Software Defined Infrastructures – Implications for Technology and Business Strategies for competing in the era of Hyper-Scale Computing

By

Rahul Pradhan

M.S., Computer Science
Worcester Polytechnic Institute, 2001

B.E., Computer Engineering
University of Mumbai, 1999

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

At the

Massachusetts Institute of Technology
June 2015

Signature of Author

Signature redacted

Rahul Pradhan
System Design and Management Program

Certified and Accepted by

Signature redacted

Patrick Hale
Director
System Design & Management Program
Senior Lecturer, Engineering Systems Division

Contents

Tables and Figures

This page is intentionally left blank

Acknowledgements

This is the culmination of an incredible journey both personally as well as professionally at the System Design and Management program at MIT. As I look back at the program I am grateful to all those who enriched my experience at MIT - from classmates to professors. It's been a great experience and a lot of fun amidst all the hard work to get to this point. I would like to thank the staff at SDM especially Bill Foley for his help navigating the various administrative details of the program. I greatly appreciate the support and guidance provided by my thesis advisor - Pat Hale not only for this thesis but throughout my stay here at MIT. Pat has done a phenomenal job leading the SDM program and making it such a close knit family. Thank you Pat – it was a pleasure working with you.

I would also like to acknowledge my family and friends for their constant encouragement and for providing the inspiration to follow my dreams. I want to thank my friend Sooraj - for his support both at school and at work.

To my parents – thank you for the love, support and encouragement through all these years. Last but not the least – thanks to my wonderful wife Shivani - for her love, support, motivation and understanding without which this thesis would not have been possible.

And finally, to my two little girls - Sara and Anishka - thank you for being the stress busters in my life - this like everything else is for you!

To,

Sara and Anishka

Software Defined Infrastructures – Implications for Technology and Business Strategies for competing in the era of Hyper-Scale Computing

By

Rahul Pradhan

Submitted to the System Design and Management Program
in June 2015 in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering and Management

Abstract

The adoption of Cloud Computing and the emergence of new Cloud services pioneered by Amazon (AWS) have brought the importance of agility and flexibility of infrastructure to the forefront. Companies ranging from small to large Enterprises today have a Cloud Strategy. Their Cloud Strategy ranges from being all-in with respect to moving their internal IT infrastructure to the Cloud to moving only specific low SLA workloads to the cloud. However, not everyone can or is comfortable letting their sensitive data leave their infrastructure and reside on third party infrastructure that is not in their control. This has led to customers building Private Clouds, which however don't give them the scale or the flexibility that Public Clouds provide. So customers are now looking at ways to replicate the success of Public Clouds in their own environments. What they find is that the existing IT infrastructure and its architecture is inadequate to provide those benefits.

To achieve the Public Cloud characteristics, customers have started looking at the infrastructure built by Hyperscalers like Google, Amazon, Microsoft, Facebook and the benefits that they have been able to achieve as a model to build their own IT data centers. The infrastructure of these Web giants consists of commodity hardware components managed and driven by intelligent Software. This has given rise to various Software Defined technologies like Software Defined Networking and Software Defined Storage. As the customer interest and adoption of these technologies increase it presents a huge business challenge to existing IT equipment vendors. Not only are they faced with technological challenges as the architecture moves in a different direction than they were charting to but it also presents a business model change which if not navigated carefully can lead to significant erosion of their revenues.

This thesis identifies the impact of Software Defined Infrastructures on the enterprise equipment vendors and proposes strategies for successfully competing in the age of Hyper-scale Computing.

Thesis Supervisor: Patrick Hale

Title: Director, System Design and Management Fellows Program.
Senior Lecturer, Engineering Systems Division

This page is intentionally left blank

1. Introduction

Cloud Computing has seen a significant ramp up in its adoption rate over the last five years. The adoption of Cloud Computing and the emergence of new Cloud services pioneered by Amazon Web Services (AWS) have brought the importance of agility and flexibility of infrastructure to the forefront. Cloud Computing has been instrumental in opening up new markets and opportunities and enabling new business models. Public Clouds can effectively give users access to unlimited computing and storage resources delivered in a pay-as-you go model. This low acquisition cost of IT resources is fueling innovation as companies can now get the necessary resources without the significant investments upfront thus freeing them to invest in areas that have a direct business impact. The elastic nature of the Cloud also reduces the risk as companies only pay for what they use and can scale down or reconfigure resources based on their demands and needs.

