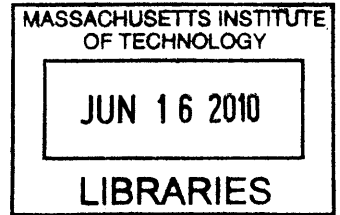


**A Strategic Framework Using Open Innovation and Platforms
to Embrace Disruptive "Software as a Service" Technology**

By

Matthew Flaherty
B.S. Syracuse University



Submitted to the System Design and Management Program
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Engineering and Management

ARCHIVES

at the


Massachusetts Institute of Technology

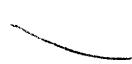
January 2010

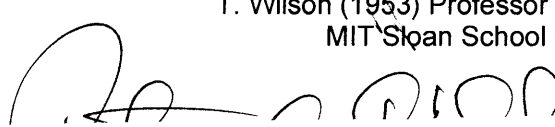
[February 2010]

© 2010 Matthew Flaherty. All rights reserved

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Signature of Author  1/14/2010
Matthew Flaherty, Fellow
System Design and Management Program

Certified by  1-14-2010
Eric A. von Hippel, Thesis Supervisor
T. Wilson (1953) Professor in Management
MIT Sloan School of Management

Accepted by  1-14-2010
Patrick Hale, Director
System Design and Management Program

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgements

Many people in my life have been tremendously supportive as I worked my way through the SDM degree program and this thesis work. Without their support and guidance, this experience would not have been possible.

I would first like to thank several IBM employees who made my SDM experience possible. First is Brent Peters, who initially sponsored my application to the SDM program. Next is Albert Silliman, who was my direct manager as I juggled the challenges of working full time while participating in the program. I'd also like to thank my current manager, Charles Hill, whose insights as the CTO of Lotus have been invaluable to this thesis work. Finally, I'd like to thank everyone whose brains I've picked over the duration of my experience, from the IBM Eclipse team in Ottawa to the IBM Lotus teams in Westford and Cambridge, MA.

I'd also like to thank the MIT staff members who make the SDM program possible, especially Pat Hale and everyone who supports him in the E-40 offices. I'd also like to thank Michael Cusumano, Michael Davies and Irving Wladawsky-Berger for their deeply relevant subject expertise. Most importantly, I especially thank Eric von Hippel for being a patient and supportive thesis advisor throughout the authoring process of this work.

Finally, I want to thank my wife Lindsay – whose incredible work ethic and dedication to learning was an inspiration as we both pursued advanced degrees as part time students. Without her support I don't know that I'd have struck out on this path, let alone seen it to completion.

Motivation

I have been employed in the software industry as a member of the IBM Lotus brand for over 10 years, working as a programmer, architect and team lead on security for several of our key products. Over the years in that role, I have developed a sincere appreciation for the application of collaboration technology in enterprises. I've also become acutely aware of the challenges posed by several trends in our industry; ranging from the business model shifts brought upon by Software-as-a-Service delivery models, to the shifting purchasing trends driven by the growing adoption of collaboration tools into everyday consumer life.

As part of my experience at IBM Lotus, I have also had the opportunity to become a committer on the Eclipse platform – the Open Source basis for Lotus' flagship Notes product. Working on Eclipse, I became very interested in open source and its significance as a development methodology and as a business strategy. I found Professor Eric von Hippel's research on lead users and open innovation an eye opening testament that the phenomenon I was witnessing was not limited to my industry.

I strongly believe that open innovation has a lot to offer in the collaboration space, and that applied correctly as strategy it can help to drive revenue in a competitive market. In order to assess a strategy that embraces openness, an understanding of the dynamics of the collaboration market must be developed, as well as a framework to evaluate strategy in the face of those dynamics.

Abstract

Over the past several decades, technology has become fundamental to the facilitation of communication, collaboration and productivity inside and between enterprises. Enterprises use numerous tools to reach their customers, manage increasingly decentralized and mobile workforces and to create digital assets critical to their daily operations.

In the last several years, changes in the availability of internet access and the compatibility of internet browsers has resulted in massively scalable services available on the internet – delivered by models termed “Software as a Service” and “Cloud Computing”. This delivery mechanism is vastly different from traditional models of enterprise software delivery where enterprise purchase, install and manage their own enterprise software packages.

This thesis will evaluate a strategy for one of the market leaders in messaging, IBM Lotus, in the face of the disruptive forces of new internet enabled delivery mechanisms like Software as a Service and Cloud Computing. In doing so, it will integrate the topics of several researchers in the field of strategy and innovation.

After a treatment of background topics and themes, it will present an evaluation of the enterprise software market in the face of the disruptive forces created by the internet. A framework for evaluating market strategies for established players will be developed using concepts of software platforms and open innovation. Finally, a case study of the established player will be viewed through the lens of this framework.

Table of Contents

Acknowledgements	3
Motivation.....	4
Abstract	5
Table of Contents.....	6
Table of Figures	9
1- Software as a Service and Cloud Computing	11
1.1- Key Terms and Definitions	12
1.2- Evolution of Enterprise Software Delivery	14
1.3- Benefits of Service-based Software Delivery.....	16
1.4- Challenges for Service-based Software Delivery.....	19
1.5- Architecture for Service-based Delivery	22
1.6- Market Trends in Service-based Delivery.....	26
1.7- Conclusions	29
2- Evaluating SaaS as a Disruptive Innovation.....	31
2.1- Research on Disruptive Innovation.....	32
2.1.1- Previous Models of Innovation	32
2.1.2- Overview of Disruptive Innovation.....	34
2.1.3- Lifecycle of a Disruptive Innovation.....	39
2.2- Evaluating Disruption in the Enterprise Software Market.....	42
2.2.1- Purchasing Behaviours	42
2.2.2- Performance Characteristics.....	44
2.2.3- SaaS in the Consumer Market	45
2.2.4- Technological Architecture of SaaS Disruption	46
2.2.5- Disruption of the Enterprise Software Business Model.....	47
2.2.6- Conclusions	49

3- A Strategic Framework for Embracing SaaS	51
3.1- Research on Technology ‘Platforms’	52
3.1.1- Platform Leaders, Wannabes and Complementors	53
3.1.2- Benefits of Technology Platforms.....	53
3.1.3- Challenges to Becoming a Platform Leader	53
3.1.4- Choosing a Platform versus a Product Strategy.....	54
3.1.5- Strategies for a Platform Leader Wannabe	54
3.1.6- Maintaining Platform Leadership.....	55
3.1.7- Conclusions	57
3.2- Research on Open Innovation.....	58
3.2.1- Lead Users	58
3.2.2- Free Revealing	61
3.2.3- Innovation Communities	62
3.2.4- Toolkits	66
3.2.5- Conclusions	67
3.3- A Strategic Framework Using Platforms and Open Innovation	68
3.3.1- Context and Assumptions	70
3.3.2- Strategies for Embracing Disruption.....	71
3.3.3- Vision Statement.....	72
3.3.4- Strategic Goals	76
3.3.5- Conclusions	88
4- Case Study: IBM Lotus Notes/Domino.....	90
4.1 - Lotus Notes the ‘Groupware’ Platform	91
4.2 - Lotus Notes the E-Mail platform.....	94
4.3 - Complementary Technologies	95
4.4 - Disruption in the Enterprise Messaging Market	97
4.5 - Strategic Framework for IBM Lotus Notes & Domino	99

4.6 - Strategic Goals	99
4.6.1 - Market related goals	100
4.6.2 - Product development goals	104
4.6.3 - Organizational related goals	108
5.4 Conclusions.....	109
5- Conclusion	111
Bibliography	114

Table of Figures

Figure 1.1 - Landscape of Cloud-based Models.....	11
Figure 1.2 – On-premise Delivery.....	14
Figure 1.3– ASP Delivery.....	14
Figure 1.4– SaaS Delivery	15
Figure 1.5 - SaaS Architectural Models.....	24
Figure 1.6 - Reference Architecture for Multitenancy	24
Figure 1.7 – Worldwide SaaS Revenues.....	26
Figure 1.8 – SaaS Adoption by Business Size	26
Figure 1.9 – SaaS Adoption by Application Niche	27
Figure 1.10 – SaaS Adoption Criteria in the CRM Market	28
Figure 2.1 – Architectural Innovation Model	33
Figure 2.2 – Traditional S-Curve Model of Innovation.....	34
Figure 2.3 – Sustaining Trajectory of Recording Density	35
Figure 2.4 – Nested Model of Product Architecture	36
Figure 2.5 – Three Examples of Value Networks	37
Figure 2.6 – Disruptive Technology Model of Innovation	38
Figure 2.7 – Investment in Business Process Automation.....	43
Figure 3.1 – Platform Strategy for Software Products.....	52
Figure 3.2 – Quantitative Study of User Innovation	59
Figure 3.3 – Quadrants of Network Innovation Model.....	64
Figure 4.1 – Lotus Notes Version 1.0.....	92
Figure 4.2 – Lotus Notes Version 8.5.....	94

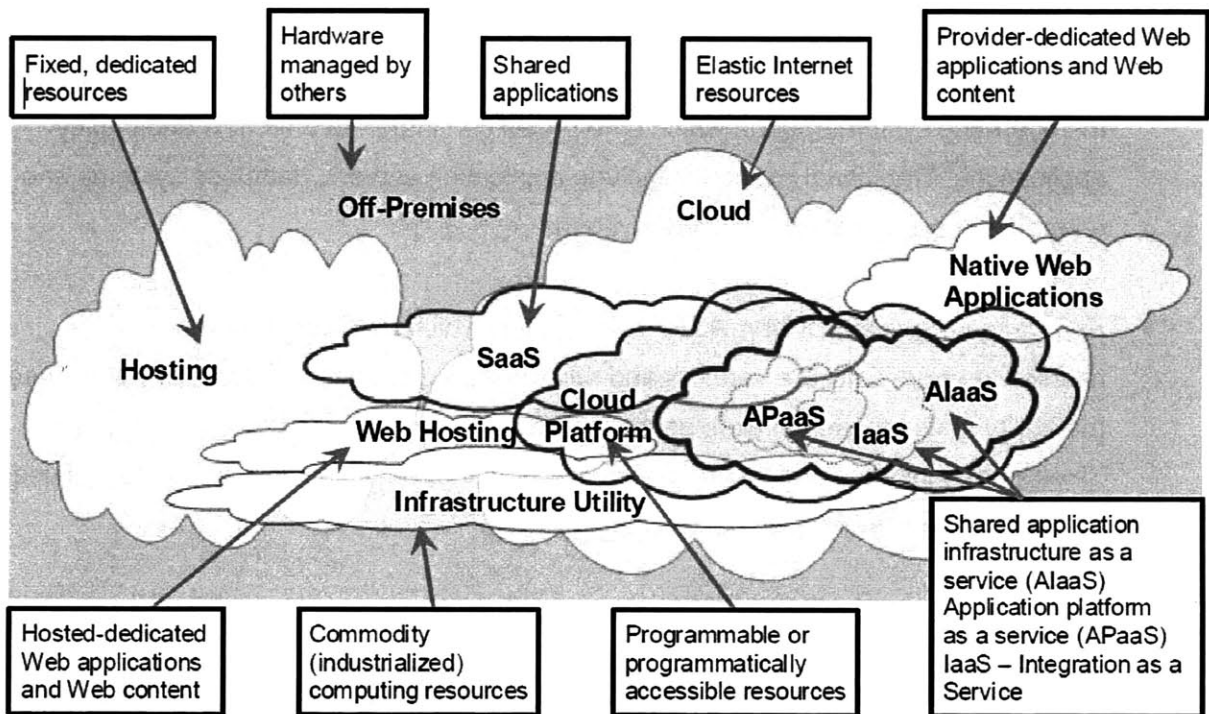
THIS PAGE INTENTIONALLY LEFT BLANK

1- Software as a Service and Cloud Computing

One of the most talked about trends in the enterprise software business is the emergence of “Cloud Computing” and the many variations on “Software as a Service” delivery. Although clearly influenced by marketing departments, these terms represent interesting changes in the way that software is built, purchased and used. This section will present an overview of the trends in Cloud Computing and Software as a Service. It will define the terms, present some historical trends in software delivery, address the market trends and changes that are presently occurring and close by discussing the opportunities and challenges that the trends represent for both customers and vendors of enterprise software.

A visual representation of the major concepts and overlap from Gartner is presented in Figure 1.1:

Figure 1.1 - Landscape of Cloud-based Models



Size of the cloudlets and overlap shown are not to scale

Source: Gartner (April 2009)

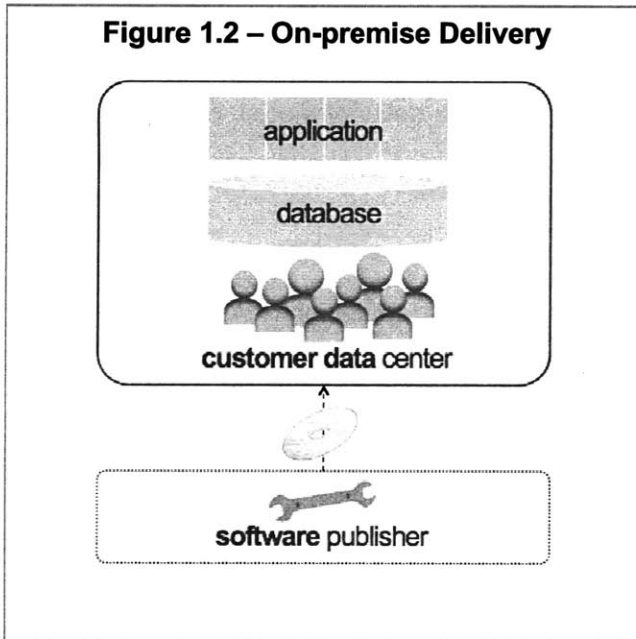
1.1- Key Terms and Definitions

Driven from research from Gartner and other analyst firms, there are several key terms and definitions that will help to understand the landscape of Cloud Computing and Software as a Service based delivery methods (Desisto, Plummer, & Smith, Tutorial for Understanding the Relationship Between Cloud Computing and SaaS, 2008) (Natis, et al., 2008):

- *Enterprise Software Vendor* - the companies that create and sell the software platforms and applications that are traditionally consumed by enterprises. Examples include Microsoft, IBM, Oracle, Google, and others.
- *On-premise Software* – traditional method of running enterprise software by purchasing licenses and support, install and running the software offering in the enterprises' own data center. As a result of the owned data center, this incurs capital costs such as software and hardware, as well as real estate, power, and personnel.
- *Middleware* – a type of on-premise software which provides an abstraction layer above the operating system(s) upon which enterprises can integrate their new and legacy applications. Traditional examples include application servers, database systems and enterprise messaging and workflow systems.
- *Application Service Provider (ASP)* – companies which purchase software and middleware from software vendors and run it in their own data centers. They then resell the ability to use these software services to enterprises, traditionally as “hosted” (or “fixed”) offerings on a one-to-one basis, meaning that each software installation is used by exactly one customer.
- *Software as a Service (SaaS)* - a software offering that is owned, operated and delivered by one or more providers. The services that this software offers are consumed in a one-to-many model, where many customers are serviced by the same software installation, implying that the software is explicitly built (or configured) to support multiple customers at once. Pricing models can be pay-for-use or subscription based on a temporal billing cycle.

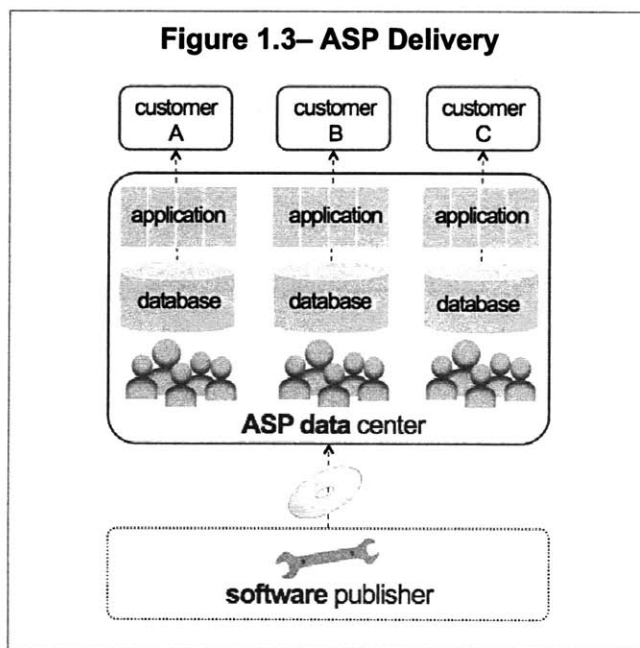
- *SaaS Provider* - a company that offers SaaS services, billed to customers on a subscription based contract.
- *Application Infrastructure as a Service (AlaaS)* – AlaaS is analogous to middleware except for that it is implemented explicitly for a service delivery model. SaaS providers offer AlaaS solutions upon which enterprises can develop, integrate and host enterprise applications in an elastic, multitenant environment.
- *Application Platform as a Service (APaaS)* – a narrower definition of AlaaS, roughly analogous to an application server in a service delivery model.
- *Infrastructure as a Service (IaaS)* – another narrower definition of AlaaS, focused on the integration of multiple tenants in a SaaS environment. Typical uses are as a shared environment for the constituents of a value chain (suppliers, partners, etc) to work together.
- *“Hybrid” SaaS* – A particular configuration of software architecture that mixes both service-based delivery of function and on-premise software deployment. This requires an interoperability strategy for the on-premise software in question.
- *Cloud Computing* – a software delivery model whereby software services are delivered to enterprises via Internet technologies in a massively scalable fashion. As with SaaS, Cloud Computing is offered on pay-for-use or a subscription based temporal billing cycle.
- *Cloud Service Provider* – a company which offers Cloud Computing Services.

1.2- Evolution of Enterprise Software Delivery



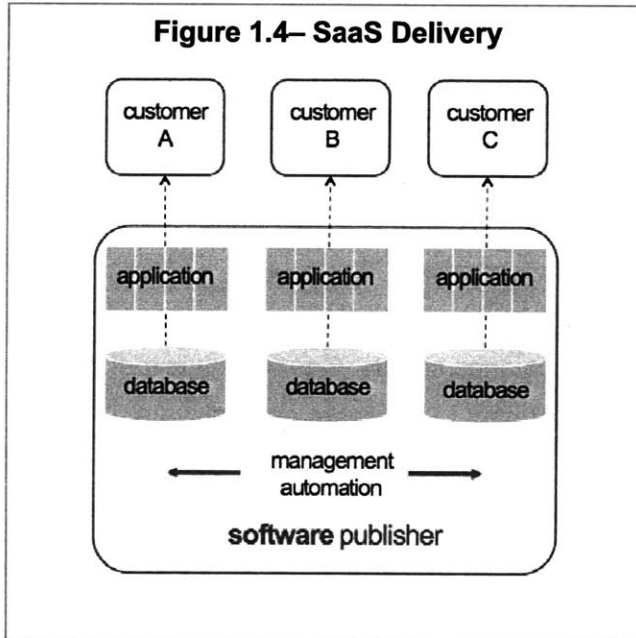
Traditional enterprise software acquisition before the arrival of the internet was based on the purchase of per-user licenses of operating systems and applications, often paired with leased hardware purchases and an associated support contract from an enterprise application vendor. The vendor would then work with the enterprise to install, configure and support the installation of their software onto supported hardware systems. The enterprise then entered into a long term relationship with the vendor where they would receive

upgrades, often at costs in multitudes of the initial purchase. These up-front costs presented a challenge: financially, enterprise technology had become an enormous capital expense, and this had many follow on effects – especially version lock and the inability to preserve changes through an upgrade (Service-now.com, 2009).



In the late 90's, Application Service Providers emerged as a way for enterprises to lease the software capability from a provider who would handle the costs associated with the upfront acquisition and maintenance of software and hardware from enterprise application vendors. ASPs sought to defray these costs by supporting multiple customers. This business model was not a strong success – due to the complexity and variety of customer deployments, client platforms and protocols, combined with the fact that software from enterprise vendors was not

built explicitly with resource sharing in mind.



As the last decade has passed, the technology by which users access enterprise technology - the familiar internet browser - has become ubiquitous, standardized and extremely capable at delivering increasingly rich user experiences. The network itself is also nearly ubiquitous, and accessible from a multitude of devices – desktops, laptops, mobile phones and more. Companies which have been born into this new world have benefited from a number of technological advancements – particularly

from the realization of architectures which can support multiple customers by their nature, supported by development in technologies like virtualization. These vendors do not simply develop and sell their software; instead they develop it and then make it available through their own data centers over the internet, often through a standard web browser which is more easily installed, deployed and maintained on enterprises computing platforms.

