
<tr>
<td>Cost and time reductions in creating external electronic relationships (ASP specific)</td>
</tr>
<tr>
<td>Client needs an application infrequently (Advantage over purchasing software)</td>
</tr>
<tr>
<td>Client is expecting volatile growth pattern in workload or workforce (ASP specific)</td>
</tr>
<tr>
<td>Client expectations of cost overruns, delays or extensive consultant activity (Advantage over purchasing software)</td>
</tr>
<tr>
<td>High substitutability of services</td>
</tr>
<tr>
<td>Low criticality of services</td>
</tr>
</tbody>
</table>

The next table provides the set of our hypotheses with regard to factors that can increase the likelihood of the client's deciding not to use an ASP. This is different from the above
because we have found that not all of the above factor can decrease the likelihood of using an ASP as they decrease.

Table 12. Factors Increasing The Likelihood Of Not Using An ASP

<table>
<thead>
<tr>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low mobility or geographically central workforce</td>
</tr>
<tr>
<td>Client perceives that ASP usage decreases security (ASP specific)</td>
</tr>
<tr>
<td>Performance of applications worse online as compared to in-house (ASP specific)</td>
</tr>
<tr>
<td>Low substitutability of services</td>
</tr>
<tr>
<td>High criticality of services</td>
</tr>
</tbody>
</table>

These tables combined with Table 9 of Chapter 4 display the results of the framework the ASP framework this thesis set out to create. These three tables attempt to model the customers decision process as to whether or not to use an ASP and if so, which of the four types created in this thesis.

Contrast Between ASPs And Outsourcing

As discussed in Chapter 2, the fact that ASPs are accessible over the Internet is one of their primary advantages over traditional outsourcing of IT functions. This has several important implications:

- Because the Internet can be connected via standard networking equipment and protocols, ASPs reduce the need for infrastructure investment.
- ASP services can go ‘live’ in a much shorter time frame than other outsourcing methods because the services of an ASP are already running.
- Previously many smaller clients could not afford the investment required for larger applications and services with traditional outsourcing. However, because the
services of an ASP can have a low starting cost these clients can now have access
to these applications.

- The Internet allows third parties who wish to interact with the client to do so with
greater ease. Other forms of outsourcing do not allow the transparency to third
parties, such as customers, as an ASP does. Transparency is where the third party
believes that they are interacting with the client's IT systems, when in reality they are
interacting with the ASP's IT systems. This is depicted in Figure 2 of Chapter 2.

- ASPs can support the rapid creation business-to-business relationships between the
client and parties such as suppliers by acting as a buffer on the Internet. This can be
attributed to the fact that ASPs not only can integrate applications internal to
organizations, but can integrate cross-organizational applications as well.
Furthermore, ASPs can act provide the facilities for 'translation' between these
applications.

- Because the services of an ASP are often purchased on a 'per user' basis, ASPs are
better suited to highly volatile workloads.

- ASPs support a highly mobile or geographically diverse workforce better then
traditional outsourcing because of the universal accessibility of the Internet.

Now that we have shown why a client would opt to use an ASP as opposed to buying,
buying or outsourcing in the traditional ways, we now move on to the strategies ASPs
should adopt in light of this.

5.3. **ASP Strategies As Dictated By ASP Framework**

This section outlines the strategies an ASP should adopt. This is based on what type of
ASP maximizes the likelihood of success and what type of market structure it should expect.
This is based on the findings of this and the previous chapter. We shall use the ASP framework created in Chapter 4 and 5 as well as the findings in the other chapters to outline these strategies. We start by reviewing the findings of Chapter 4.

In Chapter 4 we found that transaction cost theory and our own findings would indicate that clients would tend to use a Enterprise Support Service provider or a Full VSP, unless they are confident that they will not outsource more than a limited amount of functionality or if transactional costs are so extremely low that coordination costs incurred are considered worthwhile when compared to the benefit of being able to mix and match the services of any set of ASPs and ISVs. We do not feel that transactional costs are low enough in this market because of integration related costs [King, 1992], so it is our theory that clients will tend to use either Enterprise Support Service providers or a Full VSPs. This implies that the ASP will take on the form shown in Figure 12. In that figure the client interacts with one ASP for a package P(i). This has the benefit for the client of offsetting all the costs of coordination and integration to the ASP. Under this market structure, Niche VSP’s service would be resold though an EV or EH ASP because transaction costs are minimised by having all IT functions handled by one contact. This also minimises transaction costs for the client with regard to connecting to new application services because an EV or EH ASP would be responsible for integrating that new service into the client’s package of services.