This is particularly true for newer businesses starting out their operations. In order to prototype their ideas or build products that need scale startups can now directly leverage the massive computing power of the Cloud at a fraction of the cost of what it would have taken to build it themselves. This unprecedented access to computing is further democratizing innovation. The Internet leveled the playing fields by providing ubiquitous access to all and that resulted in the dot-com boom of the late 1990s and 2000s. According to leading analyst firms like Gartner – the Public Cloud services grew by 17.5% in 2014 while three-fourths of large Enterprises are on track to have a Hybrid Cloud by 2015.

Cloud Computing builds on the ubiquitous access to the Internet by expanding the services available over the Internet to scale-out computing and storage resources. Several of the hottest startups today like – Drop Box (Cloud Storage), AirBnB, Pinterest, Yelp all leverage Amazon Web Services' (AWS) public Cloud in order to run their businesses. Doing so gives them 3 crucial benefits that a startup would typically struggle with

- Low Acquisition Cost -

The Public Cloud enables companies to experiment without having to buy expensive infrastructure by offering a pay-as-use business model

- Elastic Resources –

Elasticity ensures that resources can grow as per the need to the business in an elastic, non-disruptive manner. Similarly resources can be shrunk in order to account for downtime or lack of demand - all this while the customer pays only for resources utilized. This results in significant savings for startups as well as ensuring that there is no revenue lost due to the sudden growth spurts or spikes in customer demand.

- Access Anywhere –

The global reach of the Internet enables businesses to make their applications and or Content available to a wider audience through a variety of access methods (like desktop, mobile etc.)

As a result of these benefits, the Cloud has become the preferred infrastructure provider for most startups.

The benefits of the Cloud, however, are not just restricted to startups or small to medium sized companies. They apply to large established players too. Businesses like Netflix, which offers on-demand streaming of movies, TV shows and other video content host their website, as well as applications and databases on AWS. In Netflix's case the actual video content is served via Content Delivery Networks but the applications for customers to log in to their accounts; browse content is served out of Amazon's Cloud. As the adoption of the Cloud grows and becomes mainstream, large Enterprises are starting to look at the Cloud as an opportunity to again agility and flexibility for their businesses.

In addition to these business benefits, there are macro level trends that are impacting Enterprise IT today.

- Consumerization - Consumertization of IT refers to the use of consumer products and services in an IT environment in which the products weren't designed to be used nor was the IT environment designed to accommodate such consumer devices. This trend encompasses not only consumer electronics devices like Smartphones and tablets but consumer grade services like email (Gmail), Cloud storage (Dropbox) and social media and networking sites like Twitter and Facebook. This trend poses a significant risk for IT from a security perspective. However, most IT organizations have started to embrace this trend as they realize that it is important for their employee productivity and in turn for business agility to enable these technologies and devices. This also means that the IT infrastructure needs to as agile as the Cloud infrastructure is. As employees using these devices are exposed to consumer applications that are significantly easy to deploy and access from a variety of devices they also start expecting the same experience from Enterprise applications. IT realizes that it needs to modernize its applications and delivery mechanisms not only to support newer devices and deployments but also as a productivity measure for the broader organization.

- Generating Business Value – One of the drivers for IT to be more efficient is to generate greater value for the Business. However, the complexity of the IT infrastructure with a diverse set of applications and technologies mean that IT spends a lot of time maintaining and managing existing infrastructure rather than developing new applications that can help businesses grow their top line. According to Forrester Research, IT organizations spend 72% of their IT budget on maintaining operations, and only 28% on investing in new projects. IT needs to reallocate budget, time, and resources from maintaining operations, and shift them to drive new business value and innovation. This cycle needs to change since Public Cloud infrastructure has shown that there are ways to automate and orchestrate infrastructure management and deliver significant value to the business.

- Shadow IT - The second trend that is impacting how Enterprises are thinking about this is that of Shadow IT. Shadow IT refers to the activities of the business that happen outside the control of IT (1). Shadow IT is a result of the inability of IT to provide developers with computing resources in a timely manner. Requesting computing resources in a traditional IT environment means long equipment acquisition cycles that require several levels of approvals. Cloud changed the mindset of developers. In the Public Cloud environment, if a developer needs computing resource then he can request it by the swipe of a credit card and get access to virtually unlimited compute and storage immediately. This agility in turn spurs innovation as developers can quickly spin up prototypes of their ideas in order to deliver value to the business. Developers are used to this level of service in a non-IT environment and expect the same of their IT. Due to a lack of such a service they end of using resources from the Public Cloud thus bypassing the control and oversight of IT.