1.3- Benefits of Service-based Software Delivery

For Enterprise Customers – Service based software delivery presents a number of benefits to the customer, and the needs of the customer are the major driver for the increasing adoption of SaaS and Cloud based software models. The following are a collection of some of the perceived benefits of a shift to SaaS and Cloud based service delivery (Desisto & Pring, Essential SaaS Overview and Guide to SaaS Research, 2008) (Pring, Desisto, & Bona, September 28, 2007):

- *Lower initial investment* - The most widely recognized benefit for customers is the change that SaaS purchasing makes on the customers balance sheets. One benefit is that it removes the need to make an initial capital investment in hardware and software.
- *Lower recurring costs* – Aside from the initial costs, on-premise software requires a host of skills built into the enterprise – from management of customer data center hardware and operating systems, to database and application development and management personnel. SaaS and Cloud delivery shifts these functions to the provider, and explicitly strives to makes the product customization and management simpler less of a specialized skill.
- *Usage-based billing* - SaaS and Cloud software is typically billed as a subscription contract, based on usage metrics. This gives the customer the ability to pay for exactly the amount of application usage that they consume. When applied to the right types of application, this can result in lower overall cost of delivering IT services. In particular, the elastic nature of SaaS and Cloud platforms tends to give better results when resource usage is
- *Faster time to value* – A browser is a key and ubiquitous piece of software on a desktop operating system. Since most SaaS and Cloud based software is adequately delivered over an internet browser platform, there is less need for a wide deployment of alternate software, allowing enterprise customers to quickly
- *Higher Availability* – Because the core business of providers of SaaS and Cloud based software requires them to keep their software up and running for all users, availability is a key differentiating value. This is enforced through Service Level Agreements (SLAs) that mandate an acceptable level of service quality.

- *Regular Updates* – Vendors have the ability to roll out changes to all users of the system, or select users, at any time. This allows enterprise customers faster access to new features updates, increasing the amount of value and its delivery time. Vendors can also deliver patches in a timely fashion, increasing metrics like performance, reliability and stability.
- *Standards-based customization* – As discussed in previous sections, the evolution of standards and the browsers' implementations of them has enabled more advanced and valuable experiences to be delivered over browser based technologies. Many customizations of SaaS and Cloud platforms are now being done through familiar programming models like REST, and familiar programming languages like HTML and Javascript. A side effect is that the talent pool of development resources available to work on these technologies is very high.

For Software Vendors – SaaS delivery presents mentionable benefits to the vendors of existing on-premise software platforms and applications who are making the transition to SaaS and Cloud based delivery. SaaS service providers share these same benefits, and are in a particularly strong position to deliver on the value of SaaS and Cloud based services. This will be covered in more detail in the following chapters, but some of the most important benefits are as follows (Desisto & Pring, Essential SaaS Overview and Guide to SaaS Research, 2008) (Pring, Desisto, & Bona, September 28, 2007):

- *Predictable Revenue Stream* – Moving to a subscription basis also provides financial benefits for the vendors of SaaS and Cloud based services. The revenue associated with license based software takes place in bursts, initial sales have large revenue impact and license renewals take place at unpredictable intervals. This means that
- *Architectural Simplification* - One important benefit is to mitigate the integration and testing challenge that is present with on-premise deployments. SaaS and Cloud based deployments can be tailored to a smaller number of configurations, allowing for smaller test matrices and less cost in the form of testing personnel for the provider.
- *Better User Feedback* - The ability to immediately deliver a set of function to a broad set of users allows vendors to get immediate feedback from a large installed user base. This

could be anything from simple collection of usage statistics, or more collaborative feature evaluation and design.

- *Potential Network Effects* – depending on the business model, having many customers on one SaaS or Cloud platform creates the opportunity for a number of network effects between customers, or between customers and Independent Software Vendors (ISVs).

1.4- Challenges for Service-based Software Delivery

For Enterprise Customers – Delivery of software over the internet via a subscription based method is not without its challenges, both for enterprises adopting SaaS and Cloud based technology and for vendors creating it follows (Desisto & Pring, Essential SaaS Overview and Guide to SaaS Research, 2008) (Pring, Desisto, & Bona, September 28, 2007):

- *Lessened Competitive Advantage* - Every enterprise who acquires a SaaS application in general shares the same capabilities of every other enterprise who takes it on. Customization and integration is usually possible, but not yet to the depth possible with on-premise software. This means that companies who select SaaS solutions do not have competitive advantage through these SaaS based IT systems (modulo human factors).
- *Pricing Complexity* – Most SaaS platforms are not actually offered with true utility pricing – typically an upfront capacity decision is made and prices are rarely readjusted on metered usage and unused capacity is rarely credited. This can mean that buyers are not making the most efficient use of their investment.
- *Loss of Firm's Control* – As customers shift their data to be stored in the datacenters of SaaS and Cloud providers, there is a loss of control over that data. Although SLAs provide a way to manage this relationship, customers must be careful to think about the ramifications on function such as scheduled backups and other administrative tasks.
- *Loss of IT Department Control* – Enterprises spend significant effort building IT departments who can manage and scale the IT systems that are used in an enterprise. Since SaaS pricing is often per-user and relatively inexpensive and purchasing is simple and easy; buyers may not necessarily be in the IT department. This can create later problems and cost for the IT departments, such as migration costs, data governance issues and others.
- *Security and Privacy* – Security is often cited as a prime concern for customers of SaaS and Cloud services, as their data has left their own datacenter and moved to a cloud service provider's site. Customers need assurances that the software which runs the SaaS services is hacker proof, and that measures are being taken to keep their users and their

data safe. Security related concerns also follow to privacy, as multitenant solutions need to carefully keep customer's data and users separate.

- *Network Dependence* – SaaS and Cloud based platforms are delivered over the internet, thus the customer must have an always on connection to the vendor's datacenter. This can present difficulties for mobile users, or when network connectivity is interrupted.
- *Vendor Viability* – Customers must be cautious when selecting a vendor to provide a SaaS or Cloud service. Up front they must consider the data format and agreement in place should a vendor go out of business or otherwise be unable to deliver their service to the customer.

For Software Vendors – Vendors also face a set of challenges as they deliver SaaS and Cloud based software. The following are some of the better known challenges (Desisto & Pring, Essential SaaS Overview and Guide to SaaS Research, 2008) (Pring, Desisto, & Bona, September 28, 2007):

- *Capital Investment* – In addition to the initial R&D costs to develop the infrastructure that provides a SaaS platform's function, providers must also make the capital investments required to host the SaaS and Cloud based software offering. This includes real estate, hardware and potentially software capital investment.
- *Lower revenue potential* – Traditional software vendors have been able to operate at high margins. The up-front cost of developing on-premise software is low compared to their ability to print, charge for and distribute copies. SaaS and Cloud margins are lower, as there is a recurring cost to running the infrastructure. As will be articulated later, this presents a challenge for established players, while new players have optimized their architectures for this reality.
- *Technical Difficulty and Complexity* – Delivery of elastic, multitenant architectures is a non-trivial engineering effort. Enterprise Software Vendors and Cloud Service Providers will need to learn architectural patterns that best support the attributes required to deliver the types and scale of function that they wish to provide.

- *Data Governance* – A SaaS provider is in the position of needing to store their customer's data in their own datacenter. This implies that the vendor must understand and likely implement solutions for any data governance requirements which stem from regulatory requirements in the customers market.
- *Security* – SaaS applications present an appealing target to hackers. Security is also a recurring challenge. Since SaaS vendors deliver their software over the internet, they are necessarily creating a larger and more available threat surface for security vulnerabilities than if the same function was delivered on-premises.

1.5- Architecture for Service-based Delivery

As evidenced by the failure of the ASP market in the late 1990s, the delivery of software in a service oriented fashion is a non-trivial exercise. The ASPs found that the upfront cost of creating the software that runs the service must be capitalized across multiple customers in order to be sustainable. This was a challenge for ASPs, as they did not create the platforms that they deployed, and found that the customizations required by customers were costly to create and maintain. The implication is that creating and owning a majority of the software stack is critical for success. For entrants to the SaaS market, this means that they must create a software stack from scratch. For existing vendors, they must do the same or retrofit their existing offerings to be cloud based.

In order to provide the benefits and negate the challenges summarized in the previous section, there are several required features and attributes of a SaaS or Cloud based architecture (Desisto & Pring, Essential SaaS Overview and Guide to SaaS Research, 2008) (Pring, Desisto, & Bona, September 28, 2007):

- *Multitenancy* – an architectural pattern whereby an enterprise software application supports multiple customers at one time. This allows SaaS and Cloud Providers to share costs of platform development and maintenance across multiple customers.
- *Scalability* – A measure of the capacity of a SaaS or Cloud service. A scalable architecture will support many users accessing it at the same time, and can support a large amount of data stored on it.
- *Elasticity* - an attribute of a Cloud Computing architecture whereby the resources (CPU, disk, etc) allocated to support each customer can be grown or shrunk without impact to the customer's environment.
- *Metering* – Metering is a method for cloud Service Providers to measure and monitor the usage of a SaaS platform, allowing them to charge and monetize usage of features.
- *Self Management & Usability* – SaaS platforms have functions that are managed by a Customer's IT staff. The interfaces that support this usage must be as automated as

possible and usable where not automated, keeping support costs down. Similarly, the end user experiences must also be usable. Although this has also been true for on-premise software, scaling support across numerous simultaneous users makes this an important priority for a Cloud Service Provider.

In order to support these requirements in different configurations, several architectural models have emerged and are captured on the following page in Figure 1.5 and Figure 1.6 on the following page:

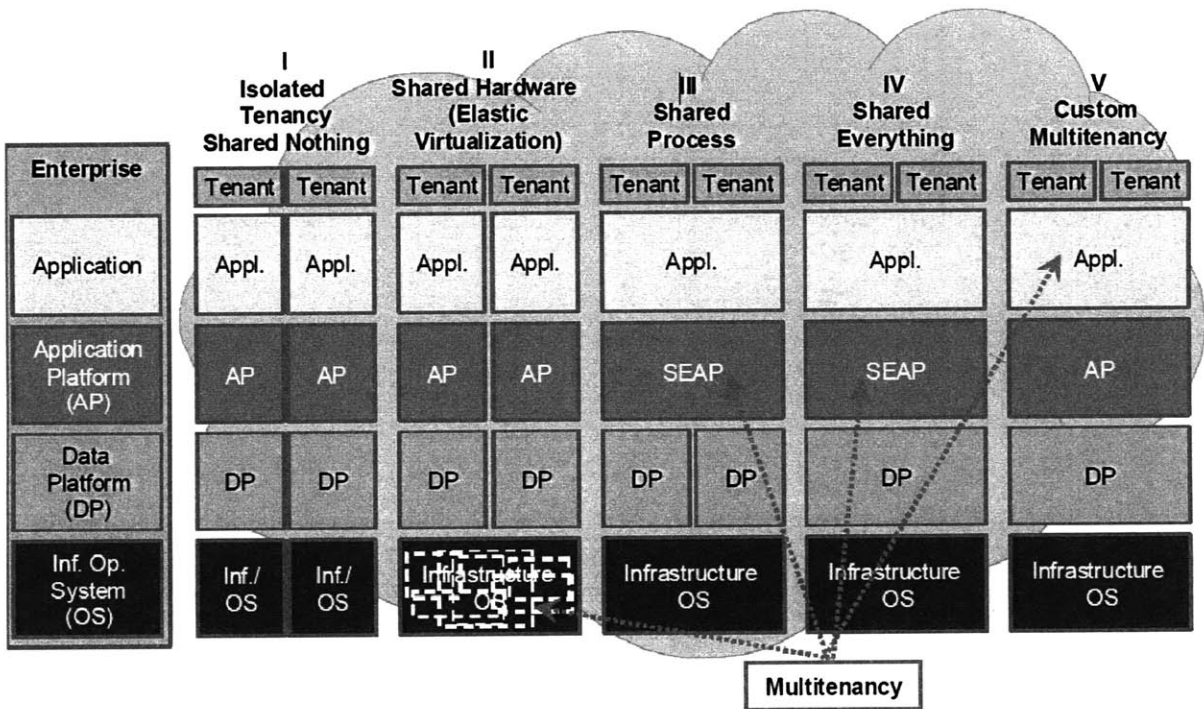
Figure 1.5 - SaaS Architectural Models

	Managed Hosting Service	Isolated Tenancy	Shared Execution	Multi-tenant/Single-Version	Multi-tenant/Multiversion
Application Execution Infrastructure	Dedicated	Dedicated	Shared	Shared	Shared
Application Versioning	Multiple	Single	Single	Single	Multiple
Data Separation	Physical	Physical	Physical	Logical	Logical

Supports SaaS

Source: Gartner (August 2006)

Figure 1.6 - Reference Architecture for Multitenancy



Source: Gartner (April 2009)

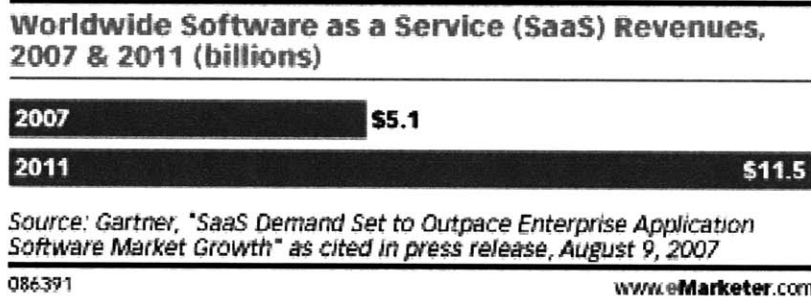
Essentially, the architecture of each SaaS or Cloud offering is divided into several layers of functionality:

- *Application & Application Platform* – the executable content of the application which provides the application logic that makes the actual functionality available. This can be implemented as a middleware layer, as in IaaS or APaaS, upon which the Cloud Service Provider builds its own applications, or allows customers and third parties to do so. Application isolation is important at this layer, especially in . This layer can also be programmable and extensible, providing an opportunity to create sticky custom applications. Amazon and Google both have offerings that are extremely programmable at this layer.
- *Data Platform* – this is the layer where customer data is stored into and retrieved from. The challenge in this layer of the architecture is scalability, as a SaaS or Cloud platform may be servicing thousands or even millions of active users. One of the approaches in this area is to build massively horizontally scalable databases, where. Google's BigTable is an example.
- *Operating System* – As with on-premise software, SaaS and Cloud platforms run on a traditional operating system. Virtualization is a key technology in the architecture at this late
- *Client Technology* – Not represented on the charts is the architecture of client technologies – be they traditional desktop, browser client or mobile. The strategy for most SaaS and Cloud technologies is to deliver HTML over HTTP, so that the broadest segment of client platforms can be satisfied. The challenge is to overcome some of the client-related inadequacies in the challenges section, such as network dependence (with offline) and security (with enhanced security function).

1.6- Market Trends in Service-based Delivery

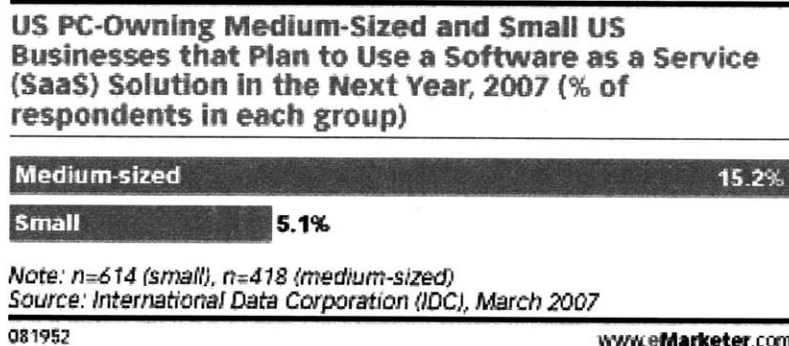
The first SaaS offerings started appearing in the 1990s in niche markets, and now have a strong foothold in a several markets. Gartner's recent report on the state of the SaaS market predicts that the market will see a 19.8% compound annual growth rate through 2013, which more than triples the overall market CAGR of 5.2% (Mertz, 2009). Growth in revenue in the overall segment is strong, as evidenced by Figure 1.7:

Figure 1.7 – Worldwide SaaS Revenues



With respect to the level and rate of adoption across different types of enterprises and different types of enterprise applications, Service and Cloud based delivery models have differing levels of impact. Figure 1.8 captures the overall demand for SaaS applications in small to medium sized businesses:

Figure 1.8 – SaaS Adoption by Business Size



The demand for SaaS also varies across niches of applications, as the suitability of SaaS delivery varies widely across niches. Figure 1.9 captures a sample of the penetration of various niches of applications, as reported by a population of marketing professionals:

Figure 1.9 – SaaS Adoption by Application Niche

Areas of Business in Which Media and Online Marketing Professionals Worldwide* Use Software as a Service (SaaS), October-November 2008 (% of respondents)

	Using	Planning to use	No plans to use
E-mail	77%	5%	18%
Web hosting	74%	8%	18%
Sales and marketing	62%	16%	22%
Content management system (CMS)	59%	17%	24%
File storage/assets storage	57%	14%	29%
Search	57%	15%	29%
Ad serving	51%	15%	34%
E-commerce platform	49%	20%	31%
Financial applications	48%	14%	38%
Office automation	44%	16%	39%

*Note: n=453; numbers may not add up to 100% due to rounding; *UK (65%), North America (13%), other Europe (11%) and other (11%)*
Source: Econsultancy, "Vertical Search (B2B) Report 2009" sponsored by Convera, January 21, 2009

101495

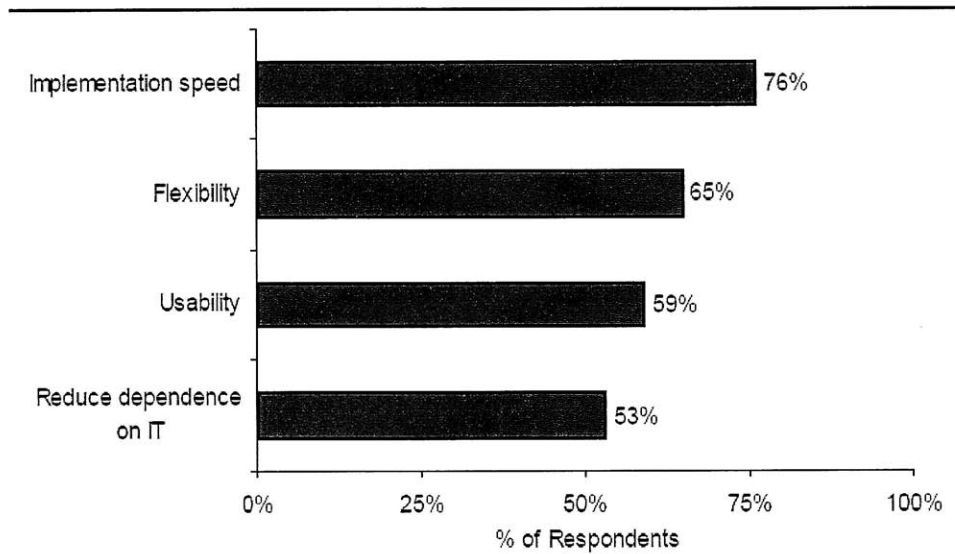
www.eMarketer.com

One of the best known SaaS applications is Salesforce.com, which is used by a company's sales force for tracking and following up on sales leads. This niche, Customer Relationship Management (CRM), is one of the most researched markets for SaaS software. In 2008, the Aberdeen group (Duhaime, 2007) surveyed a group of over 600 companies who use CRM software – they found that over 65% of those customers believed that SaaS versions of CRM software would replace the current on-premise offerings over time. The study cites the following as key factors influencing companies' decision to shift their attention to SaaS offerings:

- Expanding Global business environment
- Growing mobile sales force and work force
- Data security concerns
- Vendor longevity
- Increasing total cost of ownership of integrating front-end and backend solutions
- Need to customize to align with business model, niche market, and customer experience
- Constrained IT resources

Figure 1.10 presents some of the criteria by which customers of the CRM market have been evaluating the move to SaaS based software:

Figure 1.10 – SaaS Adoption Criteria in the CRM Market



Source: Aberdeen Group, December 2007

Overall, market demand is strong for SaaS and Cloud delivery, although there is a range of adoption that varies across company size and application niche.

1.7- Conclusions

Software as a Service and Cloud based delivery models have emerged because of a shift in enterprises' willingness to make an enormous up-front investment in some sets of IT systems – they see SaaS delivery as a way to essentially rent access to a particular IT service. At the same time, browser technology has become more powerful and internet availability has become more pervasive, and there is now a reasonable way to deliver a high-quality user experience over an internet browser to a remote customer. The commoditization of the hardware and software required to host these massive data centers and technologies like virtualization have lowered the barrier to entry for a number of smaller players to begin building their own datacenter architectures and software stacks to deliver SaaS and Cloud services.

As a vendor of traditional enterprise software, the question that must next be asked is whether this new subscription-based, internet-delivered model presents a threat to the existing business model. This question must be answered in such a way that vendors can understand the level of risk, based on attributes of their existing technology.

In the following section, the trend towards Software as a Service and Cloud delivery models will be evaluated as an instance of a “disruptive technology” relative to software vendors who are currently selling software as an installable asset based on license and support revenue.

THIS PAGE INTENTIONALLY LEFT BLANK

2- Evaluating SaaS as a Disruptive Innovation

Software as a service and cloud computing are a new delivery mechanisms that have evolved out of previous models used by Application Service Providers (ASPs). Advances in browser technologies and network availability have allowed better user experiences to be delivered to users, removing much of the need for client software besides an operating system and a browser. Much of the interaction that consumers have with the internet is through browser access, from news and information to e-commerce and more.

In the enterprise software market, the picture is not as simple. Enterprises have long used network technology, and thus the ecosystem of technologies at both the client and the server are far more diverse. Users use clients like Lotus Notes and the Microsoft Office suite to work with information, and other customized applications to do critical business processes. Broadly, this market is called the Enterprise Software market.