We predict the two main retail models will be Enterprise Support Service providers and Full VSPs, and that both should both have lower specificity services (which have less lock-in according to our theory) thus allowing them to gain customers. Once clients are using their services their high specificity would be used to maximize lock-in.

Niche VSPs must focus on lowering customization costs, and both Niche VSPs and Focused Solutions must improve interoperability by using technologies and standards that
reduce the cost of connection, thus lowering transactional costs. However, for Niche VSPs and Focused Solutions as a group, this will increase competition as switching costs are lowered. Furthermore, Niche VSPs must maintain their focus on a core service. They may gain the highest margin by selling services to the ESS and Full VSP models, hence cutting down on interaction costs with customers. Some Niche VSPs that own services with broad appeal, such as accounting packages, may be able to operate without depending heavily upon an Full VSP or ESS for resale. However most Niche VSPs will require an aggregator or reseller to provide them with an audience of possible clients, especially the providers of infrequent use applications.

From a specificity perspective, the client would prefer to be able to deal with Niche VSPs or Focused Solutions. This would allow the client the highest degree of customisability. Technologies such as UDDI, SOAP and XML are attempts to push the market in that direction by lowering the transactional costs of these relationships. However, the integration problems the client would face are great, and as a result the transactional costs of maintaining several relationships would be so high, that we conclude that the client would move toward an ESS or Full VSP. Some research in other areas of IT outsourcing arrives at the same conclusions [King, 1992; Loh, 1992].

Next generation development tools and techniques should further increase the transactional cost savings by reducing the costs of increasing specificity. ASPs have changed the ISV market and how ISVs interact with the end user of their products [Bushaus, Sept. 2000; SCN, 2000]. It is unclear what impact this will have on ISVs, however there is the potential for them to be displaced. Networking improvements that make QoS more predictable will instil greater client confidence and also spur ASP growth.
Enterprise Support Service providers or Full VSPs will optimally have both low criticality/high substitutability offerings to gain clients and high criticality/low substitutability offering to lock-in customers as found in Chapters 4 and 5. As they scale customers from low criticality solutions to high criticality solutions, lock-in is increased. However, customers are not blind to the concept of lock-in or substitutability, and ASPs may have to proceed more cautiously than ERPs did because customers may feel they were taken advantage of in previous enterprise implementations, and ASPs are in a financially weak position.

EV ASPs gain the most traction in markets that have 'special' needs. Healthcare and governments feel, maybe rightly so, that they are so different from other organizations that they require solutions aimed at them alone [Brown, July 2000; McCabe, Feb. 2001]. This is because most software that has been designed for business in general requires an extra amount of specification for them. This is due to the fact that they are different from other organizations and have a different set of criterion for success than most for profit businesses. The Enterprise Support Service providers and Full VSPs will come into direct competition in these vertical markets, and it is unclear who will prevail.

ISVs have the most to lose in this market. If they don't move to an ASP model they will lose direct contact with most of their current customers. As customer loyalty and lock-in switches to the ASP away from the ISV, the future role for the 'brand names' in software starts to become unclear. Analysts are already complaining about mySAP and Oracle's Business Online offering only its own software. ISVs are going to have a hard time competing in the utility model that seems to be on the horizon as brand name of software diminishes in value relative to the value of the ASP's name.
5.4. Summary

This concludes Chapter 5. In this chapter we examined the factors in the client’s decision to use an ASP or not. In section 5.2 we constructed two tables which represent the second half of our ASP framework and in section 5.3 we stated the strategies ASPs should adopt as a consequence of the predictions in our model. We now move on to Chapter 6.
Chapter 6. Conclusion

The ASP model is still in its early stages of development and many of the current business models are most likely flawed [Lashinksy, 2000]. However its success is backed by most of the big names in software and customers are starting to accept the idea of outsourcing their applications in this fashion.