In response to these trends and co-evolving technologies, companies ranging from small to large Enterprises today have a Cloud Strategy. Their Cloud Strategy ranges from being all-in with respect to moving their internal IT infrastructure to the Cloud to moving only specific low service level requirement (SLA – service level agreement) workloads to the cloud. However, not everyone can or is comfortable

However, a full adoption of the Cloud and a transition of on-premise IT to the Cloud is not a feasible option for most of the companies. This is because, despite all of the benefits of the Cloud, it also throws up challenges of security, privacy and compliance, which have held back several companies and industries from completing the transition from an on premise private infrastructure to an off-premise public infrastructure. Letting their sensitive data leave their infrastructure and reside on third party infrastructure that is not in their control is something a lot of the companies are not comfortable with.

This has led to customers building Private Clouds on their premises to try to generate similar benefits as that of a Public Cloud. However, Private Clouds don't give them the scale or the flexibility that Public Clouds provide. So customers are now looking for ways to replicate the success of Public Clouds in their own environments while leveraging the Public Cloud infrastructure as needed for use cases that make sense to their business. But these companies are finding that in order to achieve this Hybrid Cloud model the existing IT infrastructure and its architecture is inadequate to provide those benefits.

Several of these large customers are starting to look at the infrastructure built by Hyper-scalers (Hyperscale Providers) like Google, Amazon, Microsoft, Facebook and the benefits that they have been able to achieve as a model to build their own IT data centers. The infrastructure of these Web giants consists of commodity hardware components managed and driven by intelligent Software. This has given rise to various Software Defined technologies like Software Defined Networking and Software Defined Storage. The ability to spin-up infrastructure on demand and the flexibility to provision at varying degrees of service level agreements (SLAs) is becoming extremely important. At the same time, the capability to reconfigure the infrastructure as needed and consolidate existing physical assets to optimize their use enables significant cost savings. This has led to the creation of next-generation architectures in the data center that are software defined. The Software Defined Data Center enables all of these in addition to the ability to deploy on lower cost hardware like general-purpose server platforms instead of custom-built specialized hardware.

The commoditization of hardware platforms — and the relatively slower time to market for and higher development costs associated with infrastructure products based on custom hardware components — has compelled suppliers to shift the focus of their innovation to delivering value via software-defined solutions. This trend initially started in the computing domain and is now sweeping the storage and networking domain.

In a widely cited essay published in the Wall Street Journal in 2011, Marc Andreessen - co-founder and general partner of the venture capital firm Andreessen-Horowitz and co-founder of Netscape famously proclaimed that

"Six decades into the computer revolution, four decades since the invention of the microprocessor, and two decades into the rise of the modern Internet, all of the technology required to transform industries through software finally works and can be widely delivered at global scale" (2)

Over the last decade, a Software revolution has been underway that has transformed several industries. The ability via Cloud Computing to deliver Software on a global scale has enabled accessibility and redefined several businesses. Retail giants like Wal-Mart and FedEx have used Software to revolutionize their supply chain and logistics.

Sectors of the economy not previously thought of having a major Software component like the auto industry are on the verge of being redefined via companies like Tesla. Tesla is a new kind of a car company that has by the power of software transformed the complete lifecycle of auto ownership right from buying to driving to repairing. Software companies like Google are leading the charge with designing and prototyping the cars of the future that can drive autonomously.

And it's not only capital goods like cars and engines that are going to be redefined. It's the more mundane things as well – like the thermostat. A company called Nest changed the traditional thermostat to make it a software defined intelligent home automation engine. Nest best captures that in their mission statement "Reinvent Unloved But Important Devices In The Home" (3).

Similarly, companies like Uber and AirBnB are revolutionizing entire business sectors by the touch of their apps on any smartphone.

The infrastructure used to build these innovative products and businesses is also in the process of being transformed by this trend in order to support these innovative business models and delivery platforms. The era of the software-defined datacenter will have a profound impact on how computing platforms get classified — and more importantly shapes the competitive landscape, creating a new breed of traditional (hardware-based) and next-generation (software defined) platforms giving rise to the concept of Software Defined Infrastructure.