Understanding how Software as a Service and Cloud delivery is going to impact the Enterprise Software market is critical to existing firms' survival. This chapter will explore models of innovation and their impact on established firms' capability to endure generations of technology changes. The discussion will begin with a treatment of research in the area, particularly on the concepts of 'Disruptive Technology'. This model is one that focuses on how technologies' applicability to existing customer needs factors into a company's ability to adapt and adopt it. The second half of this chapter will evaluate the emergence of Software as a Service and Cloud delivery models as a disruptive technology relative to delivery of software by traditional vendors of Enterprise Software, such as IBM, Microsoft, and others.

2.1- Research on Disruptive Innovation

Technology moves at an incredibly fast pace. Over the years, patterns have emerged in the evolution and adoption of technology which have been explained by an area of research called 'Disruptive Technology'. The original researcher in this area is Clayton Christensen of Harvard University, whose work "The Innovators Dilemma" is required reading material for those working in technology based industries. Over the last 15 years, he has applied his research and theory to a broad range of technology oriented industries, starting from the computer Hard disk industry and continuing to others such as hydraulic machinery, flash memory, discount retailers, and steel production.

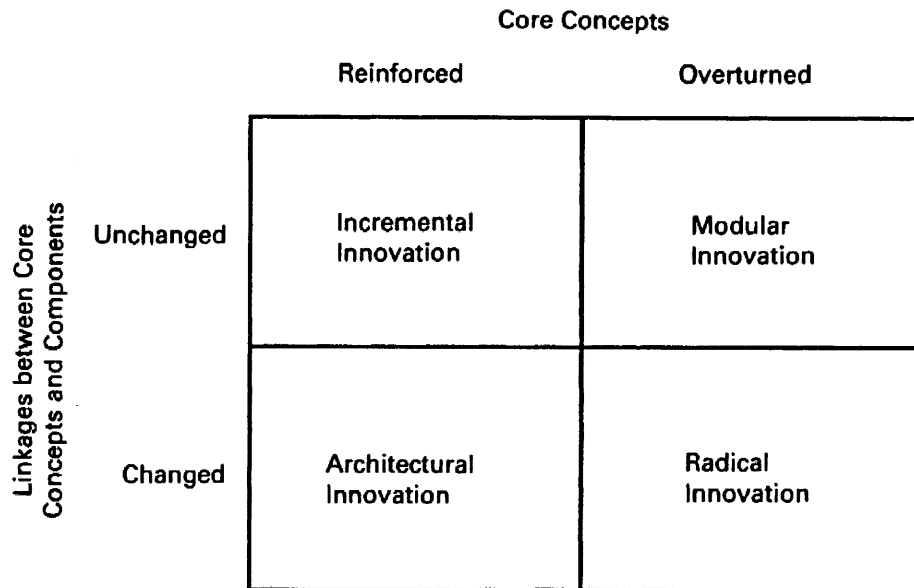
2.1.1-Previous Models of Innovation

Previous models of the failure of technology based businesses inability to survive shifts in technology generations focused on the firm's technological capabilities. Abernathy and Clark (Abernathy & Clark, 1985) defined two fundamental forms of innovation:

- *Incremental* – Innovation which increases performance around a well known axis, which conforms to a firm's existing technological and organizational capabilities.
- *Radical* – Innovation which changes the product or development process so radically that it challenges the existing skills or resources, resulting in a net negative effect on productivity. In essence, radical innovation often produces a new dominant design.

Henderson and Clark (Henderson & Clark, 1990) later refined the model of incremental and radical innovation into one that separated types of innovation further into four quadrants, as seen in Figure 2.1:

Figure 2.1 – Architectural Innovation Model



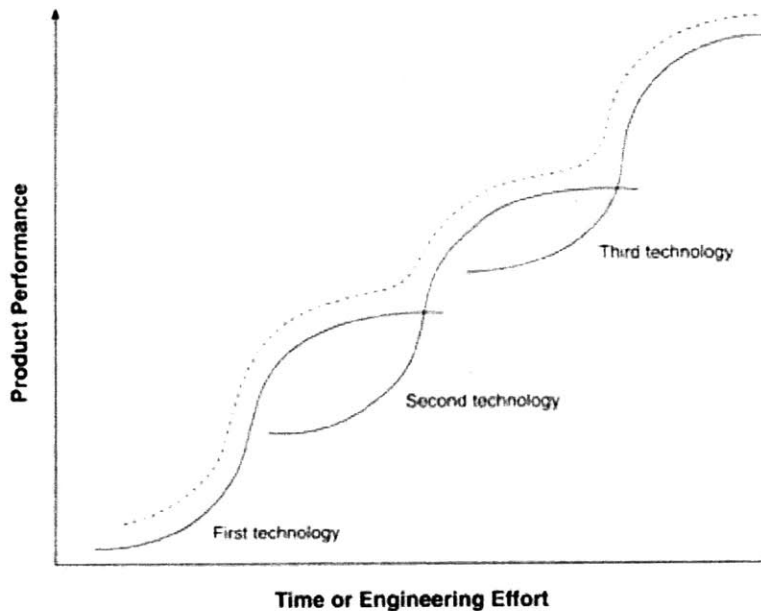
In this model, the technological capabilities of the firm are measured in a way that accommodates the different ways that system architectures are produced as a particular configuration of components. Along the vertical axis is innovation that changes the configuration of components in the architecture, while the horizontal captures innovation that changes the actual characteristics of components themselves. This means that incremental innovation does not change the components or how they are arranged, while a radical innovation changes both the architecture and the components. It also adds two additional

- *Architectural* – Innovation which rearranges components into a new architecture, requiring a firm to struggle to learn a new architectural paradigm.
- *Modular* – Innovation which changes one or more core components, and as such can be replaced without requiring changes to the overall architecture.

The bulk of innovation theory thus focuses on the capabilities of a firm and its knowledge about production and rearrangement of components and architectures into new products. Captured over time visually, the models of innovation is represented as overlapping S-Curves, where

each S-Curve is a generation of technology experiencing incremental innovation with a radical technology shift between them. This model is captured in Figure 2.2:

Figure 2.2 – Traditional S-Curve Model of Innovation



2.1.2-Overview of Disruptive Innovation

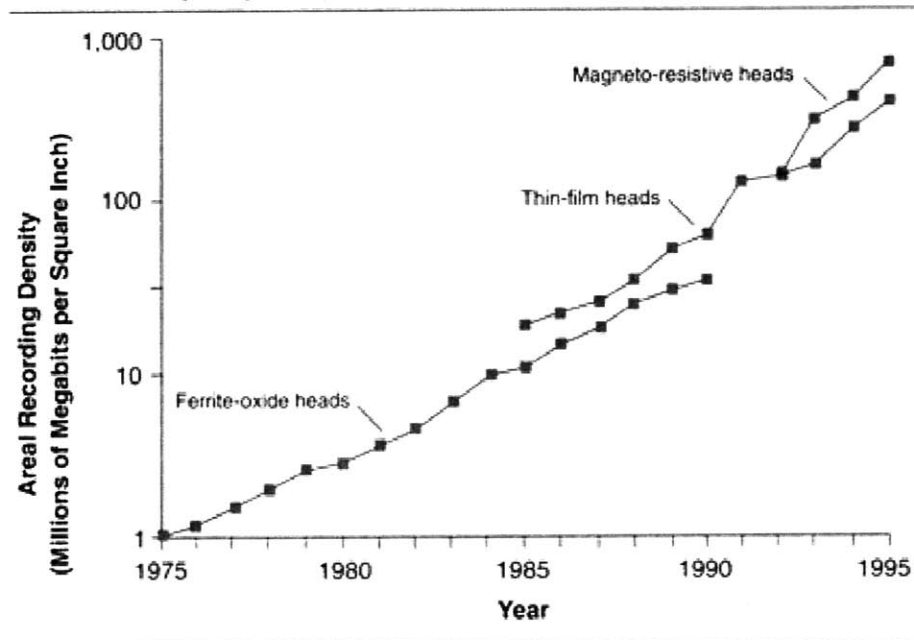
The disruptive technology model shifts from the capabilities model to one which focuses on the needs of a firm's customers. In performing his research of the hard drive industry, Christensen first considered what he called the "technology mud slide" hypothesis – a belief the rate of technological change is simply so fast that firms must struggle to stay on top of the waves of technological change. One misstep and they are swept under the flow and carried away to obsolescence. In testing this hypothesis, Christensen instead found that it was not the rate of change that destroyed firms, instead that there appear to be different types of innovation which present very different challenges to dominant firms.

The first type of innovation is termed a sustaining innovation. A sustaining innovation tends to increase the performance of the industries' products in well understood ways, often in support of established trends that firms intend to follow. Examples from Christensen's work include the incremental speed increases of Intel's microprocessors, which rose at about 20% per year; and the quality of Eli Lilly's insulin, which improved in purity at about 14% per year (Christensen

1998). In the face of a sustaining innovation, firms can make ready assessments of the implication of a technology on their performance trajectories, and work to incorporate it. The dominant firms in an industry were always the ones to adopt, commercialize and succeed with sustaining innovations - . In the working example from the hard disk drive industry, the changes in head technology that allowed the firms in the hard drive industry to pack more data onto hard disk drives are represented over time by the graph below:

Figure 2.3 – Sustaining Trajectory of Recording Density

Figure 1.4 Impact of New Read-Write Head Technologies in Sustaining the Trajectory of Improvement in Recording Density



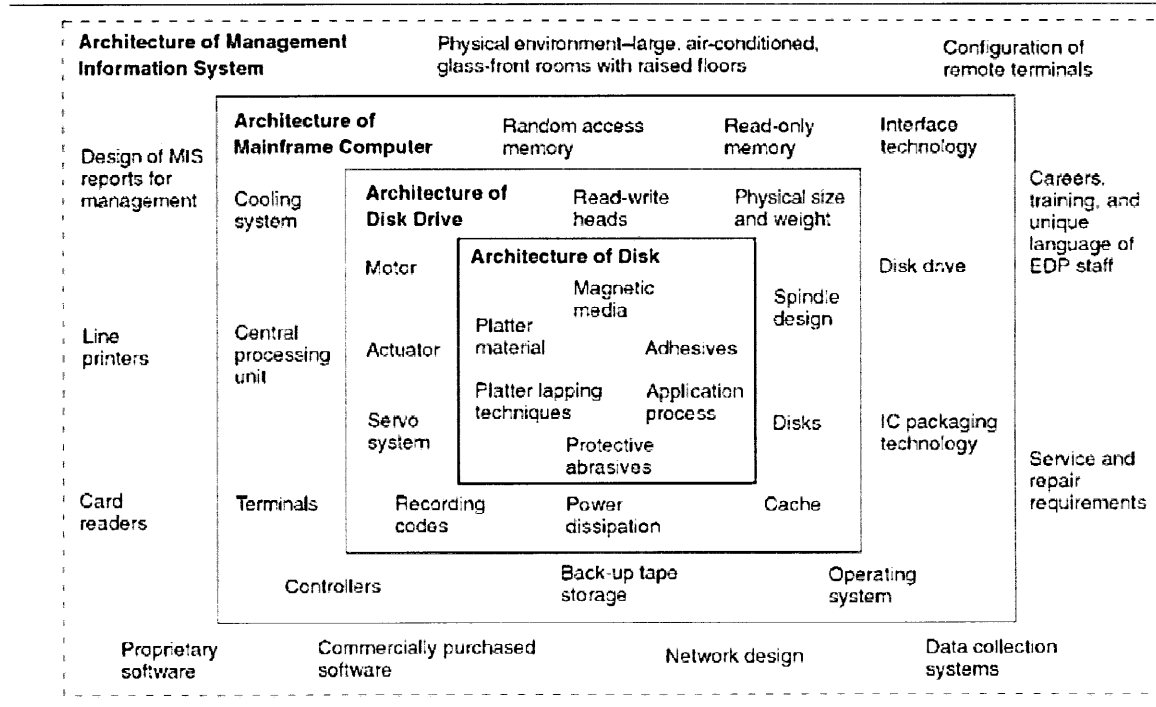
Source: Data are from various issues of *Disk/Trend Report*.

Despite the fact that each generation of technology experienced incremental innovation and even radical innovation as jumps between generations of technology, the dominant firms always remained on top when innovation was sustaining. This was true regardless of whether the innovations were incremental or radical, component or architectural, or on any other categorization. The paths of technological innovation looked like the traditional S-Curve in section 3.2.1.

The second type of innovation that Christensen identified was the disruptive innovation. The product architecture model of a value network is depicted in Figure 2.4:

Figure 2.4 – Nested Model of Product Architecture

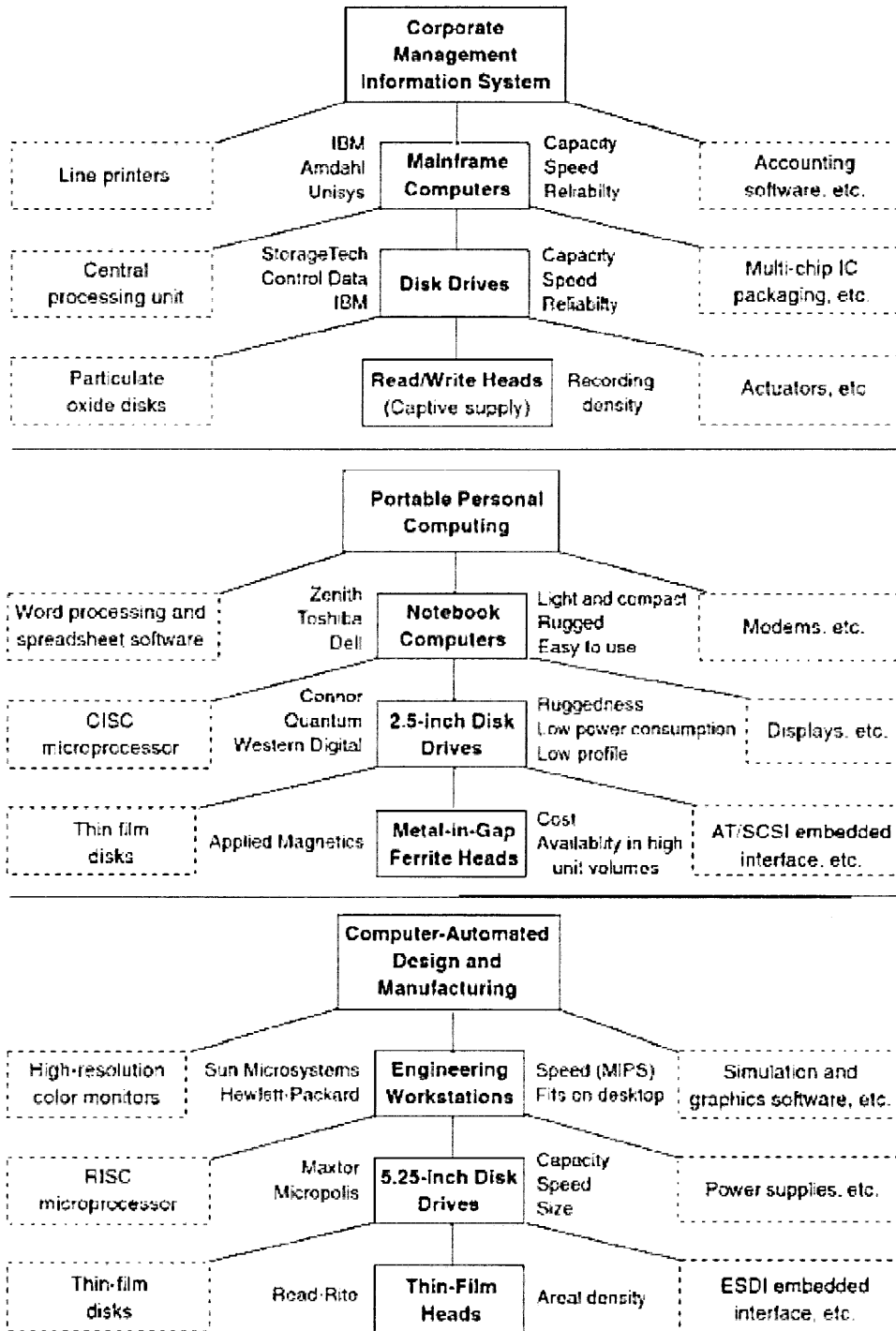
Figure 2.1 A Nested, or Telescoping, System of Product Architectures



Source: Reprinted from *Research Policy* 24, Clayton M. Christensen and Richard S. Rosenbloom, "Explaining the Attacker's Advantage: Technologic Paradigms, Organizational Dynamics, and the Value Network," 233-257, 1995 with kind permission of Elsevier Science—NL, Sara Burgerhartstraat 25, 1017 CA Amsterdam, The Netherlands

A value network is the context that a firm operates, where it assesses needs, creates solutions, and competes for profits. The components that make up a product system that meets a customer's needs may be produced by a single firm, but more often are created by a number of firms in the value network, as exemplified in Figure 2.5:

Figure 2.5 – Three Examples of Value Networks

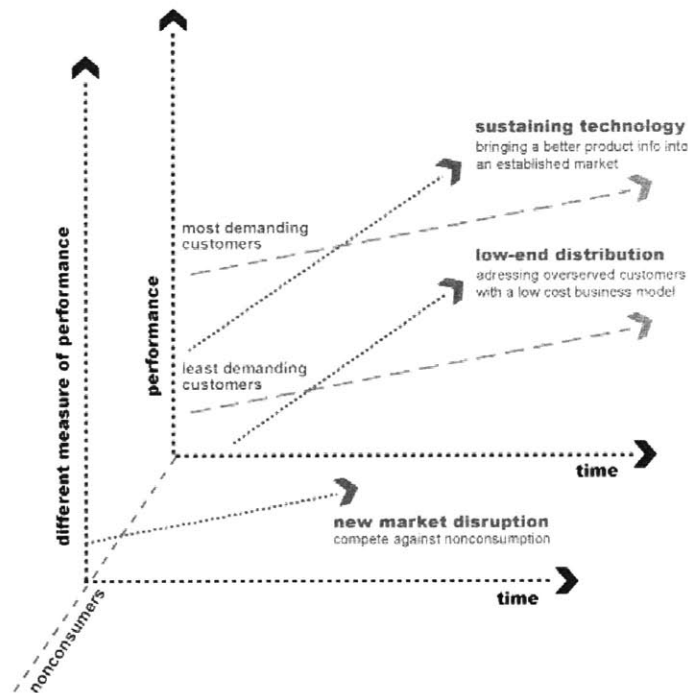


Along the right hand side of the diagram are the performance characteristics along which particular technologies are selected by their customers. As a member of a value network, each

firm becomes adept at delivering the performance characteristics that are valued by their customers. This knowledge becomes encoded in the firm's size, capabilities and cost structures – delivering on known performance characteristics is core to the firm's ability to compete.

The key to disruptive technology is that this type of innovation does not offer improvement to a valued performance characteristic of the existing market; instead it often has a less than desirable performance metric. As such, the firm's customers have little interest in the technology, and the firm is unable to deliver the technology because it cannot make a business case based on demand. The innovation is thus improved in a different market than the existing one, where its performance on the valued characteristic improves over time. This is depicted in Figure 2.6 below.

Figure 2.6 – Disruptive Technology Model of Innovation



After some time, the rate of innovation driven by competition in the mainstream market exceeds the customer demand. The disruptive innovation then targets the mainstream market, in one of two ways:

- *Low-end disruption* – targets the least profitable and overserved customers with a lower cost alternative in an existing value network.
- *New market disruption* – targets nonconsumers in the original network, those who were originally did not possess the resources or skills to participate.

The prototypical process is best described by an example, as presented in The Innovator's Dilemma.

2.1.3-Lifecycle of a Disruptive Innovation

One of the most illuminating aspects of the research area is that the companies that failed in the face of disruptive technology did not fail because they made bad decisions – they failed because they made good decisions in the face of their market and customer demands. Christensen describes the typical decision making process in the following series of steps (Christensen, 1998):

1. Disruptive Technologies Were First Developed within Established Firms

Contrary to popular perception, the leading companies were actually the first ones to develop the disruptive technologies in their laboratories. The innovations were typically a unique architectural combination of off-the-shelf components, and were not initiated by senior management.

2. Marketing Personnel Then Sought Reactions from Their Lead Customers

Once the technology was robust enough to pique the interest of the business leadership, marketing teams would take the prototype technology and show it to their lead customers. The lead customers were not interested, because the technology still was lacking in key parameters they cared about. Practically, this meant that resources were not allocated to work on the new technology.

3. Established Firms Step Up the Pace of Sustaining Technological Development

In search of higher margins, the established firms created sustaining innovations that allowed them to target their larger markets.

4. *New Companies Were Formed and Markets for the Disruptive Technologies Were Found By Trial and Error*

Frustrated engineers from the established firms eventually left and started new firms. Unable to steal the established customers, they were forced to find or create new markets.

5. *The Entrants Moved Upmarket*

Once established, the entrants became able to create sustaining innovations as well – allowing them to increase the missing performance for the established customer bases. They recognize the higher volumes and margins in the established market as attractive and attack.

6. *Established Firms Belatedly Jumped on the Bandwagon to Defend Their Customer Base*

Established firms revisit their existing prototypes and bring them to the changing market. The cost advantages developed by the entrants allow them to sell at lower prices and still capture reasonable margins. Price wars ensued, and established firms survive at best.

To help explain why it is that firms cannot seem to stay on top when faced with a disruptive innovation, Christensen has defined five principles that define disruptive innovation (Christensen, 1998).

1. *Companies depend on Customers and Investors for Resources*

Although managers might think they control how resources are allocated in a company, the best companies perfectly execute on the needs of their customers – those that don't, fail.