Arguably, current apprehension about ASPs is, in part, a function of larger concerns associated with many Internet software companies, namely the dot coms. As time goes on and some ASPs are successful, these concerns will be reduced and customers may be more willing to embrace ASPs. Dell, Microsoft, Sun, Hewlett-Packard and Oracle have entered the market lending their credibility to fledging ASP companies though partnerships and certification programs [Fonseca, Nov. 2000; Trager, Feb 2001; Tillett, Mar 2001; Burke, Nov. 2000]. ASPs will start gaining customers as dot com fears start to slip away and network improvements will eliminate many customer concerns [Mears, Feb. 2001].

This thesis set out answer the following question: What are the determinants of ASP success? To answer this question we first set about defining and framing the concepts and issues surrounding the ASP domain. We then introduced a new stratification of ASPs based on target market versus the span of the processes they support. Having done this, we created an analytic framework in an attempt to predict what business models and strategies will be successful in the application service provider market space and what shape that market will have. To do this we modeled the decision process which the client uses to decide whether or not to use an ASP and if so which ASP type to use. We concluded the following strategies maximize the likelihood of long-term success depending on whether
they are of type Full VSP or Enterprise Support Services versus types Niche VSP or Focused Solution:

Table 13. Summary Of Successful Strategies

<table>
<thead>
<tr>
<th>EV or EH</th>
<th>NH or NV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keep customization costs low.</td>
<td>Keep customization costs low by using mass customization and building expertise in the service offered.</td>
</tr>
<tr>
<td>Offer both high and low substitutable services</td>
<td>Foster relations with aggregators that EV and EH are predicted to become</td>
</tr>
<tr>
<td>Mix and match best of breed software</td>
<td>Stay focused on core knowledge</td>
</tr>
<tr>
<td>Focus on integration and functionality delivered, not of the names of the ISVs that created solutions</td>
<td>Utilise UDDI and SOAP (or similar technologies) for their connectivity standards to drive down the costs of connecting to EV and EH</td>
</tr>
</tbody>
</table>

Also as a result of our analysis, we stated that transactional costs and asset specificity in this market are very high. Hence the ASP market will take on a hierarchy structure in which clients will opt to interact with one ASP that satisfies the maximum number of IT requirements as depicted in Figure 12.

Since this thesis attempts to predict changes that have not occurred yet and market structures in a still forming market space a simple conceptual analytic framework was constructed instead of using a systematic empirical study. A conclusive test of this model and the predictions made as a result of it will, therefore, require further empirical work. Empirical studies of current ASPs and ASP customers to determine reasons for buying service as well as benefits realized could also further research in this field greatly. Furthermore, there is a great deal of analysis that can be done on the interactions of the variables used within the ASP framework that were considered beyond the scope of this document.
Bibliography


Appendix A  List Of Companies

Accenture: A provider of consulting services and systems integration services

AdWare Systems: A Full VSP (Vertical Service Provider) focused on the advertisement and marketing industry

Agiliti: ASP of type ESS (Enterprise Support Services) and an application aggregator

Applicast: An ESS type ASP

Ariba: ISV (Independent Software Vendor) focused on providing supply chain business solutions

AT&T: Large telecommunications company

Bluetrain: ASP of type ESS and an application aggregator

Breakaway: an ESS type ASP

Cap Gemini Ernst & Young: A provider of management consulting services, systems integration, and technology development

Citrix: Provider of software for porting existing software to the Internet

Concentric: an ESS type ASP

Corio: An ESS type ASP.

Data return: Provider of data centre services

Digex: Provider of data centre services, owned by WorldCom

eBay: Online auction company. Recently announced adoption of .NET platform thus making it a Focused Solution ASP

Exodus: Network management service provider

FutureLink: An ESS type ASP

GovHost.com: Full VSP targeting governments

Great Plains: ISV of business software recently acquired by Microsoft

IBM: Hardware manufacturer and ISV.
ICSA.net: International Computer Security Association. Researches and consults on security issues related to computing

Intacct: Niche VSP providing accounting application services

Intel: Worlds largest manufacturer of computer processor chips

Interliant: An ESS type ASP

International Data Corporation (IDC): A provider of consulting services and IT industry research