2. *Small Markets do not Solve the growth needs of Large Companies*

Disruptive Technologies often allow for the creation of entirely new markets. Since these new markets are necessarily small, they cannot produce the revenue required to allow a larger company to meet the growth metrics required by investors.

3. Markets that don't exist can't be analyzed

Established firms succeed in part because of their ability to assess the feasibility of a strategy and then soundly execute on it. Disruptive technologies and their new markets don't allow for assessment, because the data to be analyzed does not yet exist. This area is not well suited for business planners, and requires a different approach that accepts the fact that some details cannot be known – Christensen calls this “Discovery Based Planning” (Christensen, 1997).

4. An organization's capabilities define its disabilities

Christensen describes a theory of resource allocation based on “Resources, Processes and Values” (Christensen 1998). Resources, which are people and other assets, can easily be repurposed or hired/purchased to work on new and very different technologies. Processes and Values, on the other hand, are very hard to change – they are ingrained into the organizations behaviors and standards.

5. Technology supply may not equal market demand

Despite their original performance gaps, disruptive technologies are disruptive because they evolve to become acceptable in mainstream markets from a performance standpoint. In established markets, firms race with their competitors to create sustaining innovations to increase the performance of their in-market products. This competition often outpaces what customers can absorb, leaving gaps in the low end of the market for disruptive technologies to encroach.

These fundamental principles help inform a set of frameworks and best practices that Christensen describes to help established firms handle the emergence of a disruptive technology. In the following section, we will apply Christensen's frameworks to develop a method for assessing the impact of the potentially disruptive force embodied in new Software-as-a-Service and Cloud Computing delivery models for enterprise software.

2.2- Evaluating Disruption in the Enterprise Software Market

Nearly every company in every industry has business processes that can be improved by the proper use of technology. Over the past several decades, IT has become a critical part of industrial productivity, automating and enhancing workflows that are critical to businesses delivery of products and services. Some examples of market segments in the broadly defined enterprise software market include:

- *Enterprise Resource Planning* – Software platforms which assist with managing the data used in basic business processes like accounting, human resource management, and supply chain.
- *Customer Relationship Management* – Software which assists in managing the interaction of the workforce with customers and sales leads.
- *Enterprise Content Management* – Software which catalogs the information assets of the business, such as specifications, presentations and other materials.

The following section will perform an assessment of the disruptive nature of SaaS in the Enterprise software market based on the research presented in the previous chapter. Doing so will require a treatment of the purchasing behaviours of customers of enterprise software, and the performance characteristics they consider. Then the discussion of how SaaS has developed after the failure of the Enterprise ASP model to take hold will follow. The chapter will end with an assessment of the architectural configuration that allows SaaS to be a viable delivery model, and why this presents a disruption to existing enterprise vendors.

2.2.1-Purchasing Behaviours

A typical corporation purchases a number of IT systems to support its business processes. Most companies centralize the decision making processes around IT into a single IT department, which reports into a Chief Information Officer (CIO). The IT department selects software which is required by the different lines of business (LOBs) and is responsible for piloting it, deploying it and managing it. The IT department must assess and balance the needs of the entire

organization as it applies to a particular IT system being considered. Specifically, when considering an IT system there are several classes of stakeholders whose needs must be considered:

- *End Users* – the general staff who will make use of the system on a regular basis.
- *Developers* – programmers and designers who will perform any customization of the system
- *Administrators* – the administrative staff responsible for keeping the system running, performing upgrades and integrating it with other IT systems.

The amount of money invested in a particular IT system is related to the mission criticality of the particular system to the business. For example, consider the supply chain software used at a company such as Walmart –clearly this would be their biggest IT investment. For each industry, there is range of investment in many individual business processes. For example, authors at IBM Research have proposed the following model of investment in IT systems (Hill, Yates, Jones, & Kogan, 2006):

Figure 2.7 – Investment in Business Process Automation

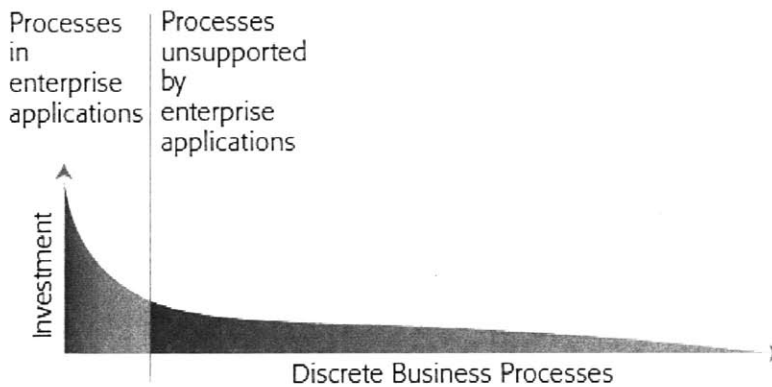


Figure 1
Does a long tail of business processes exist?

For the leftmost processes where the criticality of the system is high and thus the relative investment, the buyer is the central IT department. As the systems progress to the right, buyers become the particular line of business managers who have a higher need – decentralized

decision making comes into play. Often the cost of the systems are lower cost, or are customizations or add-ons to platforms purchased by central IT.

As IT systems are adopted in a decentralized way in lines of business, often the applications or the platforms are based upon are later adopted for support by the central IT department. This is viral adoption of Enterprise systems.

2.2.2-Performance Characteristics

In order to evaluate whether Software as a Service is a disruptive innovation to the traditional Enterprise Software requires a discussion of the valued performance characteristics that IT departments. The following are couple of the key characteristics of Enterprise systems:

- *Value* – the key performance characteristic when selecting an Enterprise IT system is the suitability of the IT system to the business process, and how much value it provides. In essence, this is a measure of the cost savings the IT system produces for the enterprise.
- *Time-to-value* – a measure of how long it takes to get a new IT system into production; how easy it is to install, set up, and train users. This is a function of t

Value is a compound performance characteristic that the IT staff considers in totality, based on the needs of the stakeholders in the previous section. For example:

- *End users* - End users primarily value the capability of the system to properly perform its required function in a business process – its ability to help them get their job done. This implies familiarity relative to other competing systems and other typical IT systems. They also care about its integration with other end-user technologies like mobile devices, etc.
- *Developers* – Developers value systems that are easier and more powerful for which to develop. This means support for languages and toolkits which allow them to perform process customizations.

- *Administrators* – Administrators value the ease of installation and configuration, the range of technologies with which an application can be integrated, They also value other like scalability, performance, compliance, security and other system level characteristics.

It is the IT department's responsibility to balance these individual needs and to work with senior and line of business managers to assess the particular impact of an IT system on the overall business and evaluate different choices to meet the compound needs.

Time to value is dependent primarily on the ability of the IT staff to install and deploy the software, the developers to customize it, and the general staff's ability to absorb and begin using it. For IT systems that are purchased directly in the line of business, the evaluation is often not as critical – they choose whichever is easiest to deploy and manage but still meets their need. As mentioned earlier, this often results in viral adoption.

2.2.3-SaaS in the Consumer Market

As discussed in Chapter 1, SaaS was originally attempted in the Enterprise market by Application Service Provider companies. Although the ASP firms were technologically incapable of making the changes to support the financial requirement to support the customizations of many users, SaaS delivery took hold and became a way to deliver software to end users. Consider the following popular consumer oriented markets where SaaS is used:

- *Online shopping* – Sites like amazon.com, newegg.com and others.
- *Banking* – Banks large and small, credit card companies and trading accounts
- *Email* – Consumer email like Gmail, Hotmail and others
- *Social Networking* – Sites like Facebook, Myspace and Twitter

These sites are accessed with a traditional browser, lessening the need for end users to deal with the challenge of installing additional software on their home computers. This is the consumer analogue to fast time-to-value, except in the consumer market the characteristic is even more important to the providers. As most of these services make revenue based on the amount users accessing and performing transactions on the service, it is important for services

to be usable. Consumers should want to return to the site for more interactions. This is the concept of stickiness applied to online communities.

In order to support these usages, the services in the consumer market must make their services scale to massive amounts of users, and to be usable in a standard browser. Targeting the standard browser allows the Consumer sites to reach the broadest set of users, as most computers come preloaded with an internet browsers – and end users are not good administrators, and thus are not likely to install a providers native operating system software. Sites that offer customizations offer the most standard of technology choices to developers, such as basic web languages like HTML and Javascript or other open technologies like PHP, Perl and Python. Instead of pushing proprietary development technologies, consumer firms choose open technologies because the goal for consumer markets is typically to attract end users. Preferring customizations in the most standard of technologies targets a wider developer base and allows for interesting ‘mashups’ of consumer content.

The platforms that are being built in the consumer market are quite capable. Facebook has over 350 million active users. Both Google and Amazon are offering their underlying platforms in different ways for developers to use and extend. These platforms meet many of the same performance characteristics required of enterprise software at with a very different cost structure. Consumer sites have learned how to scale their services out to massive amounts of users, surmounting the challenge that ASPs failed to. Now that the platforms have been built, many of these consumer companies are in a position to offer services to alternate markets, leveraging the scaling technologies built into their platforms.

2.2.4-Technological Architecture of SaaS Disruption

As described above, the ability to scale out to massive amounts of users and deliver software took off in the Consumer service market. There are several technological trends that have impacted this development, but the two most important ones are the widespread availability of internet access, and the ability to massively scale horizontally through the use of virtualization technology, which more effectively uses processing power.

In Figure 1.1 in Chapter 1, Gartner described the layered architecture of a SaaS system and how the key technology of virtualization is used at every layer. Virtualization allows processing power to be used incredibly efficiently, and a properly structured architecture allows computing

power to be added to support massive load. Virtualization is especially important at the following layers of the stack:

- *Operating System* – Operating systems can now be scaled across multiple machines, using technologies first developed by companies like VMWare, but increasingly integrated into operating systems from Linux to Windows. Being able to virtualize an operating system means that capacity can be added in the form of new virtual servers, ensuring that the CPU utilization of each hardware node is maximized.
- *Data Platform* – The data platform is the SaaS analogue to the traditional relational database. Instead of relational databases, SaaS platform providers have created databases tailored for massive horizontal scalability, taking advantage of the virtualized OS layer below.

The chart does not assess the client projection, which also factors into the nature of the disruptive challenge for Enterprise Software vendors. In particular, time to value is much higher with browsers - end users are already familiar with browser interfaces, and no software is typically required to be installed on the end user desktops. Browsers have continually evolved to be increasingly standard as a method of accessing the internet, and are an integral method of SaaS delivery.

2.2.5-Disruption of the Enterprise Software Business Model

Relative to the description of a disruption in Section 3.2.3, the disruption of Enterprise software vendors by service based offerings is well into its lifecycle. Enterprises have developed browser based software for years and delivered it as an on-premise offering, but Enterprise customers have valued strongly the rich user experiences delivered by native operating system software, customizing it with their own IT staff to deliver higher value. The vendors have thus continued to develop sustaining native OS technology and added features to every layer of their enterprise stacks. At the same time, simple but usable applications have been developed for the web, paired with massively scalable datacenter delivery. In some markets, the entrant technology has moved upmarket – the classic example being Salesforce.com in the CRM market. The Enterprise Software vendors have taken notice of this trend, and are clearly trying to augment their own portfolios with browser delivered, subscription based offerings.

Customers of Enterprise Vendors invest heavily in maximizing the value performance characteristic of their IT systems – often building add-ons and customizations with dedicated staff trained to work with Enterprise Vendor’s offerings. Training staff to use the system and keeping IT staff on hand to administer systems is a cost that gets amortized over many years. At the high end of the spectrum are customers who are overserved on function, who rarely even perform upgrades because the cost of change outweighs the benefit of new functionality offered. Large customers are thus very resistant to change, even to roll out new versions of software. At the low end of the spectrum are small businesses, which have simpler needs and no wish to manage and sustain expensive IT systems for themselves. These customers are overserved on function, and are a perfect target for disruption from a subscription based model. Given the nature of existing IT systems inability to spread costs across multiple customers, customers are overserved on processing power across both ends of the spectrum – thus the disruption has room to continue upmarket.

Enterprise software finds itself even more challenged by the nature of how Enterprise software is purchased. As described earlier, central IT departments make the decisions about what enterprise software to purchase and deploy, based on an evaluation of the needs of various stakeholders in the business. At the same time, they often find themselves faced with needing to support applications which have been virally adopted from line of business buyers. As pointed out in Chapter 1, since SaaS offerings are simply subscription based, and often require less than a credit card to sign up, they are increasingly purchased by line of business buyers. This means that the more that a particular IT system can appeal to the LOB purchaser; the more likely it is to get picked up. Another trend that compounds this viral adoption of SaaS is the so-called “consumerization of IT” – where IT purchasing decisions, both at the IT and LOB level are increasingly influenced by the preferences of the end users rather than the administrators or developers. This is largely because the general population has become more comfortable and opinionated about technology as it has become so critical to everyday life through the growth of the consumer market.

Subscription licensing does not offer the revenue potential that license and support contracts to do traditional Enterprise Software vendors. Because of the lower profit margins, the incentive to switch delivery models is very low, and contradicts the existing value systems of traditional Enterprise Software Vendors. Coupled with the fact that IT departments have already invested in customizations and staffing related to their entrenched systems, this presents a serious

challenge for Enterprise vendors to embrace the disruptive SaaS technologies and delivery mechanism without disrupting their existing revenue stream.

2.2.6-Conclusions

Enterprise customers must consider the needs of several stakeholders in their organization when selecting software to augment their business processes. The most important aspect to them is the value to the businesses processes that the software can bring, as well as how long it can take to bring that value to the organization. At first, browser-based SaaS offerings lacked the power and depth of native applications, but the advances in virtualization and browser capability that have evolved in the Consumer market present a low-end disruption to overserved enterprise customers. Therefore, enterprise software vendors must develop the capability to deliver scalable and useful SaaS based offerings without losing their existing customer base as they develop their replacement offerings. The following chapter will present a framework for embracing SaaS technology by structured use of platforms and open innovation principles.

THIS PAGE INTENTIONALLY LEFT BLANK

3- A Strategic Framework for Embracing SaaS

In previous sections, the discussion began with a background on the subject of the ongoing case study for this paper – the collaborative groupware platform IBM Lotus Notes. A discussion of the ongoing changes in the Enterprise Software market followed, presenting the benefits, challenges and ongoing trends in Software as a Service (SaaS) and Cloud based software service delivery. Finally, there was a discussion of SaaS and Cloud delivery as a disruptive technology relative to the existing model of software and services delivered by Enterprise software vendors like Microsoft and IBM.

Given that it is clear the challenge that SaaS and Cloud based delivery presents to incumbent firms in the market for enterprise software, it is important to have a model for responding to the challenge. In this section, a strategic framework for structuring and executing the will be presented which leverages research in the areas of Open Innovation and Software Platforms. This chapter will have the following structure:

1. An overview of the research of technology “platforms”
2. An overview of the research in open innovation
3. The development of a strategic framework using aforementioned concepts
4. Continued case study using IBM Lotus Software, including:
 - a. Current work being done in support of this model
 - b. Areas where work should be getting done but is not
 - c. Areas where work could be improved or refined

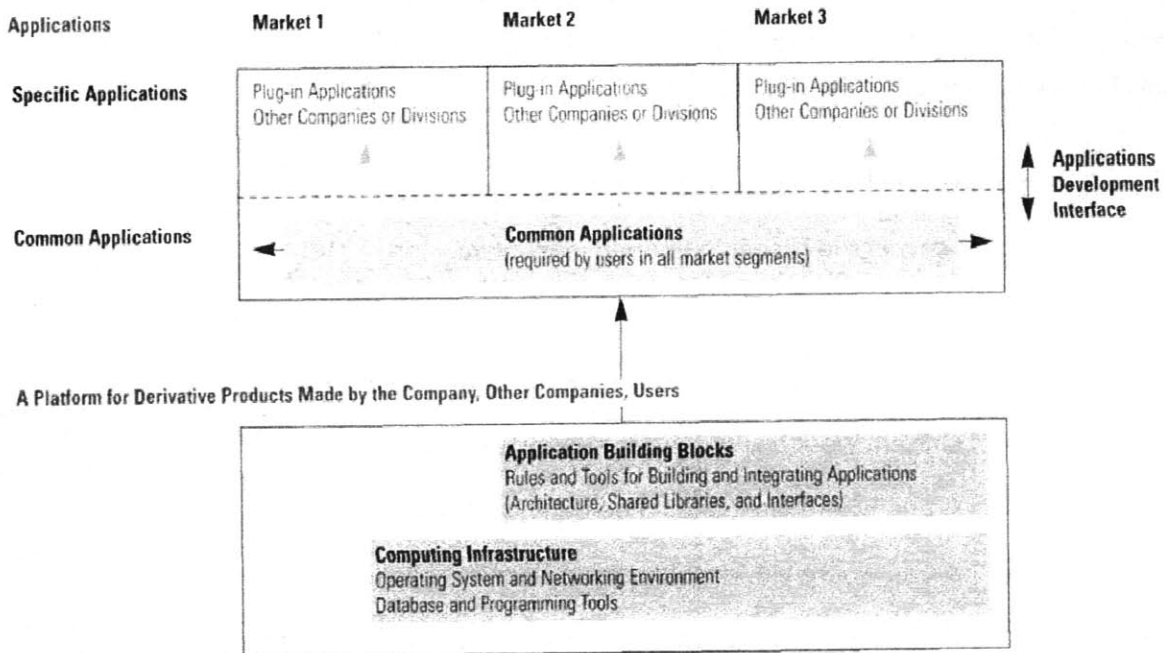
After this section on a strategic framework is complete, the discussion will close with some key conclusions and recommendations for areas of further research.

3.1- Research on Technology ‘Platforms’

A fundamental concept that is a key to the competitiveness of companies in the software industry is that of product “platforms”. Meyer and Selinger (Meyer & Seliger, 1998) define a product platform as “a set of subsystems and interfaces that form a common structure from which a stream of derivative products can be efficiently developed and produced”. These subsystems are modular, allowing the platform to be used to service the needs of different consumers, both internal and external. The authors illustrate the concept as follows:

Figure 3.1 – Platform Strategy for Software Products

Figure 1
Platform Strategy for Software Products



A platform has complementary products that are built on top of it, often built by different firms than the ones that produce the platform itself. Together the firm that produces the platform and the firms that produce the complementary products form a product “ecosystem”. According to Meyer and Selinger, the creation of a platform can provide substantial R&D productivity benefits, and the associated ecosystem allows for rapid development of market share and revenue.

3.1.1-Platform Leaders, Wannabes and Complementors

Cusumano and Gawer (Cusumano & Gawer, 2002) describe platform leaders as companies that drive industry wide innovation for an evolving system of separately developed pieces of technology. Platform wannabes are companies that would like to be in the industry leader position. Complementors are companies that make ancillary products that extend the platform's market.

3.1.2-Benefits of Technology Platforms

Creating a product platform offers a number of benefits to a company. Meyer and Selinger describe the ability to more rapidly develop products on top of the platform in order to meet the needs of market niches. They also point out that customers of the platform enjoy the advantage of consistency of interfaces when performing integration, which increases loyalty to the platform producing company. Lastly, being a platform creates the opportunity to become a standard for large scale innovation, which allows the company to potentially serve as a sales and distribution channel. With proper licensing, a company can share in the profits created by this complementary innovation. Cusumano and Gawer add that platform producers also benefit from the competition in the ecosystem, allowing prices to resist downward pressure. Evans, Hagiu and Schmalensee describe platforms as enabling "network effects" (Evans, Hagiu, & Schmalensee, 2006), where the value of the platform increases as more adopters come online. This is most evident in technologies like social networking platforms. Another important aspect of platform is that well designed architectures are sticky, meaning that adopters of the platform are hesitant to take on the switching costs of moving to an alternative platform. Again, with proper licensing strategies, this creates the potential for recurring revenue streams.

3.1.3-Challenges to Becoming a Platform Leader

There are challenges to becoming a platform leader. One is that a market may not have the characteristics to support a platform leader. Large markets may have enough room for differentiation in user needs that multiple platforms can exist simultaneously, occupying different niches. Another challenge is the complex relationship management required between platform producers and complementors, where in many cases a complementor can simultaneously be a competitor in adjacent markets. Cusumano and Gawer present several strategies for overcoming these challenges, which will be discussed shortly.

3.1.4-Choosing a Platform versus a Product Strategy

The first action that a software product vendor should take is to explicitly embark on an exercise to decide whether to produce a product or a platform. The strategies for becoming a platform producer are quite different from those of pure product development. In assessing whether a product can be produced instead as a platform, Cusumano and Gawer state that two fundamental conditions must be met (Cusumano & Gawer, 2002):

1. It should perform at least one essential function within what can be described as a “system of use” or solve an essential technical problem within an industry.
2. It should be easy to connect to or build upon to expand the system of use as well as to allow for new and even unintended end-uses.

They describe methods for testing these conditions. The first can be tested by asking whether the system can function when the particular product or technology is removed. For the second, the test should be to see whether external companies have either succeeded in creating products that interoperate or have started down a path to do so.

3.1.5-Strategies for a Platform Leader Wannabe

Once the decision to pursue a platform strategy has been made, there are methods that platform leader wannabes can pursue to achieve that end.

The first method is called “coring”, which is how a company can identify a technology that has platform potential and then make it integral to the functioning of a technological system. A key attribute of the technology must be that it solves a need that is present in a large segment of the system. Often locating or designing this technology is easier than achieving its adoption in the ecosystem. Once the technology has been identified, the wannabe must create monetary incentive for members of the ecosystem to attach themselves to this technology. While creating this incentive, the company must protect their own ability to profit.

This balance between incentives and protecting profit are explained by positive and negative examples. The first is Google, whose search technology allowed them to link advertisers with users. This created economic incentives for advertisers, while protecting their profit by not

disclosing the algorithm they use to achieve their targeted results. Another positive example is Qualcomm, who solved the problem of incompatible cellular networks. While licensing the technology to system providers, they strongly protected their IP via patents and active litigation. A final negative example is EMC, whose WideSky storage standard was widely discounted by players in their ecosystem, who banded together to form an open alliance to channel their own standard.

The second method of achieving platform leadership is “tipping”. Tipping is a method by which a company shapes the market by facilitating the winning of a platform war between two or more competing platforms. Cusumano and Gawer recommend the most promising action to take here is to gain control over an install base, license related technology and foster complementary innovation.

Companies should also use pricing as a weapon in a standards war. Cusumano and Gawer describe that platforms are often used in “double-sided” markets – where the platform serves as an intermediary between two paying partners. An example is video game consoles, where game developers license technology to create games and consumers pay money to own them. Pricing here can be used to drive network effects by subsidizing the correct side of the market. Care must be taken not to undercut complementors by setting price to zero.

Another way to tip a market is to use platform envelopment. This is when an adjacent market is absorbed to extend the scope of the existing platform. This is readily seen in cell phone markets, where phones often overlap with cameras, media players and PDAs.

3.1.6-Maintaining Platform Leadership

For companies that are either platform leaders or want to become one, Cusumano and Gawer describe “Four Levers of Platform Leadership” – tools that a platform producing company must use to gain or maintain platform leadership.

- Scope of the Firm
- Product Technology
- External Relationships
- Internal Organization

The first lever is the scope of the firm. The scope of the firm is the delineation between what parts of the whole product system the firm creates internally, versus what parts of the system will be created by complementors. Firms must be cognizant of their own capabilities, and if a company does not have a technical competency in creating a product for a market where the platform will be used, they should allow complementors to create it. The same holds true when considering their business capabilities - firms should not enter markets where their business processes, such as sales channels, are not complementary. Firms must explicitly examine the incentives and capabilities of others in the market in order to be able to effectively influence the design. When selecting a complementor, a recommendation is to be very visible in assisting the complementor to deliver on its product by sharing interface definitions and designs. This signals that the market opportunity is promising, potentially inviting more innovators. Another potential signaling opportunity is to invest in complementors.

The second lever is the product technology itself. The platform architecture is fundamental to facilitating the creation of products on top of it. Architectural decisions can impact everything from the structure of the industry to the levels of investment that will be possible for complementors. The use of modular architectures is recommended to keep innovation costs low for complementors and to encourage specialization on top of these modular interfaces. Fully open interfaces present both an opportunity and a challenge, as they foster stronger innovation but also make the potential for imitation riskier. Finally, it is important for the platform to evolve as externalities evolve and for the company to maintain control over the core architecture.

The third lever is external relationships. Firms are required to interact with complementors to design specifications and standards that will allow the platform to interact with products. A platform firm must also influence the decisions that complementors make regarding the products they develop, ensuring that the ecosystem grows fortuitously. Cusumano and Gawer call this "ecological control". The most important way to establish this ecological control is to ensure that complementors have trust in the platform producer. This means that producers should be careful when sending signals from acquisition activity, investments and market selections. In general, platform producers should sacrifice short-term interests, instead choosing to invest in the common good.

The final lever is internal organization. This lever is important to assist in maintaining consistent behavior in the ecosystem from the platform producer. The platform producer should be careful

to structure internal organization in such a way to minimize conflicts of interest. One way to manage this is to have a clear separation, where some groups focus on competition and others instead on consensus building in the ecosystem. Another is to keep the platform producing organization separate from the applications organizations.

3.1.7-Conclusions

Platforms are an important and complex topic in the software industry. Choosing the right strategy requires an analysis of the firms markets, followed by a dedication to rigorous architecting and competent relationship management. From the literature, it is clear that becoming a platform leader can present challenges, but presents tremendous benefits to the platform producer in the long term.

3.2- Research on Open Innovation

One avenue of potential product strategy that will be explored in the following section is the concept of 'open innovation'. The Merriam-Webster dictionary defines innovation as "the introduction of something new", and an appropriate definition of the word open in the context of this discussion is "completely free from concealment: exposed to general view or knowledge". This area of study revolves around the variety of ways that innovation is created and delivered, but does not immediately command a monetary price – the innovation is shared openly and freely.

In business, innovation is the engine that keeps businesses alive. As users needs change and as markets evolve, businesses harness the ability to innovate in their organization and workforce to deliver products that meet those changing needs. Conventional wisdom would be that innovation is the domain of businesses and that revealing innovation instead of locking it away in traditional intellectual property vehicles would be anathema to a solid business strategy. The work of Professor Eric von Hippel of MIT revolves around research that shows innovation does not simply take place in the halls of research and development labs. Instead, there is an increasing amount of evidence that end users and consumers of products are often the most prolific innovators, and that they are surprisingly likely to openly share those innovations with others for no cost.

3.2.1-Lead Users

The traditional view of manufacturers being the sole creators of innovation in markets is challenged by research that shows that so called "lead users" are frequently innovating in many markets (von Hippel, 2005). Through research of innovation in several fields, von Hippel found that many users innovate in range ways, ranging from simple customizations to whole product development. A sample of the findings follows in Figure 3.2:

Figure 3.2 – Quantitative Study of User Innovation

	Number and type of users sampled	Percentage developing and building product for own use	Source
Industrial products			
1. Printed circuit CAD software	136 user firm attendees at PC-CAD conference	24.3%	Urban and von Hippel 1988
2. Pipe hanger hardware	Employees in 74 pipe hanger installation firms	36%	Herstatt and von Hippel 1992
3. Library information systems	Employees in 102 Australian libraries using computerized OPAC library information systems	26%	Morrison et al. 2000
4. Surgical equipment	261 surgeons working in university clinics in Germany	22%	Lüthje 2003
5. Apache OS server software security features	131 technically sophisticated Apache users (webmasters)	19.1%	Franke and von Hippel 2003
Consumer products			
6. Outdoor consumer products	153 recipients of mail order catalogs for outdoor activity products for consumers	9.8%	Lüthje 2004
7. "Extreme" sporting equipment	197 members of 4 specialized sporting clubs in 4 "extreme" sports	37.8%	Franke and Shah 2003
8. Mountain biking equipment	291 mountain bikers in a geographic region	19.2%	Lüthje et al. 2002

From these findings, it is clear that many users innovate. A meaningful number of users in each of the markets investigated are actively innovating in their respective product category. There is a potential response bias in this type of study, potentially compounded due to the selected populations' strong affinity for their product category. However, study of "outdoor consumer products" - a broad category with a weaker affinity – provides evidence of a potentially large population of user innovators, 5%-10% of this market could number in the millions. Many of the innovations in these studies are likely to be minor, consistent with the general manufacturing population. However, minor innovations are not necessarily trivial – meaning that minor innovations have strong cumulative benefit. Progress in Rayon manufacturing and computers are given as evidence. A strong area of user innovation is process improvement, where technology is applied for purposes. Lathes and milling machines are given as examples here.

Von Hippel found that the users creating innovation in these categories share a couple of important characteristics (von Hippel, 2005):

- 1) They are at the leading edge of an important market trend(s), and so are currently experiencing needs that will later be experienced by many users in that market.
- 2) They anticipate relatively high benefits from obtaining a solution to their needs, and so may innovate.

The reasoning for the first point comes from diffusion theory of users and their needs –market needs are not static, and that they change over time. Therefore, the users at the leading edges of market have needs that will eventually be experienced by the remainder of the market. The second is from studies that have shown that the benefit that an entity expects directly impact the investment that the entity will make towards purchasing or developing a solution.

Von Hippel describes user's needs as often being heterogeneous, meaning that it often the case that many users have different needs for a particular product type. This implies that manufacturing firms will have difficulty making the case to produce products that can be suitable for these product types, as they cannot defray the costs of the specialized development. As a result, these users have to make a decision whether they will develop it themselves or have it made by a specialized manufacturer. In studies of Library IT systems and mountain biking, the results show that . Market segmentation studies also show this evidence, although a typical market segmentation study uses cluster analysis to find homogeneity, not heterogeneity. Despite this difference of intent, a survey of 14 market segmentation studies specifying an average of 5.5 clusters with an average in-cluster variance of 46%. This low variance implies a strong heterogeneity of need.

Since needs are heterogeneous, users with specific needs must decide whether to innovate on their own or to purchase a solution to their specific need. Von Hippel's research shows that transaction costs and information asymmetries affect the likelihood that a lead user will choose to enlist the help of a specialized manufacturer or instead innovate on their own. The transaction costs that are identified include (von Hippel, 2005):

- 1) Differences between users' and manufacturers' views regarding what constitutes a desirable solution

- 2) Differences in innovation quality signaling requirements between user and manufacturer innovators
- 3) Differences in legal requirements placed on user and manufacturer innovators

The first two are costs associated with the ramifications with working through an agent – the end user (the principal) has specific preferences and expectations of the manufacturer's product, while the manufacturer (the agent) has a need to maintain low costs, both for the specialized manufacturing as well as signaling costs (support, etc). The third is a cost that typically does not exist for the user innovator, but can be a significant challenge for a manufacturer innovator depending on the market.

User innovators also benefit from the “sticky” information about their own unique needs. This information incurs a cost to it being shared, from reasons ranging from being tacit and loosely understood to explicit financial fees potentially charged by the holder of the information.

Typical product design methodology uses market researchers to study markets, draw out user needs and communicate them to product designers. In this model, lead users, and their solutions, are often regarded as anomalies and discarded from the needs samples. Lead users can instead be chosen to harvest ideas and their solutions, and methods for identifying and interacting with lead users exist. The first is to search for strong needs in advanced analog fields, as an example consider the development of automobile brakes having an advanced analog field of airplane braking. At times, lead users in one field can be used to find experts in analog fields – they often know these experts. This method is called pyramiding. The second method of finding lead users is to simply look in the target market. When possible, screening can be used.

3.2.2-Free Revealing

The traditional model of innovation is based around private investment, where the cost of innovation is absorbed by private investors and repaid as profits from the manufacturing and licensing of innovations. It then stands to reason that any leakage of this innovation funded by private investment into the public hands would cause a net loss in the profits available to the innovator. Von Hippel's studies of several different areas of innovation reveals examples where innovators did not choose to protect their intellectual property with patents and licensing schemes, but instead chose to freely reveal their innovation.

There are reasons that innovators freely reveal. Free revealing often is the most practical choice for innovators, emerging from the basic premise that it is actually quite difficult to prevent imitation. Some of the reasons for this include the fact that very often there are many other innovators who are in the position to know about the same innovation in a relatively short timeframe. One of key points given was that the revealing of a general solution - instead of a specific implementation of an innovation - facilitates imitation. There are several other reasons that a general solution is hard to keep secret, ranging from the fact that general solutions are often visible in other related problem domains, and that it appears that many firms and individuals are often in the possession of similar information. In addition, even if a secret is not readily available from the aforementioned sources, an innovating firm is likely to give up details sufficient for imitation simply through the leakage of information in product release marketing advertising or related materials.

There are tangible benefits to free revealing. In a number of cases, the choice to free reveal was complemented by an active effort to diffuse the information, which suggests that there is benefit to others seeing the information. One of the key observations is that there is a positive reward to the innovators reputation when a well prepared and articulated innovation is released to the public domain. An area where this is clearly evident is in open source software development, where developers have been able to make professional careers out of reputation, enabling them to work on code that is their passion and be paid by corporations who subsequently benefit from enabling and harnessing that fact.

3.2.3-Innovation Communities

Von Hippel describes methods by which lead users work together to create innovations. He defines an “innovation community” as “nodes consisting of individuals or firms interconnected by information transfer links which may involve face-to-face, electronic, or other communication” (von Hippel, 2005). Innovation communities are familiar in the world of open source software development, where users collaborate in email lists, defect tracking systems and support forums – all while freely revealing their innovations in the form of code patches. The emergence of these innovation communities around open source projects is rooted in the economics of supporting the innovation. In most cases maintaining a patch or innovation to an Open Source product incurs a substantial maintenance cost of patching, building, and redistributing the innovation every time the underlying project changes. It is therefore beneficial to the innovator to

reveal the innovation and rely on the community as a manufacturer, allowing it to absorb these maintenance costs.

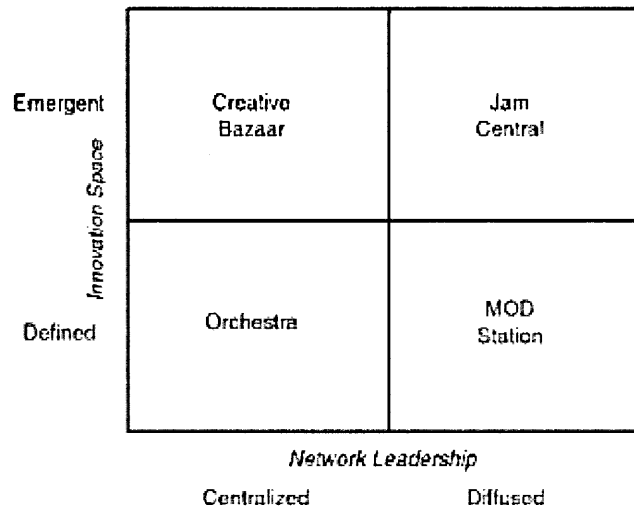
Eric Raymond's related research on Open Source shows the intrinsic value of innovation communities on product quality. In particular, he coined the phrase "with enough eyes all bugs are shallow" (Raymond, 2001). He points out that the economy of scale in the community is built on the fact that a bug is likely to be in the expertise and interest set of one or more users, an economy of scale based on the size of the innovation community. Community responsibility is strong in open source communities, where Lakhani and Wolf found that community motivation is one of the top three determinants of hours spent of Open Source project Apache Web Server (Lakhani & Wolf, 2005):

- Enjoyment-related intrinsic motivations in the sense of a form of creativity
- Extrinsic motivations in the form of payment
- Obligation/community-related intrinsic motivations

This implies that a well created and tended innovation community can not only draw in innovation, it can also make contributors stay with the community by imbuing in them a sense of community responsibility.

Related to von Hippel's work, Nambisan and Sawhney (Nambisan & Sawhney, 2007) propose several models for creating innovation communities. Their names and attributes are captured in Figure 3.3 below:

Figure 3.3 – Quadrants of Network Innovation Model



The innovation space on the vertical axis describes the type of innovation being worked upon, ranging from a loosely defined and exploratory space in the emergent type to a clearly defined and structured space in the defined type. On horizontal axis is Network Leadership, a measure of the centralization of leadership ranging from extremely centralized to diffuse.

From this treatment, Nambisan and Sawhney propose four models for innovation communities (Nambisan & Sawhney, 2007):

- **Orchestra** – In the Orchestra Model, the innovation space is well defined and the leadership is centralized. The behavior is like an Orchestra with a Conductor, where the conductor sets the direction and coordinates the contributions of the members. Two variations on the model exist. One is the Orchestra-Integrator model, where a dominant firm creates the architecture and supporting members contribute the components that make up the final deliverable. The leader is responsible for the marketing of the final deliverable. An example here is the development of the Boeing 787 Dreamliner, where Boeing owns the final deliverable, but coordinates the contribution of hundreds of global partners. The second model is Orchestra-Platform, where the leader offers the core architecture and the supporting members contribute value through extensions or enhancements. The example here is Salesforce.com, who define a software platform for managing sales and marketing information. Salesforce.com allows their partners and

customers to contribute ideas and applications to their core platform and share them with other innovators.

- **Creative Bazaar** – The Creative Bazaar model is appropriate when a leader is seeking to reach out into a collective wisdom to harness new ideas, as with the models around lead users discussed earlier. The physical analogy is to a shopping Bazaar, where a multitude of vendors come together to offer a range of items, from ingredients to fully complete items. The method can be used to seek out a range of innovation, from those that are merely raw ideas in an innovators mind, to those that are thought out, implemented and ready for market.
- **Jam Central** – The Jam Central model is based on the model of creative music creation of musicians “Jamming” together. In this model, the innovation is emergent, much like the music created at a jam session. Additionally, no one member of the community is the full-term leader, instead leadership is diffuse and shifting. An example of this model is the Apache Software Foundation, where the charter and individual projects are all collectively and democratically owned.
- **Mod Station** – The Mod Station model comes from the world of computer gaming, where game developers allow players to work as developers and modify the game to suit their individual needs. In this model, the innovation is well defined – like the available customizations on a game engine. The leadership model is diffuse, as the owner of the platform does not explicitly lead the innovation community – the community members are free to create whatever they choose.

These types of innovation communities will appear later in the discussion of how to structure innovation communities in the face of disruptive technology.

3.2.4-Toolkits

Von Hippel suggests that a way to harness lead users is to facilitate the manner in which they can interact with a manufacturer of a product or service. In particular, he describes the concept of a “toolkit” which allows manufacturers to get immediate feedback from these lead users regarding the designs that they would like to create. The goal of the toolkit is to make user innovation both quicker and more inexpensive, potentially increasing the volume of user innovation. The usage of toolkits can help to channel innovation in directions that are most beneficial to the manufacturer.

The primary reason that toolkits are compelling is that they allow the manufacturer to shift the iteration and learning cycle that refines the users need to the user himself. As found in the research of lead users, the user knows their needs better than any manufacturer. Therefore, when implemented correctly this method necessarily produces solutions that best meet the user’s needs. In order to achieve best results, there are implications to both the product development process, as well as the architecture of underlying products and platforms.

With respect to process, the goal of a toolkit is to eliminate the need for cross-boundary iteration between user and manufacturer sites during product development. An example is the introduction of spreadsheet programs, which are essentially a toolkit used in the execution of financial analysis – a form of manufacturing process. In the past, a CFO would work with assistants to perform an analysis, refining the work iteratively based on his or her expert knowledge. This would be time intensive. The introduction of a spreadsheet into the process allows the assistants to instead develop a model in a spreadsheet, which could then be tested live by the CFO. This dramatically reduces the time for iteration and produces better results.

Regarding architecture, the product platform must support the right level of modularity to facilitate this type of customization. The example here is the creation of customized integrated circuits. In the past, this process was a highly iterative process between the user and the manufacturer, where the prototypes came at high cost and were often insufficient to meet the user’s needs. The change in this example was when it became clear that the production of transistors and chips was something that was independent of the user’s needs, and could be developed without iteration. The user’s responsibility was then to lay these chips out on a circuit board, assisted by a toolkit to do so. This architectural shift facilitated the separation of roles highlighted above.

Von Hippel describes five fundamental requirements in the development of a toolkit which are critical to achieving positive results. First, it must enable users to iterate over trial-and-error cycles – the cycle of trial and error is fundamental to the refinement of user needs and thus solutions. Second, it must offer a solution space that encompasses designs that users want to create – this means that the production system that the toolkit models must be sufficiently flexible to support an appropriate range of user designs. Third, the toolkit will be user friendly and operable with little specialized training – there should be no heavy training required outside of the users existing domain of knowledge. Fourth, it must contain libraries of modules that users can use as building blocks – the user should be able to build upon prebuilt modules to create more advanced designs. Finally, it will ensure that designs can be produced without modification by the manufacturer – the designs that the user creates should be free from errors in translation to the production system.

3.2.5-Conclusions

The concepts of open innovation described by von Hippel have wide application across many domains, and have particular relevance to the business of software based on their prevalence in the domain of open source software. These fundamental building blocks will be used in later sections to facilitate the execution of strategic goals for the Lotus suite of products

3.3- A Strategic Framework Using Platforms and Open Innovation

Organizational change and readjustment in the face of disruption is extremely difficult, as evidenced by the examples found by Christensen in his research on the subject. However, surviving a disruption is possible and strategies can be employed by incumbent firms to weather the situation, to change with the evolving marketplace, or to even come out on top.

A strategic framework has a structure that helps to set expectations for the organization to follow as the transition progresses, starting from a high level and continuing down to specific actionable goals. Specifically, the following areas of focus and deliverables are defined:

- *Vision Statement* – A high level message that sets the context for the business transformation, meaningful and addressed to the entire organization.
- *Mission Statement* – A high level overview of the methods that will be employed to facilitate the business transformation.
- *Strategic Goals* – A set of actionable strategic efforts towards transformation that will be undertaken by specific parts of the organization on specific timeframes.

Consider the broad case of IBM as the internet emerged to become a cornerstone of the communications infrastructure of the world in the 1990s. As this phenomenon began, IBM was firmly entrenched as the leader in mainframe and proprietary client/server computing architectures. IBM embarked on an effort to project appropriate market messages (the e-Business initiative), changes in product architecture to embrace and leverage the disruptive internet technologies, and organizational changes to bring on talent and new skills through acquisition and hiring. As mentioned earlier, the Lotus product set evolved to support many new internet standard protocols – POP3, IMAP, LDAP and HTTP. The result was a reemergence of IBM as a business services company, using the internet and its open protocols to better deliver on integrated products and solutions for its customers.

A strategic framework is a mechanism by which senior leadership can organize a broad approach to tackle a transitional problem, helping the entirety of the organization to understand the overall direction and their role in facilitating organizational change. In the following sections,

a specific strategic framework will be presented that will be tailored to help an enterprise software firm tackle the challenge of embracing Software as a Service delivery using methods of open innovation and software platforms covered in the previous sections.