Interpath: An aggregator ASP of type ESS

J.D. Edwards: Consulting company and ISV

Jamcracker: ASP of type ESS and an application aggregator

KPMG: Consulting and integration services company

Microage: ASP service reseller

Microsoft: Large ISV that provides software for the personal computer market. Recently released .NET framework for the development of Internet native applications

MIRUS: Full VSP targeting multi-unit restaurant chains

MySAP.com: Online extension of SAP, ESS type ASP

Oblicore: ASP Offering SLA monitoring services

Oracle: Large ISV of database management software of the same name. Has ASP division called 'Oracle Business Online', type ESS

Pandesic: Joint venture between Intel and SAP. Provided e-Commerce services, went out of business in July of 2000. Niche VSP type ASP

Peoplesoft: Large provider of ERP systems

Peoplesoft Online: Online extension of Peoplesoft, ESS type ASP

Portera: Developer and ASP for subscription based software for professional services industries (consulting firms, creative agencies and other professional services organizations), Niche VSP type ASP

Qwest communications: Large telecommunications company
Qwest Cyber.Solutions: Owned by KPMG and Qwest. Qwest Cyber.Solutions is an ESS ASP

SAP: Largest provider of ERP systems.

Sprint: Large telecommunications company and ISP

Sun Systems: Manufacturer of Sun computers, and an ISV for UNIX based computer systems and competitor to Microsoft and the personal computer market

USINTERNETWORKING (USI): An early entrant in the ASP market. Qualifies as a both a Focused Solution (in the areas of finance, accounting and supply chain) because of its use of .NET, as a Niche VSP and ESS

UUNet: Provider of data centre services, owned by WorldCom

Viasoft: ASP service reseller

WWW Consortium (W3C): Develops interoperable technologies (specifications, guidelines, software, and tools) for the Internet

WorldCom: Large telecommunications company and ISP
Appendix B  List Of Abbreviations Used

(Brief descriptions are given where necessary)

CLEC : Competitive Local Exchange Companies (common term in telecommunications industry applied to local telephone companies that resulted from the deregulation of the telephone industry common in the 1990s)

COM : Component Object Model (programming model that Microsoft uses for development)

CRM : Customer Relationship Management

DBMS : DataBase Management System

DCOM: Distributed COM (see COM above, a distributed version of the component object model used by Microsoft. DCOM is mainly used in distributed, multi-tier applications and databases)

DHTML: Dynamic HTML (an extended version of HTML that allows for the use of the CPU of the client computer. Using DHTML, the client computer is used for more than just presentation, which is the case with HTML)

ERP : Enterprise Resource Planning

ESP : Emerging Service Provider (term coined by Eastern Management Group and applied to Competitive Local Exchange Companies (CLECs), providing multiservice access services (data, voice and wireless hardware installation and service) and integrated providers such as in-Building service providers)

ESS : Enterprise Support Service

HTML : HyperText Mark-up Language

HTTP : HyperText Transport Protocol
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IPO</td>
<td>Initial Public Offering</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>ISV</td>
<td>Independent Software Vendor</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>MLPS</td>
<td>Multi-Label Protocol Switching (Method proposed for assigning priorities to data packets. MLPS would allow for various QoS on the Internet)</td>
</tr>
<tr>
<td>MRP</td>
<td>Material Resource Planning</td>
</tr>
<tr>
<td>NPV</td>
<td>Net Present Value</td>
</tr>
<tr>
<td>OLAP</td>
<td>OnLine Analytical Processing</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote Procedure Call</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol (New connectivity standard for services via the Internet, proposed by Ariba, Microsoft and IBM.)</td>
</tr>
<tr>
<td>TC</td>
<td>Transactional Costs</td>
</tr>
<tr>
<td>TCO</td>
<td>Total Cost of Ownership</td>
</tr>
<tr>
<td>UDDI</td>
<td>Universal Description Discovery and Interface (New connectivity standard proposed by Ariba, Microsoft and IBM.)</td>
</tr>
<tr>
<td>VAR</td>
<td>Value Added Reseller</td>
</tr>
<tr>
<td>VC</td>
<td>Venture Capitalist</td>
</tr>
</tbody>
</table>
VSP : Vertical Service Provider

W3C : WWW Consortium

XML : eXtensible Mark-up Language (a version of HTML that allows for the creation of data tags by the user. Often proposed as a replacement for electronic data interchange)