3.3.1-Context and Assumptions

The strategic framework that follows is built to address the needs of a firm which finds itself operating in a particular context, which is defined by set of environmental conditions. The conditions that apply to this framework are as follows:

- *Traditional Enterprise Software Vendor* – the firm using this framework should be a traditional enterprise software firm. These firms sell software via license and support contracts. They retain users through brand loyalty derived from actual value, stickiness from product customizations and overall workforce familiarity.
- *Market can sustain platform(s)* – Per Cusumano’s research, the demand for solutions in the firm’s markets must be strong enough to sustain one or more platforms.
- *Intend to be a platform “Leader”* – The firm is already, or intends to be (and potentially could be) a platform “leader” in their primary market. This implies established brand presence, customer base and development capabilities. The framework is for established players looking to make the transition to SaaS delivery.
- *Facing a Disruption from SaaS* – Obviously, this framework is intended to transition to SaaS delivery, so the firm should have at least realized that SaaS and Cloud may impact their revenue stream. The framework will solidify this perception into a reality of impact and timeframe.

3.3.2-Strategies for Embracing Disruption

Clayton Christiansen's research on disruptive technologies also provides some guidance for companies who find themselves in a position of being disrupted by a technology that is undercutting them.

The following are the key findings of the methods that successful firms employed in embracing disruptive technologies:

- *Match the disruptive technology to an organization with customers that need it* – In order to successfully increase customer demand and finance further development, firms must match the technology to customer demand. Doing so creates a positive demand loop that will allow the organization to get the resources that it needs.
- *Match the projects to appropriate size organizations* – embracing disruptive technologies means developing competency through the launching of projects which utilize the technology. Successful firms match the technology to an organization of appropriate size, ensuring that the technology will get the focus it deserves.
- *Plan to fail early and inexpensively* – Entering these new markets with new technology is bound for numerous failures as matching markets are eventually found. Successful firms planned for this by entering early, performing numerous trials, and iterating on the technologies.
- *Utilize existing resources but not processes and values* – As described, resources in a firm can be retargeted to embrace a disruptive technology. Processes and values can be much harder to change, and can cause failure
- *Found or developed new markets* – Instead of forcing the disruptive technology on their existing markets, successful firms instead chose to match the disruptive technologies into markets where the technology would be readily accepted and . This meant that they had to understand the capabilities of the technology, as well as the needs of the markets into which they were selling.

3.3.3-Vision Statement

The vision statement is the highest level articulation of the end state to which the business leadership would like the business to evolve. It gives the organization a common understanding of purpose around which to rally, and helps to define both internal and external strategies that will allow the evolution take place. The vision statement should be clear enough so that any member of the organization can read it and understand the transition that is to take place, but not the exact methods to be employed in the exercise.

As described by the background and assumptions outlined in the previous sections, the firms that produce traditional on premise software for the enterprise market find themselves in a situation where new delivery methods are disrupting the landscape of software selection, purchasing and pricing. As a result, they need to design ways to transition their existing revenue streams from their existing model to a fully service delivery model, or something blended and hybrid.

A vision statement is simple and straightforward, and should be customized to the specific firm, but a boilerplate statement from which to begin would read like the following:

“In the face of new software delivery methods and revenue opportunities driven by improvements in the internet as a standards-based platform, the firm intends to transition its software delivery mechanisms to adopt this new method over time”

This vision statement should be routinely reinforced with the organization through structured messaging and presentation. The vision statement is then refined into a mission statement which provides more structure and the beginnings of a method, as outlined in the next section.

Mission Statement

The mission statement builds on the vision statement and begins to articulate how the firm will carefully execute on the transition from on-premise software licensing to a SaaS based delivery model and revenue stream. Mission statement continues to be high level and appropriate for the entire organization, but should add a level of detail that begins to remove ambiguity about how the firm will structure and schedule its efforts.

The mission statement for the transition comes in several parts, the first two being market oriented:

- ***The firm will embrace disruptive Software as a Service technology; grow it into a platform in a niche market with strong associated needs over an appropriate timeframe.***

This first section of the mission statement articulates the need to build the ability to bring software as a service delivery into the DNA of the organizations values and processes. At the same time, this statement mandates that SaaS be used in a targeted manner at first – preventing the organization from shifting too quickly, resulting in potentially shoddy work and duplicated efforts as every team seeks to understand the ramifications of the technology changes. It also increases the likelihood that the firm will send positive market messages, taking care not to confuse its existing market and customer base. Entering too loudly can send an inappropriate message about the viability of encroaching disruptor and competitor's offerings. By definition, a SaaS delivered version of the traditional on-premise feature set will not satisfy all existing customers, as a SaaS offering will be lower function by definition in the short term. Mismanaging the timing of such a deliverable can leave customers in the position of believing that the on-premise offering is dead or dying and that the SaaS offering is not fully baked - this can then lead them to evaluate competitors offerings. This necessitates a focus on a non-overlapping market, over a timeline established by strong research of the changes in customer demand in one's own market and similar adjacent markets.

- ***Over time, we will repurpose our existing products to integrate in value-driven ways with the new SaaS based platform's functionality***

The second market oriented section of the mission statement speaks to how other products developed by the firm should prepare for the eventual transition. They should not seek to re-

implement their existing functionality immediately, but instead seek to create linkages between the value of the emerging platform and the existing product suites – understanding that existing customers will not be fast to move off of their current on-premise solutions, but will be more likely to stay with the firm if they see a path towards increased value from SaaS adoption and an increase in competence that the firm is building internally.

The last two parts of the mission statement relate to the product technology and the organization responsible for delivering on the development of that technology:

- ***We will leverage lead users and innovation communities to design and build the new SaaS platform, driving network effects, iterating rapidly and harnessing innovation capabilities.***

This product development oriented section focuses the learning experiment required to pick up the skills to deliver software in a service oriented manner. By leveraging lead users, the firm can not only learn the strong needs of users who wish to adopt software as a service, but can benefit from the fast cycle times that they will require, as well as the potential solutions to complex problems that they may have developed in the area. Because they are not focused on becoming a platform, instead on solving their unique needs - the barrier to open sharing from these users is lowered. This will provide an ecosystem and community to work with highly motivated stakeholders to help resolve the gaps in service delivered software.

- ***We will focus on open and modular platforms to enable us to make the transition from our on-premise technology to new SaaS offerings in the most cost effective and control oriented way.***

This final section of the mission focuses on the architectural choices that will the development team should use to help make the transition. In particular, open technology will help to empower the lead users from the previous part of the mission, building a larger and more dedicated user community. Modularity will also empower lead users to create innovation on top of and around the platform, while at the same time providing opportunities to swap out commoditized parts of the technology stack.

The sum of this mission statement provides the structure within which the firm will execute its transition. The leadership of the firm should use this statement to both inform external activities like advertizing, marketing and business development by setting appropriate messages and creating incentive programs.

3.3.4-Strategic Goals

The last facet of the strategic framework is the set of strategic goals that will be executed to help shift development resources to adopt the disruptive SaaS technology over time. Strategic goals are specific deliverables that the firm must execute on in order to meet the mission statement. Each strategic goal will be a structured exercise in itself, and will contain each of the following:

- *A Deliverable* – ranging from a business case, to a whitepaper, to an organizational chart, each strategic goal will result in a specific deliverable.
- *An Owner* – someone who is responsible for the end-to-end completion of the strategic goal, and is responsible for the delivery of the required deliverable.
- *A List of Stakeholders and Responsibilities* – A set of people who are essential to the creation of the deliverable, and their role in the creation.
- *A Process* – A recommended method for completing the deliverable, which is to be completed by the owner and stakeholders.
- *A Timeline* – A recommended timeline to complete the deliverable, or a method to establish a timeline.

The strategic goals will help the firm to build a more complete picture of the threat faced by SaaS and Cloud, and provide the firm with an appropriate sense of urgency with which to set a timeline for transition. These goals will help select a market where experimentation with SaaS concepts should be incubated, and set expectations for the development organization to ready itself to integrate with the eventual SaaS deliverable. They will also provide guidance on how to best innovate and iterate on ideas with lead users, and how to structure the organization and enhance its skills. The strategic goals will focus on the following three broad areas:

- *Market Research* – Goals that will help to assess, define and target markets.
- *Product Development* – Goals that will influence product technology and its development
- *Organizational* – Goals that affect organization structure, skills and other personnel concerns

In the following sections, these broad areas will be fleshed out into a set of strategic goals.

Market related goals

The market related goals are the responsibility of the business development, product management and strategy personnel in the firm. Completing these strategic goals will help to understand the pressures from adjacent SaaS markets, the performance characteristics and identify markets where the firm can do experiments related to the performance characteristics that customers will expect from the firms SaaS based offerings. Much of this market oriented strategy is based on the methods of assessment presented in Christensen's Innovator's Dilemma and Innovator's Solution, coupled with an approach for identifying lead users in niche markets based on von Hippel's research.

Goal #1: The firm will assess critical performance characteristics in the existing value network and how they are changing over time.

The first goal that must be undertaken is to build a complete understanding of what set of performance characteristics are valued by customers of the firms product set, and how that set of characteristics is ordered and changing over time.

Deliverable: Value Network Analysis (Own On-Premise Markets)

Owner: Senior Management

Stakeholders: Product Development Staff (research)

Timeline: Short (<1 month)

The process for executing on this strategic goal is to perform a value network analysis as recommended by Christensen. This method was described generally for enterprise software in Chapter 2, but the firm should perform a specific study on its own market(s) and customer base. The output should be a study that specifically articulates the performance characteristics that the customers in the firm's markets, presenting them in an ordered set which captures how they have been changing in recent history.

Goal #2: The firm will identify and evaluate adjacent and potentially disruptive SaaS markets; assessing their critical performance characteristics and how they are changing over time.

The second strategic goal that the firm will undertake is to actively research other nearby markets that have already adopted SaaS and Cloud delivery, and perform a value network analysis of their products.

Deliverable: Value Network Analysis (Adjacent SaaS/Cloud Markets)

Owner: Senior Management

Stakeholders: Product Management Staff (research)

Timeline: Short (less than 1 month); parallel with Goal 1

As with Goal 1, the firm must also undertake the process to identify and evaluate markets where SaaS has been adopted. The firm should create value network analyses for these SaaS markets, and document the performance characteristics of customers in these markets. The firm should cast a wide net, and do broad research initially – the more markets that are sought out, the more likely that that firm will find markets where it can start to grow its competency in the new delivery methodology.

Goal #3: Select an appropriate niche SaaS market to target the firm's SaaS platform efforts – by definition, these are *NOT* existing customers.

The final market oriented goal is to use the information gathered in the first two market goals to help identify areas where the firm should invest effort in identifying lead users to help rapidly refine needs and locate solutions.

Deliverable: Target Market and Justification

Owner: Senior Management

Stakeholders: Product Management Staff (research)

Timeline: Short (<1 month); after Goals 1 & 2

After identifying the ordered set of needs in its own market and those in the markets for SaaS platforms, the firm should perform pairwise comparisons between each of their performance characteristics and each performance characteristic of each adjacent market. The comparisons will result in 4 distinct sets:

1. *Both Firm & SaaS possess (potential disruptive characteristic)*

2. *Firm possesses, SaaS does not possess* (SaaS feature gap)
3. *SaaS possesses, Firm does not possess* (Likely not a disruptor)
4. *Neither possess* (Likely not a disruptor)

The most important set to examine is the first set of potentially disruptive characteristics, especially if there is a significant number of overlapping characteristics. A product in the SaaS market that satisfies this set of characteristics represents an adjacent technology market where a competing firm could move into the firm's Enterprise market given the proper resources - either directly or through technology licensing or other partnerships. Disruptive technologies are those which do not currently satisfy the needs of a firm's existing customer set and are thus not interesting, but may do so in the future. This means that where there is an overlap between one of the firm's key performance characteristics and a market where the same characteristic has a lower ranking should be carefully evaluated. It is in these markets and characteristics where the firm should seek out lead users, which will be captured in the next section. These characteristics will be referred to as 'SaaS Disruptors'.

The second set of characteristics represent areas where the on-premise delivery mechanism still has an advantage over a SaaS delivered option. In order to eventually be performance competitive to on-premise software delivery, a SaaS offering will need to overcome these feature gaps. This is the second area where the firm should invest in looking for lead user. These characteristics will be referred to as 'SaaS Feature Gaps'.

Product Development Goals

Strategic goals in the area of product development are primarily targeted at the development community – line management, architects, developers and testers who create the products and services that are delivered by the company. They are necessary to help create a structure to embrace SaaS delivery, to ensure that proper platforms are defined, and that realistic development plans and architectures can be executed. They follow from the market goals, and are to be informed by the results of the investigations that were outlined in the previous section on Market Goals. The Product Development strategic goals that are defined are as follows:

Goal #4: The firm will define an open, modular architecture which maintains the ability to open strategic parts of the platform. The firm will create processes for deciding what is open, and what is not.

Modular architectures allow firms to make dynamic decisions about the evolution of the platforms that they are building. Mixing in concepts of open innovation allows the firm to focus on creating innovation at the non-commoditized part of the platform, as well as enabling the wider adoption and broader innovation on top of it.

Deliverable: System Architecture Documentation

Owner: System Architect

Stakeholders: Product Management (markets) and Development Staff (technology),
Legal Staff (licensing)

Timeline: Mid-term (>1 month); after Market goals completed

Once the target market(s) for lead user investigations have been identified, the firm must begin the process of creating a SaaS platform that allows those lead users to build and innovate. The core values of modularity and openness will allow the firm to initially attract and subsequently sustain the broadest set of lead users. As the firm refines the architecture with these lead users, it must seek to learn and acquire the core technologies that allow a SaaS platform to scale and grow to eventually allow delivery of the firm's more traditional enterprise applications.

The system architecture for a SaaS platform must adopt modularity as a core concept, allowing for the variety of platform strategies outlined by Cusumano. The key benefit of modularity is that it allows the firm to replace parts of the platform with different systems using identical interfaces, potentially without impact to uptime and stability. Modularity also allows for innovation on top of the platform, allowing complementors to create additional value for both themselves as well as the platform itself.

Consider the challenge for the firms who adopted the SaaS model of software delivery in the late 90's- which proved to be the inability to meet the customization needs of their customers and still be able to scale at cost to satisfy many customers to cover their initial capital investment. This was due to the fact that the stacks of enterprise software that they were deploying were not built with multitenancy in mind, and ASP firms did not have the explicit ability to modify the code or enlist contractual changes to turn around the feature gap in time to survive. Without having proper modularity, these firms were locked into platforms that they could not change – which resulted in failure.

Open technology provides additional benefits for collaboration with both competitors and complementors. In the consumer market for SaaS products like those offered by Google, Amazon, Ebay, Facebook and others, each has built their individual stack to overcome this challenge. In many places, these companies have built on open source stacks or contributed portions of their own into the wider community for enhancement. As with the basic infrastructure of the open source Apache HTTP server, there is greater benefit to more eyes on the service delivery challenge than there is to keeping that innovation internal. As recommended by Cusumano's first lever regarding the scope of the firm, each firm keeps their core value technology internal but works in collaboration with the ecosystem to solve basic delivery challenges, as a larger web service ecosystem presents greater opportunity even for competitors. For Google, this is their search algorithm; for Ebay, their network of bidders and sellers; for Amazon, their retailers and delivery network. This change to the value structure of online services is different from the days of purchasing the lower parts of the stack from a vendor – open source has effectively commoditized the basic infrastructure, allowing firms to focus on higher value deliverables.

Enterprise firms who are preparing to make the transition to SaaS delivery must carefully assess whether to retrofit multitenancy, virtualization and metering into their existing

product lines or build entirely new stacks. In reality, the true solution will likely be a blend, but the open activity in the other SaaS markets offers an opportunity to build capability in this new delivery model and benefit from the innovation and best practices that have evolved as part of those markets.

Goal #5: Find and interact formally with lead users, both in the existing customer base and in the selected SaaS niche markets

Lead users can help the firm to find both needs and solutions to the challenges faced in areas of relevant innovation. The firm should harness these lead users to refine ideas in specific areas of the technology platform.

Deliverable: Lead User Studies

Owner: Product Manager

Stakeholders: System Architect, Product Development

Timeline: Short (< 1 month), repeating; after Market Goals, parallel with Goal 4

As described in the background research, lead users are users who are strongly experiencing needs that represent a future market with a larger size. In addition to fully understanding those needs, they may also have solutions already developed and may be willing and likely to share them with a broader community.

In the market goals, the firm identified needs that could be disrupted or should be met in order to make the firm competitively capable for SaaS delivery and fill gaps that SaaS currently has against the current on premise products. This exercise also allowed the firm to identify markets where overlapping needs were strongest. The firm should use these markets and needs to identify lead user. Key to the effort is to finding the right lead users, which is an exercise in reaching into the existing selected SaaS market(s) and finding lead users, or using the identified needs to find yet other markets where the need is most extreme (SaaS based or otherwise). Once these lead users area identified, the firm must perform structured lead user studies as described by von Hippel in his Lead User study guide (Churchill, Hippel, & Sonnack)

Goal #6: Establish or embrace innovation communities for both sets of aforementioned lead users

After the firm has identified and studied the lead users in the SaaS markets (or other markets with extreme need), it must find or create communities where it can maintain that relationship and draw more lead users from the SaaS market.

Deliverable: Community development plans

Owner: Product Manager

Stakeholders: System Architect,

Timeline: Short term (<1 month) then ongoing; after Goal 5

As described by von Hippel, innovation communities are the places where lead users interact and share innovations. In the world of software, examples are becoming more frequent and familiar. The most classic example is the Linux platform, developed and maintained on a distributed network of source code repositories and mailing lists. Another is the Apache Software Foundation, which serves as the innovation community for the popular Apache HTTP server.

Tapping into innovation communities can provide firms with a place to seek out needs and solutions from users who are willing to share them. With a proper platform and toolkit strategy, lead users are capable of working together and with the firms to create innovations.

The firm should put together a plan on how to facilitate the interactions with the lead users in the target markets. This should include whether appropriate innovation communities already exist where the firm should begin interacting, or if the firm should create new innovation communities for the markets. If creating new communities, the firm should decide how users will join, contribute and potentially be rewarded if appropriate.

Goal #7: In both the emerging platform and existing products, create open toolkits and allow the innovation communities to use and extend them.

In support of the innovation communities identified or created in Goal #6, the firm must make sure that community is bootstrapped with enough initial content that it can be successful and grow. One important part of this content is freely available toolkits that allow developers and customizers to interact with the system.

Deliverable: Open Toolkits and Documentation

Owner: Product Development

Stakeholders: Product Management

Timeline: Mid-term (>1 month) then ongoing; after Goal 5, parallel with Goal 6

As described by Cusumano's research on platform adoption and examined in Goal #4, it is important to define exactly the scope of the firm versus the scope of complementors. SaaS and Cloud platforms are billed by usage or subscription, so the goal is to have subscribers see the value of the platform and adopt it as widely as possible. The stickiness and network effects of a SaaS/Cloud system are generally based on having more users and more customizations for those users, and a strong open innovation community can help drive those effects. This will help to create loyalty in the ecosystem, and allow value linkages to be created by 3rd party innovators. This means that toolkits used to interact with the system should be as open as possible, so that the highest number of client endpoint types can be targeted.

This deliverable is an architectural document that describes the interfaces to the system, and the various methods that will be used to access them. Examples would be desktop browser access, mobile client access, and potentially other client libraries written in native operating system languages. Libraries that are written for native operating systems or helper libraries for mobile or desktop browsers should be freely available and documented in the innovation communities, so that users can download them and easily integrate into the platform in ways that they find useful.

Lead users and innovation communities can take root in this type of open environment, and there are many examples of this phenomenon. Consider the example of CouchDb, a Documented Oriented Database system hosted at the Apache Foundation. CouchDb exposes its documents over HTTP, using an architectural pattern called "REST". Accessing these documents with a browser is straightforward, using the native support for

HTTP built into the platform. However, there are cases where a developer may want to script the system using a higher level language like Ruby, Perl or shell script. One library called CouchDb-Ruby was developed and released under a liberal license, allowing a proliferation of client libraries to be built in other similar languages.

As in the example, prototypical libraries should be created to seed the innovation communities that were defined in Goal #6, allowing the community to tailor and modify it to support their own use cases. Automated systems to allow for the creation and support of additional complementary libraries should be developed.

Also, these toolkits must be complemented by other assets that allow the innovation communities to better interact with the system. In particular, one would be open specifications for a set of remote interfaces that are able to change and evolve rapidly with the needs of the niche lead users. Another would be trial access to the system, and controls that allow for the simple customization of the parts of the system that are not freely modifiable.

Organizational Goals

Organizational goals are intended to help shape the organization and its skills to support the aforementioned product development goals. Execution on the organizational goals requires that the market goals are completed and that the product development goals are in progress or complete, as the organizational goals build on their output. The strategic goals related to Organizational concerns are as follows:

Goal #8: The firm will create an appropriate organizational structure to develop the new platform and its niche applications.

A familiar quote in the software industry is that “Architecture follows Organization”. This means that the firm must create organizational structure that supports the architecture separations that it wishes to create in its product platforms.

Deliverable: Organizational Design Document

Owner: Senior Management

Stakeholders: Development Staff (architecture)

Timeline: Mid-term (>1 month); after Goal 5

In Goal #4, the architecture for the modular platform was defined and decisions were made about what would be open for the innovation community and complementors modify and what would be kept internal as value. Per Cusumano, the firm must make sure to build appropriate organizational walls so that the firm is consistent with its complementors.

As a disruptive technology, the platform itself should be developed in such a way that the organization with the charter to develop and deploy the SaaS offering does not run up against entrenched values in the existing organization. To that end, a new organization for the platform should be created which does not embrace any of the core values or processes of the traditional organization, though it may leverage some of the existing resources. Additionally, senior management for this organization should come from a leader with a fresh perspective on the new delivery method.

For those areas where work is being done fully in the open, such as the open toolkits, particular care must be taken to build organization and leadership that can support such a mission. Due to the diffuse nature of value recapture from open development in a traditional firm, it can become hard to justify expense if not properly managed. This means that the mission should live in one organization, with a leader that understands how to articulate the importance, performance metrics and progress of such work.

Goal #9: Where the firm has adopted open technologies, it will actively seek to build internal skills through training, hiring or acquisition.

Making the move to open technologies as an underlying stack means that firms are relying on technologies that they did not create themselves. This means that they must build skills that can support the fine tuning, diagnosis and changes to the underlying technologies that they have selected.

Deliverable: Hiring and Training Plans

Owner: Development Management

Stakeholders: Development Staff (recommendations)

Timeline: Ongoing; after Goal 5

As discussed in Goal #4, companies both large and small are building complete stacks from the ground up and collaborating on the commoditized portion of the stacks in the open. The firm must make a decision about what underlying technologies it will select. Where the technologies are openly developed, the firm must make a commitment to ensure that they will maintain proper skills in the organization to make their offering function and diagnose problems in the core. At a minimum, this means training for the part of the organization that supports the stack, and potentially reaching out and hiring members of the communities that exist around core technologies. As recommended by Cusumano, avoiding sending a message of overt control is important, so care must be taken not to over-hire from a selected technology – this will shrink the potential of the community as existing contributors become wary.

3.3.5-Conclusions

As network availability has increased and browsers have become more capable, Software as a Service and Cloud Computing delivery methods are encroaching on Enterprise software markets. Established players must react to quickly build competency in the delivery mechanism while not confusing their existing customer base with duplicate but lesser performing products.

Strategies for overcoming disruption involve careful selection of markets and properly aligning them with organizations with the right customer base, size, and values. Paired with an iterative methodology of trial and error, firms can embrace disruptive technologies and survive transitions to new generations of technology.

This chapter presented a strategic framework for an Enterprise software firm to adopt disruptive SaaS delivery, building on platforms and open innovation to enable the same qualities as those recommended to embrace disruptive technology. Through careful selection of markets and lead users, a firm can develop an open, modular platform which can satisfy the needs of niche markets. This strategy will allow the firm to avoid confusing their existing markets, and create the potential to do rapid learning exercises with an innovation community of lead users.

The next chapter will present a case study on open innovation by IBM Lotus and their Notes/Domino product suite.

THIS PAGE INTENTIONALLY LEFT BLANK

4- Case Study: IBM Lotus Notes/Domino

Over the past several decades, the widespread use of computer technology and related software has resulted in significant growth in productivity in the workplace. Productivity in the areas of communication and collaboration has greatly benefited not only from better technology on the desktop, but also from the emergence of global internet connectivity. Starting from simple text based methods like bulletin board services, newsgroups and email, collaboration technology has become essential to the livelihood of a large number of businesses. In the workplace, collaboration technology is used by many enterprises to ensure that their workforce is well connected and working together. As changes in collaboration technology have been driven by improvements in both processing power and connectivity, firms competing in the marketplace for these offerings struggle to keep up with the changes.

One of the key technologies in the marketplace is email, which has evolved and standardized over time into a familiar offering which is used by almost every computer user. The market for email in developed economies is large and competitive, with the major players in the market being Microsoft and IBM. Each of these vendors makes their revenue based on license sales of email software, including rich client software that is installed to client desktops – specifically Microsoft Outlook and IBM Lotus Notes.

Major shifts in this and other marketplaces for network connected services are underway, including the emergence of subscription based, browser delivered software access called “Software as a Service” or SaaS. The major driver for this shift is the nearly ubiquitous availability of high bandwidth network access, as well as the increasing capability of browsers to deliver rich and dynamic user experiences. This work will evaluate the strategic options for an established player, IBM, in the face of the disruptive force of SaaS. This section will begin with a brief overview of the enterprise email offering from IBM, Lotus Notes.

4.1 - Lotus Notes the 'Groupware' Platform

Over 20 years old, Lotus Notes is an IBM software platform historically referred to as "Groupware" which is used to facilitate collaboration in an enterprise. Version 1.0 shipped to customers in 1989, and the install base has grown to over 145 million installed seats as of the most recent version, Notes/Domino 8.5 (IBM). It is most often deployed in enterprises as an email client, allowing users to send and receive email messages, manage calendar workflow and store their business contacts.

Notes is not simply an email client, instead it is a platform whose core design is centered on the concept of databases. A typical relational database presents structured data to the system in the form of tables and columns, which require strong typing and fixed size fields (Wikipedia). Lotus Notes is more commonly referred to as a 'Document Oriented Database', which instead models its data as documents with an arbitrary number of arbitrarily typed fields (Wikipedia). Each database contains some number of unstructured documents (called 'notes') which can be aggregated in different ways based on their contents via a concept called 'views'. As an example, a collection of documents containing user's names can be exposed as a view which allows the application to sort or categorized those documents based on the user's name. With Lotus Notes, this schema-less approach has been paired with the Designer product, a WYSIWYG editor for the simple creation of 'forms' – a user interface construct that renders a document and provides features like field validation and advanced user input. In Lotus' experience, this simplicity and a decentralized nature has allowed Notes to be used by line-of-business administrators to create simple but useful applications to support their business needs.

Using the basic building blocks of databases, notes, forms and views, the product supports deep customization of document based workflows. As a reference example, the prototypical email experience in Lotus Notes is built on these concepts, as pictured below in Version 1.0:

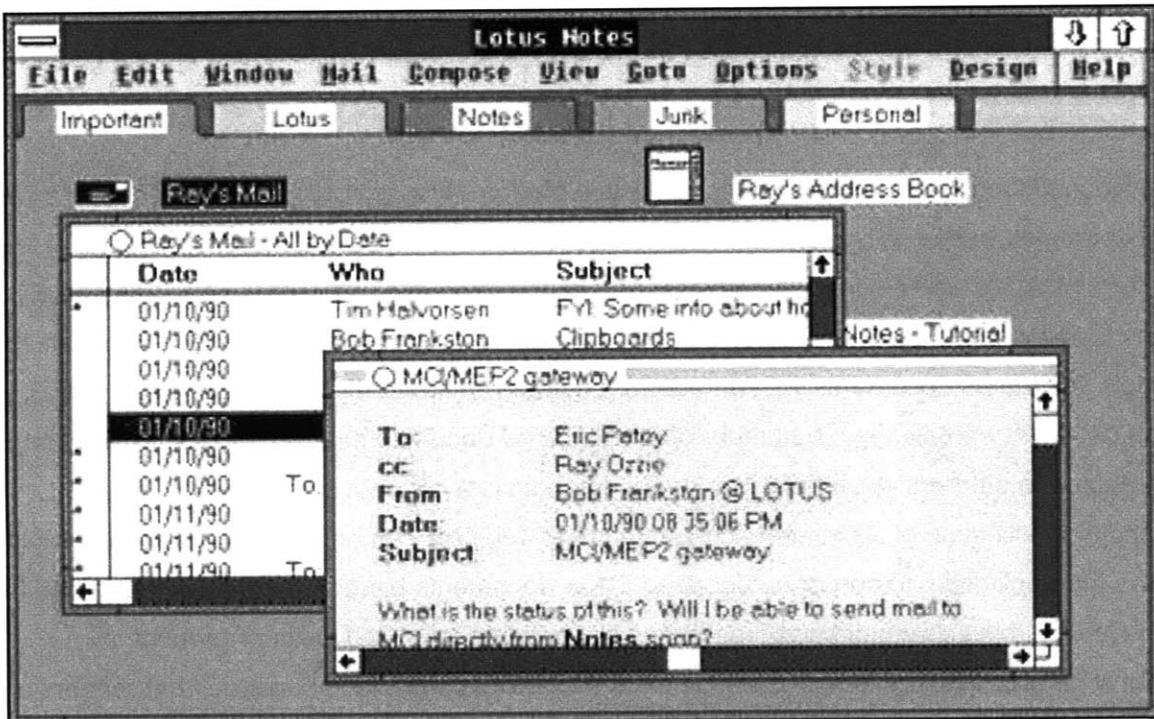


Figure 4.1 – Lotus Notes Version 1.0

When the early versions of Lotus Notes were written, a great amount of the standardization that has been a driver to the success of the internet had not yet occurred. For instance, TCP/IP had not yet emerged as the networking protocol of choice – therefore the Notes platform was capable of communicating over many different types of protocols, including Banyan VINES, Novell IPX/SPX, TCP/IP and others. Above these base protocols, there was no higher level protocol like HTTP- thus Notes had its own higher level called 'Notes Remote Procedure Call' or NRPC. Notes allowed companies to bridge various networks, creating a Notes network layered on top of a company's various IT departments' network choices – it acted as networking glue.

As most networks were very slow and low bandwidth, Notes was designed from the start to enable efficient use over these types of connections. This meant that data was easily stored offline for disconnected use, and that an algorithm for efficient synchronization was developed

called “replication”. This allowed the product to be used in many interesting ways, a favorite example being the use as an email system for a Navy ship that was connected to the outside world via a slow, unreliable satellite link which was only available for short periods of time.

As with networking, security protocols and features had not yet standardized and cryptography had just taken a leap forward with the 1977 publishing of the RSA algorithm (Rivest, Shamir, & Adleman, 1983). Lotus Notes took advantage of this technology and was one of the first applications to surface a full featured Public Key Infrastructure based on RSA. Every user who uses Lotus Notes has a public and private key, which can be used to perform critical security operations such as authentication to servers, encrypting documents and creating digital signatures. This fundamental feature is used throughout the product to provide a highly reputable level of security. Beyond basic PKI, the now- familiar Secure Sockets Layer did not yet exist, and Notes implemented a similar proprietary encrypting protocol.

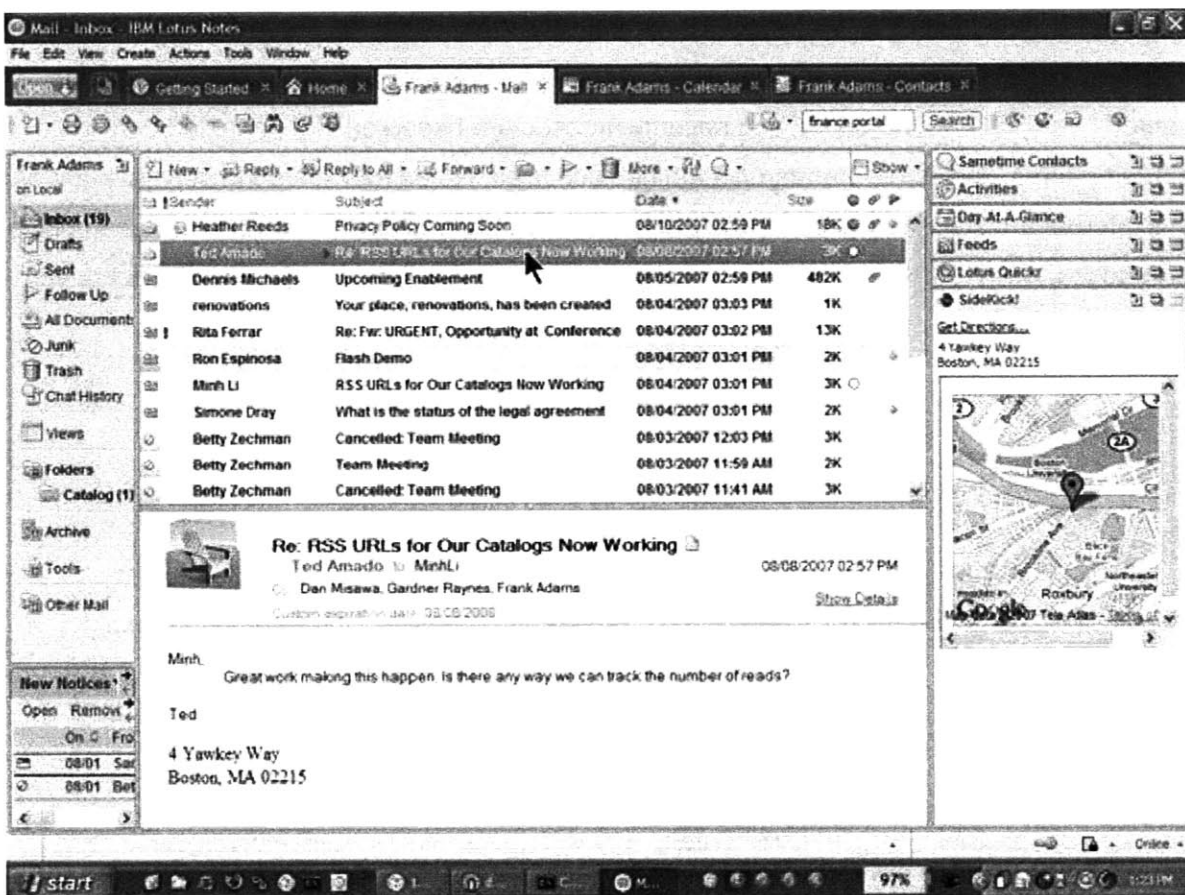
In essence, Lotus Notes was a capable browser platform before the internet had standardized into what is now familiar to the world. Many of the abstraction layers that exist in the Notes architecture have analogues in what exists currently in browsers, from XHTML documents served over HTTP to the underlying security architecture based on RSA-enabled certificate infrastructures. While this standardization has occurred, Notes has strived to stay relevant as this internet architecture emerged. Staying away from the browser war which has taken place around it, Lotus Notes has remained competitive in an area where browsers have not driven down the cost of the software – as a premier E-Mail platform.

4.2 - Lotus Notes the E-Mail platform

Most users of Lotus Notes are familiar with it as an E-Mail client. As mentioned earlier, the email client of Lotus Notes is built upon the same fundamental platform concepts of databases, views and notes, except in the email case the user's mail database contains email messages with fields like 'Subject', 'Body', etc. Over the years, the user interface and features that are supported in the email experience have evolved in a way such that Notes remains competitive with its closest rival, Microsoft Outlook.

Traditional thick-client email applications allow a user to manage their emails, perform calendar workflow, and maintain a list of contacts. The following is a screenshot of the email experience

Figure 4.2 – Lotus Notes Version 8.5



in the Lotus Notes client version 8.5:

Although Lotus Notes was one of the technologies that defined the Email market, competition and evolution in standards are actively at work shaping the competitive landscape. In the 80's and 90's, email protocols such as IMAP, POP3 and SMTP emerged, allowing interoperability amongst clients and servers from different vendors. The lightweight directory access protocol (LDAP) emerged as the protocol of choice for directories, allowing companies to serve their organizations directory information from a centralized repository. Multipart internet message exchange (MIME) emerged as the transport format of choice, and X.500 internet addresses became the method by which different "walled gardens" of email offerings began to interoperate. Although avoiding the browser platform wars by accommodating all browsers and integrating browser based technology, Lotus Notes has actively participated in the evolution of the messaging market – adopting and exposing standards where appropriate, and focusing on customer pain points elsewhere, such as calendar workflow, deployment mechanisms and management utilities. In the Enterprise messaging market, this strategy has been relatively successful – the mature market has grown sizably over the last ten years and stabilized with IBM and Microsoft as key players sharing the bulk of the market.

4.3 - Complementary Technologies

Many complementary technologies have emerged around this messaging base as computing power, network bandwidth and availability of connectivity have increased. They tend to differ on a variety of criteria – synchronous vs. asynchronous, directed vs. broadcast, and on other axes of communication type.

One of the most widely used complementary technologies is the familiar "Instant Messaging" client software, where IBM Lotus has the Sametime product offering. Although not as widely interoperable as email, Instant Messaging is used in many enterprises and by millions of consumers to facilitate immediate interaction between two persons at a computing device. A particularly strong area for Microsoft is the Productivity Suite, where their ubiquitous Office product is the flagship offering. In this space, IBM offers Lotus Symphony – a free office suite based on the open source OpenOffice offering. Related to productivity suites is the area of file sharing, where content storage servers offer access models that allow users to share and

author collaboratively. In this space, IBM has the Lotus Quickr product where Microsoft has Sharepoint. Voice over internet protocol (VOIP) is another complex adjacent area where collaboration companies are competing – IBM's offering in this space is Sametime Unified Telephony. A final emerging area is Social Software, a new market segment based around bringing the value perceived in social network data, such as the ability to identify “gatekeepers” per the work of Tom Allen in his book “The Organization and Architecture of Innovation: Managing the Flow of Technology” (Allen & Henn, 2006).

4.4 - Disruption in the Enterprise Messaging Market

As mentioned at the start of this section regarding their experience in the e-Business initiative of the 90s, IBM is not a stranger to the concept of disruptive technology and the methods by which the challenge can be surmounted. Some of this learning was presented by Irving Wladawsky-Berger, former IBM Vice President and visiting MIT professor, in his class (Wladawsky-Berger, 2007).

The Lotus brand participated in this initiative by adopting the standard internet technologies of the time, modernizing some of the interfaces to support protocols like HTTP, IMAP, POP3, SMTP, LDAP and others. Over the last 10 years, the internet has continued to evolve and is now even more a critical part of the fabric of everyday life. Email is a commodity for consumers and is given to them for free in return for advertising dollars from services like Gmail, Hotmail and Yahoo. Social networks like Facebook, MySpace, Twitter and others are a growing destination, especially for the younger generation.

The challenge for the Lotus brand is not only on the delivery side, where an interest in a transition from on-premise to service based delivery of collaboration services is among the highest demand categories for SaaS delivery. Lotus also faces a challenge that is unique within the IBM software group. As the most end user facing product group in the IBM software group, Lotus has to deal with the additional disruption that is emerging from the aforementioned penetration of consumer collaboration services into the end users' daily life.

Lotus makes the bulk of its revenue from license sales of the flagship Lotus Notes and Domino platform. As mentioned earlier, Lotus Notes and Domino is a groupware product but is primarily purchased for corporate email usage. Email access is practically ubiquitous to consumers and seamless browser and mobile access is an expectation that is set in the consumer market. This means that the consumer Email providers are increasingly setting the performance, user experience, integration and mobility standards for end users in the enterprise. Those providers are also at an advantage in the technology required to deliver that function in a scalable way. Compounding this is the fact that in younger demographics, Email is commoditized to the point of being a point-to-point messaging feature embedded in social network user experiences. Simple changes to provide an email address and a personal contact list function would allow the social network providers to launch an enveloping platform attack on the consumer email market.

As covered in the previous chapter, intersecting with this shift in end user expectations is an increasing influence of end users on the decision making processes that result in the selection of enterprise collaboration technology – a trend sometimes more broadly referred to as “Consumerization of IT”. They have expectations based on their consumer email experience. Also, consumers are purchasing their own mobile devices with advanced internet capabilities, and want to be able to connect them to enterprise networks. IT software buyers see this as an opportunity, as it shifts some of the capital costs of purchasing devices to their employees. On the other hand, it presents a challenge in standardization of mobile platform, as mobile ecosystems are not inherently compatible and the skills to create applications and experiences for the various devices are fragmented. Beyond complexity and standardization, security can also be an issue that requires remediation.

4.5 - Strategic Framework for IBM Lotus Notes & Domino

With the backdrop for the environment which Lotus is competing in described, a discussion of the current state of Lotus' transition relative to this strategic framework will follow. This discussion is informed by the author's experience, as well as discussions with several senior leaders in the Lotus organization.

Lotus has not explicitly set an internal vision and a mission for the transition to SaaS based delivery. However, several different initiatives are in progress related to this transition and have relevance to the strategic framework discussion at hand.

The following section will describe the steps that Lotus has taken to adopt the disruptive delivery model of SaaS, and will also provide recommendations that reflect the key of this work where appropriate.

4.6 - Strategic Goals

The vision and mission statement for the Lotus Notes and Domino product suite to adopt SaaS delivery remains unchanged from the boilerplate defined in Chapter 3. Lotus has several different initiatives to embrace SaaS, and a meaningful amount of platform strategy and open innovation methodologies at work, but has not actively engaged them as part of a integrated initiative. This section will use the lens of the strategic goals framework from Chapter 3 to assess progress relative to these areas and provide recommendations.

4.6.1 - Market related goals

The following are the market related goals that are executed by product development staff:

Goal #1: The firm will assess critical performance characteristics in the existing value network and how they are changing over time.

Evaluation: From speaking to leaders in the Lotus organization, a set of performance characteristics that are articulated by customers emerged. These characteristics are largely refinements, and sometimes completely identical, to the characteristics broadly defined in Chapter 3 as SaaS was evaluated more generally in the Enterprise market. The following are some of the characteristics that were identified as critical to customers in the Enterprise Email market:

1. *Value (Familiarity/Usability)* – Email is such a pervasive tool, that value is largely defined by how familiar the product is to the dominant design for Email client software (as represented by Lotus Notes, Microsoft Outlook and Mozilla Thunderbird), and by the usability and functionality of both the end user, developer and administrator tools.
2. *Time to Value* – As in Chapter 3, customers care about the ability to quickly bring the value of the Email system to their user population. This means that it must be easy to install and roll out to large numbers of users, both at the server and the client.
3. *Integration & Extensibility* – Typically, customers are integrating their new email system into existing IT infrastructure. Thus, they care about support for things like operating system support, feature integration with the native platform, and the ability to extend the platform with additional features and customizations.
4. *Security* – As a platform, customers care about the ability of an Email solution to protect itself from attacks and viruses on the platform, as well as the ability of the platform to protect itself.
5. *Availability (Disconnected Use)* – From a client perspective that began when internet connectivity had not yet evolved to be almost ubiquitous, availability meant

the ability to work with Email while disconnected (such as while on an airplane). This use case still exists and is popular with Lotus customers.

Goal #2: The firm will identify and evaluate adjacent and potentially disruptive SaaS markets; assessing their critical performance characteristics and how they are changing over time.

Evaluation: For the purposes of this case study, a brief evaluation of the consumer Email market was performed. Some of the key performance characteristics that emerged in this area were:

1. *Value (Familiarity/Usability)* - A dominant design for consumer web based mail has also emerged, as well as a consistent feature set including features like email signatures, anti-spam, folder management and others.
2. *Availability (Uptime/Scalability)* - The expectation of consumer email services is that the service will be available and capable of supporting workloads that scale to massive amounts of users.
3. *Time to Value (Ease of Access)* - In order to keep users coming back to support the advertizing based revenue model. No client software is required for the user to begin working with the email system, so that they can access it from anywhere and interact in a way that does not require the management of desktop software.

Goal #3: Select an appropriate niche SaaS market to target the firm's SaaS platform efforts – by definition, these are *NOT* existing customers.

A comparison of the performance characteristics results in two sets of characteristics that need to be considered. The first is the SaaS Disruptors:

1. *Time to value (Familiarity/Usability)* - Desktop email faces the fact that consumer email is driving a new dominant design as more and more consumers adopt the free services.. This has been ongoing for quite some time, so most of the major

vendors have built web based interfaces to support on-premise enterprise deployments. Because customers have been asking for this new dominant design for some sets of underserved users in their enterprises, this is consistent with the existing firms a sustaining innovation

2. *Availability (Uptime/Scalability)* – Customers have not been asking for their services to support the types of workloads that are common with the consumer email services. Also, the operational and capital investment changes required to support running an infrastructure as opposed to selling an infrastructure. As discussed in Chapter 2, the key technologies required here are virtualization and horizontally scaled databases, technologies which are used to scale commodity hardware to support massive workloads. This characteristic should be a key area to develop with lead users.

Evaluation: The recommendation would be that Lotus should take this virtualization technology and database configuration and apply it in a market different from email. To that end, Lotus has chosen Web Conferencing with the LotusLive offering. Over time, the LotusLive platform will evolve to support other services in its own datacenters, including email.

The second set is the SaaS Gaps, areas where Lotus should develop in a way that allows them to take SaaS offerings and make them competitive with current technology, so that any converged offering of SaaS plus an offering that closes these gaps will be market viable.

1. *Security* – Security is one of the most mentioned concerns as SaaS is considered by enterprises. Moving a customer's data to a service providers datacenter is a step that they do not consider lightly, and the increased attack surface created by exposing the application over the internet also incurs risk. On-premise solutions using rich clients keep the data in the enterprise, and have long time support for features like encryption and digital signatures.
2. *Availability (Disconnected Use)* – Offline usage is a use case that browser applications in general do not support well. When not connected to the internet and to the specific service, a browser is unable to offer the ability to work with content that is not stored on the server.

3. *Integration & Extensibility* – Browsers are stand alone applications, and thus do not integrate deeply with other desktop applications or the operating system themselves. From desktop icons, to event notification to multimedia support, browsers allow for very little besides interaction within the HTML chrome. This prevents many common use cases like branding, kiosk mode and others.

Evaluation: These Gaps have already coalesced into a platform competition for a class of products called “Rich Internet Application”, or RIAs. Offerings from Microsoft (Silverlight), Adobe (AIR), and Mozilla (Prism) are essentially browsers with extensions to fill these gaps. Lotus should become active in this platform war, and either choose a platform or develop one of their own on top of or beneath their Notes & Domino base.

4.6.2 - Product development goals

The following are the product development goals that are performed by the development staff:

Goal #4: The firm will define an open, modular architecture which maintains the ability to open strategic parts of the platform. The firm will create processes for deciding what is open, and what is not.

Evaluation: The legacy codebase of Notes/Domino described in the introductory section does not have a particularly modular architecture - the product itself is monolithic. Over the past several years, the product adopted the open source Eclipse Platform as an underlying base, whose core differentiator is its ability to be a cross platform integration container. Eclipse applications are written as “plug-ins” which extend and enhance the existing functionality of the platform and applications written on it. This allows user to extend the Notes platform with open source complements, or to write applications that blend together the proprietary Notes platform with open source complements.

Recommendation: As discussed in the introduction, internet browsers are evolving such that they are almost as capable as Notes/Domino at delivering rich user interfaces. Where the Notes/Domino platform has strong advantages, the team should use a modularity strategy to carve out particular pieces that give leverage against other SaaS competitors, and potentially evolve them as integrations to a browser as extensibility allows. From the earlier discussion of performance characteristics, the focus should be on the SaaS Gaps of offline, desktop integration, security and multimedia.

For instance, Lotus should carve out the database layer and make it a separate architectural component, and attempt to offer the database with compatible programming model interface to that of the open source equivalent, CouchDb, or one of the other competing NoSQL programming models. This allows for ecosystem growth and will help enlist the participation of lead users who are active in the emerging community around NoSQL databases. The same modularity exercise and niche targeting approach should be taken with the other SaaS gaps.

On the server side, Lotus must continue to evolve the SaaS Disruptor characteristics of scalability and time to value by continuing to adopt the technologies that enable SaaS

delivery, like virtualization and horizontally scalable database technology like that of BigTable and Hadoop.

Lastly, modularity at the interoperability level should also be pursued, meaning that the organization should develop an interoperable application development model between the products in both the on-premise and SaaS portfolio. Also Lotus should create management tools that cater to hybrid configurations of SaaS and on-premise offerings, ensuring continuity of skills.

Goal #5: Find and interact formally with lead users, both in the existing customer base and in the selected SaaS niche markets

Evaluation: IBM Lotus does a solid job of finding vocal users in its own business partner and customer communities, and interacts with them regularly. The Lotus Business Partner forum is an active community where customers and business partners interact with development and each others to share ideas. The yearly Lotusphere conference also provides a face-to-face opportunity to interact. However, these interactions cannot be categorized as lead user interactions, because there is not a sense of how strong their needs are, and those needs are not matched to the specific opportunities identified in the strategic market goals developed earlier.

Recommendation: IBM Lotus should use the performance characteristics identified as SaaS Disruptors and SaaS Gaps in the market goals to identify lead users that may be able to help (or may already have) developed solutions to the needs. Given the modularity goals above, the organization should target a niche market with the new client effort, and work with the lead users in that niche market to develop the SaaS Gaps into market competitive offerings that can eventually be transitioned back to the mainstream. For example, one potential market could be as a desktop software offering that bridges multiple social networks on the desktop, leveraging new directory standards like OpenSocial.

Goal #6: Establish or embrace innovation communities for both sets of aforementioned lead users

Evaluation: In addition to the lead user efforts described in Goal #5 above, Lotus has actively created a more open innovation community to interact with its customers and business partners called openntf.org (Openntf.org). Into this community Lotus releases code assets and supports them with developer interactions and support. Unfortunately, this effort does not target the SaaS feature gaps and niches and instead is primarily an echo chamber for the current customer base, which has little incentive to move to service based offerings because of the unclear revenue opportunities.

Recommendation: Lotus should use the SaaS Gaps to identify lead users who have the requirement to satisfy each gap in a browser context, and then look for places where the lead user coalesces into a community. Lotus should especially ensure that they are interacting in communities that are not necessarily echo chambers of current customers. One potential opportunity is to leverage the existing Eclipse community, which uses Eclipse as an integration platform for a large number of uses. Consider the case for enhancing security, which is a premier problem in health care settings. Using Eclipse as a health care platform is the charter of the Open Healthcare Framework (Eclipse.org), a potential area to seek out lead users and interact in their community.

Goal #7: In both the emerging platform and existing products, create open toolkits and allow the innovation communities to use and extend them.

Evaluation: As chronicled in the introductory section of this chapter, the Lotus Notes product began its life as a platform. This means that it contains a number of ways to be extended via APIs, which are a form of toolkit. Due to the adoption of the Eclipse platform on the client, customers and business partners can also create applications that integrate with its open source toolkit. This commitment to openness has not followed through into the legacy toolkits; therefore much of the product remains closed and proprietary. Unfortunately, Lotus has not historically excelled at getting trial versions and fully open toolkits into the hands of stakeholders who influence purchasing decisions like end users and developers in line-of-business organizations.

Recommendation: The Lotus team should create more open toolkits, and seek to create positive feedback loops with users in the innovation communities who take on the task of extending and porting those toolkits. In all likelihood, the Lotus team will need to create a

core extensibility layer of the platform, and create a reference toolkit on top of it. In a properly managed innovation community which attracts lead users and is serviced by a dedicated Lotus toolkit team, the reference samples that are released as open toolkits can be adopted and adapted by the innovation community.

In addition to open toolkits, the Lotus team should strive to release free developer preview versions of both their client software and their development tools, and host an instance of their SaaS platform for experiments and innovation.

4.6.3 - Organizational related goals

Goal #8: The firm will create an appropriate organizational structure to develop the new platform and its niche applications.

Evaluation: As Christensen's literature is well read and researched at IBM, and as SaaS has been identified as a disruptive challenge, organizations doing SaaS work at IBM have been separated from the on-premise development teams. This was made publicly known with the announcement made of Kristof Kloeckner's leadership of a new Cloud Computing division (EWeek, 2009). This allows the SaaS teams to operate outside the traditional value systems of on-premise teams, and to create new processes to create and sustain new growth.

Recommendation: Although the development of server infrastructure for SaaS is removed from the core on-premise development teams, a challenge for how to evolve the client infrastructure remains. This client development should be done in a way that targets one of the markets identified by the lead user exercise, and also separates the team from the existing values in the existing Notes organization, although it may leverage some of the same resources.

Also, there needs to be an active effort to identify how to transition customers and business partners from one delivery method to another by creating incentives to transition services or embrace hybrid configurations. Cross-functional integration teams that own the problem of how to create linked value from the SaaS to the on-premise offerings should be created. These organizations could potentially be matrix, leveraging team members from both the on-premise and SaaS teams.

Goal #9: Where the firm has adopted open technologies, it will actively seek to build internal skills through training, hiring or acquisition.

Evaluation: IBM has a strong history of supporting the development of Open Source technology, which also exists in the Lotus organization. From support of platforms like Linux to the adoption of the Eclipse platform, Lotus is a strong consumer of open source technologies. As a business strategy, this allows IBM software to readily integrate with other open technologies and for IBM to then capture value as a business consultancy and

systems integrator. Lotus does support some individuals in Open Source roles, but the diffuse nature of the commitment makes any long term growth difficult.

Recommendation: Although Lotus is a strong consumer of open source, it is not a strong producer. Lotus should consolidate its internal open source developer community under a single organizational umbrella, and enable management to create an environment where a commitment to knowledge of underlying open source platforms is valued and kept current. This will help Lotus to better leverage open source platforms, and build the brands credibility in the open source communities in which it interacts. Being better connected will enable Lotus to gather support if large initiatives need to be proposed and taken on. As more and more of the underlying stack becomes commoditized by open source options, the brand must maintain its competitive capability and ecosystem potential by adopting these technologies, cooperating with communities and maintaining expertise.

5.4 Conclusions

The Groupware and Messaging platform Lotus Notes and Domino faces pressure from new Software-as-a-Service delivery methods. In order to embrace the disruptive business model and the disruptive technologies that come with it, IBM Lotus has started a number of initiatives to bring the skills into their toolset. Though the launch of LotusLive to help bring SaaS offerings to market is consistent with principles extolled by Christensen, there are areas that Lotus could improve their efforts relative to a framework that seeks to embrace open innovation, lead users and their communities. This includes strengthening the browser based technologies relative to the gaps they have versus rich client technologies, and leveraging lead user communities who are not necessarily customers and have strong needs related to these gaps and the core disruptive technology characteristics.

THIS PAGE INTENTIONALLY LEFT BLANK

5- Conclusion

Software as a Service and Cloud Computing are a shift in software delivery model driven by improvements in network availability and economies of scale in processing power delivered by virtualization and other backend infrastructure technologies. Despite attempts in the late 1990's and early 2000's by Application Service Providers to attempt to deliver Enterprise Software in a multitenant, Enterprise Software Vendors have not had customer pressure to shift their delivery to browser delivered and self-hosted until relatively recently.

At the same time, consumer vendors like Google, Amazon and Ebay have been delivering massively scalable services over the internet for well over a decade. This gives them an advantage, using learned skills in datacenter optimization using technologies like virtualization. Enterprise Software Vendors from Microsoft to IBM and others make their money on the sale of software licenses and support contracts, and have realized that this internet-delivered, subscription based delivery model represents a threat to their existing revenue models. As such, they have launched a variety of initiatives to adopt the technologies, often cognizant of the challenges as described by Christensen.

Adopting and embracing a disruptive technology and addressing the new architectures and business models requires careful strategy and execution. Christensen describes a method called 'Value Network Analysis' that maps the architecture and firms involved in the delivery of a product or service into a hierarchy and defined ordered sets of performance characteristics that are valued by customers. Identifying adjacent value networks with similar performance characteristics and investigating their ability to encroach on a firms own value network is a sound method to identify potential disruptive technologies.

As prescribed by Christensen, firms are most successful when they keep the following items in mind:

- *Match the disruptive technology to an organization with customers that need it*
- *Match the projects to appropriate size organizations*
- *Plan to fail early and inexpensively*
- *Utilize existing resources but not processes and values*
- *Found or developed new markets*

The identification and adoption of the performance characteristics and the methods to embrace disruptive technologies described by Christensen can be complemented by a strategic use of both open innovation and platforms.

Open innovation is an umbrella of research that focuses on categories of users who openly and often freely reveal their innovations, their motivations and the method in which they interact. Lead users as researched by von Hippel have interesting characteristics – they care deeply about their specific needs, and often have solutions to problems that they will freely reveal. They are also experiencing needs that are forerunners to broader market arrival of the same need. Pairing the performance characteristics that a firm requires to embrace disruption with lead users with the need for the same characteristics can enable strong collaboration and iterative investigation of the problem area and the sharing of solutions. Creating innovation communities where numbers of lead users can interact will help them to better express their strong needs, and create more and better solutions.

The use of platforms is an architectural method that uses modularity and layering to allow multiple products and components to be built to work together. Platforms offer cost structure advantages and offer strategic potential to deliver multiple products to different markets. Platform strategy as described by Cusumano requires rigor – specifically the selection of coring or tipping as a method to grow the platform, and using Cusumano’s “Four Levers” – scope of the firm, product technology, external relationships and internal organization.

A strategic framework to use open innovation and platforms to embrace Software as a Service creates a structure for a firm to embrace disruptive technology. The framework describes identifying disruption, assessing performance characteristics, and taking the steps to create the architecture, process and organization to follow Christensen’s observations of what has made other firms successful. Using open innovation and platforms, firms can maximize their use of the commoditizing technologies in lower layers of SaaS delivery infrastructure and ensure that they are iterating and finding strong needs and their solutions.

IBM Lotus faces a disruptive threat from Software as a Service, and has undertaken the work required to build skills in delivering software in this new method. The LotusLive product, an online product which initially targeted online web conferencing as a market, reacts properly to the research as described by Christensen. The organization is self contained, and attempts to

shed the existing values of on-premise software delivery. The target market, web conferencing, has a strong need for the potentially disruptive characteristics of massive scalability and fast time to value. Despite these initiatives being in flight, IBM Lotus could benefit from finding lead users who are not necessarily current customers, or even in the same market. With those lead users, IBM Lotus should target development not only of performance characteristics that are potentially disruptive, but also the performance characteristics that are current gaps against Lotus' legacy technology. Some examples of these characteristics include disconnected use, security, desktop integration and streaming multimedia support. Lotus should consider taking on a structured approach to filling these gaps, as described by the methods presented in this work.

Bibliography

Abernathy, W., & Clark, K. (1985). Innovation: Mapping the Winds of Creative Destruction. *Research Policy* (14), pp. 3-22.

Allen, T., & Henn, G. (2006). *The Organization and Architecture of Innovation: Managing the Flow of Technology*. Burlington, MA: Butterworth-Heinemann.

Christiensen, C. (1997). *The Innovator's Dilemma*. New York, NY: Harvard Business School Press.

Churchill, J., Hippel, E. v., & Sonnack, M. (n.d.). *Lead User Project Notebook*. Retrieved 12 18, 2009, from Eric von Hippel's Homepage:
<http://web.mit.edu/evhippel/www/Lead%20User%20Project%20Handbook%20%28Full%20Version%29.pdf>

Cusumano, M., & Gawer, A. (2002, Spring). The Elements of Platform Leadership. *MIT Sloan Management Review* , pp. 51-58.

Desisto, R., & Pring, B. (2008, May 30). Essential SaaS Overview and Guide to SaaS Research. Gartner.

Desisto, R., Plummer, D., & Smith, D. M. (2008, April 7). Tutorial for Understanding the Relationship Between Cloud Computing and SaaS. Gartner.

Duhaime, G. (2007, December). CRM Software as a Service Update 2008. Aberdeen Group.

Eclipse.org. (n.d.). *Open Healthcare Framework (OHF) Project*. Retrieved January 12, 2010, from Eclipse.org.

Erdogmus, H. (2009, March). Cloud Computing: Does Nirvana hide behind the Nebula? *IEEE Software* , pp. 4-6.

Erdogmus, H. (2007, July). On-Demand Enterprise Services: Where's the Catch? *IEEE Software* , pp. 5-7.

Evans, D., Hagiou, A., & Schmalensee, R. (2006). *Invisible Engines: How Software Platforms Drive Innovation and Transform Industries*. Cambridge, MA: MIT Press.

EWeek. (2009, February 02). *IBM Unveils Cloud Computing Division, Strategy and Partnership* . Retrieved January 12, 2010, from eWeek: <http://www.eweek.com/c/a/Cloud-Computing/IBM-Unveils-Cloud-Computing-Division-Strategy-and-Partnership/>

Henderson, R., & Clark, K. (1990, March). Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Existing Firms. *Administrative Science Quarterly* , 35 (1), pp. 9-30.

Hill, C., Yates, R., Jones, C., & Kogan, S. (2006). Beyond Predictable Workflows: Enhancing Productivity in Artful Business Processes. *IBM Systems Journal* , 45 (4).

IBM. (n.d.). *Global Businesses Choosing Lotus Software; More Than Half of Fortune Global 100 Now Using Lotus Notes / Domino*. Retrieved October 20, 2009, from MarketWire: <http://www.marketwire.com/press-release/Ibm-NYSE-IBM-938111.html>

Lakhani, K., & Wolf, R. (2005). Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. In J. Feller, B. Fitzgerald, S. Hissam, & K. Lakhani (Eds.), *Perspectives on Free and Open Source Software*. Cambridge, MA: MIT Press.

Mertz, S. (2009, May 5). Market Trends: Software as a Service, Worldwide, 2008-2013. Gartner.

Meyer, M., & Seliger, R. (1998, Fall). Product Platforms in Software Development. *Sloan Management Review* , pp. 61-74.

Nambisan, S., & Sawhney, M. (2007). *The Global Brain: Your Roadmap for Innovating Faster and Smarter in a Networked World*. Upper Saddle River, NJ: Wharton School Publishing.

Natis, Y., Gall, N., Cearley, D., Leong, L., Desisto, R., Lheureux, B., et al. (2008, July 8). Cloud, SaaS, Hosting and other Off-Premises Computing Models. Gartner.

Openntf.org. (n.d.). *Openntf.org - Open Source Community for Lotus Notes Domino*. Retrieved January 12, 2010, from Openntf.org: <http://www.openntf.org/>

O'Sullivan, D. (2009, Spring). The Internet Cloud with a Silver Lining. *The British Journal of Administrative Management* , pp. 20-21.

Pring, B., Desisto, R., & Bona, A. (September 28, 2007). *The Cost and Benefits of SaaS vs. On-Premise Deployment*. Gartner.

Raymond, E. S. (2001). *The Cathedral and the Bazaar: Musings on Linux and Open Source by and Accidental Revolutionary*. Sebastopol, CA: O'Reilly and Associates.

Rivest, R., Shamir, A., & Adleman, L. (1983). *Patent No. 4,405,829*. Cambridge, MA.

Service-now.com. (2009). *A Brief History of SaaS*.

Truitt, M. (2009, September). Editorial: Computing in the "Cloud". *Informational Technology and Libraries* , pp. 107-108.

von Hippel, E. (2005). *Democratizing Innovation*. Cambridge, MA: MIT Press.

Wikipedia. (n.d.). *Document-oriented Database*. Retrieved October 28, 2009, from Wikipedia: http://en.wikipedia.org/wiki/Document-oriented_database

Wikipedia. (n.d.). *Relational Database*. Retrieved October 28, 2009, from Wikipedia: http://en.wikipedia.org/wiki/Relational_database

Wladawsky-Berger, I. (2007, Spring). ESD 57: Technology-based Business Transformation. Massachusetts Institute of Technology.