In the mobile-cloud era, businesses need to innovate and respond quickly to maintain competitive advantage, and now have many different options to source IT services. IT needs to be increasingly agile and responsive as a strategic partner to the business units. The business units have increasingly relied on external service providers to deliver the services they need quickly and in an agile manner. However, it falls to IT to maintain control to ensure uptime, performance, security and compliance. As a result, IT is faced with a defining challenge of this new era: delivering on the new business need for speed and agility and the ongoing need for control and efficiency.

As we look at this competitive landscape, the players in this space can be broadly classified as belonging to two distinct categories.

- Incumbent Equipment Vendors
- Cloud Service Providers

The Incumbent Equipment Vendors (IEVs) are the traditional IT equipment providers like Cisco Systems, EMC, IBM, HP who supply IT infrastructure to the world's largest companies. The Cloud service Providers (CSPs) include newer firms like Amazon web Services (AWS), Microsoft Azure, Google Compute Platform, Rackspace who provide Public Cloud services as the IT infrastructure for other companies. In most cases, these two different segments end up clashing head on for the same set of customer dollars.

The table below captures the technology and business challenges faced by these two sets of players. Incumbent equipment vendors face technology challenges due to the emergence of next generation architectures that are Software Defined along with business headwinds on two front – one due to their customers moving to the Public Cloud and secondly due to commoditization of the hardware that these firms depend on to monetize.

	Incumbent IT Vendors	Cloud Providers
Business Challenges	Movement to the Public Cloud Hardware Commoditization	Threat from new entrants Bridging on-prem to the Cloud
Technology Challenges	Software Defined Infrastructure	Security, Privacy and Compliance

Figure 1: Technology and Business Challenges

On the other hand Cloud Providers face challenges due on the Security, Privacy and Compliance front that are significant barriers to entry for most Enterprises to move most of their workloads to the Cloud. The total cost of ownership of moving to the Public Cloud for a mid to large Enterprise isn't clear either. The shifting of a capital expense model to an operating expense model while appealing initially may turn out to be more expensive in the longer term. It is clear that there is no one winner in terms of the Cloud deployment models. There will be some companies for whom the Public Cloud model would make complete sense but for a majority of Enterprises it will be a mix between having applications and workloads on premise and in the Cloud depending on the use cases and business requirements.

The Traditional vendors want to make delivering Cloud like capability and experience in a Private Cloud environment while at the same time enabling their customers to burst to the Cloud if needed. The Cloud providers on the other hand are looking at packaging their public Cloud infrastructure as product that can be spun up in a private environment so that Enterprises can leverage the same benefits as those provided by a Public Cloud but on infrastructure under their control.

This sets up a very intense battle between the traditional vendors who are racing to build infrastructure with Cloud-like capabilities and the Cloud providers who are trying to bundle their infrastructure and deliver it as a turnkey solution. Both sides have significant challenges. The Enterprises are not used to building Cloud like architectures and the Cloud Providers don't currently have the infrastructure that can run traditional Enterprise applications. As the customer interest and adoption of these technologies increase it presents a huge business challenge to existing IT equipment vendors. Not only are they faced with technological challenges as the architecture moves in a different direction than they were charting to but it also presents a business model change which if not navigated carefully can lead to significant erosion of their revenues.

This thesis aims to study and understand the impact of the emerging technologies of Software Defined Infrastructure on the technology and business strategies of firms competing in this next phase of technology evolution.

1.1 Thesis Organization

The thesis is organized into chapters as follows –

Chapter 1 provides an Introduction and sets the context for the research by briefly explaining the current situation and framing a problem statement and the goals of the thesis.

Chapter 2 discusses the Technology Evolution and the evolution of the Data center architecture. It also describes Hyperscale computing and discusses the infrastructure requirements in these environments. The chapter uses examples of Google and Facebook and their innovative solutions to their Data Storage challenges to demonstrate the differences between the Enterprise deployments and Hyperscale deployments.

Chapter 3 focuses on the technology architecture for hardware defined and software defined infrastructures. It uses System Architecture analysis to discuss Software Defined Infrastructures and outlines the main design principles behind it.

Chapter 4 focuses on the technology adoption and business analysis by look at the disruptive technology of SDI, the diffusion of innovation while looking at the Business ecosystem and the value chain dynamics of a SDI.

Chapter 5 utilizes the frameworks and models discussed in the thesis to offer strategic recommendations for firms to compete in this space.

The thesis will use SDI as a short form for Software Defined Infrastructure and IEVs for Incumbent Equipment Vendors like Cisco, EMC, HP, IBM, and Oracle.

1.2 Research Methodology

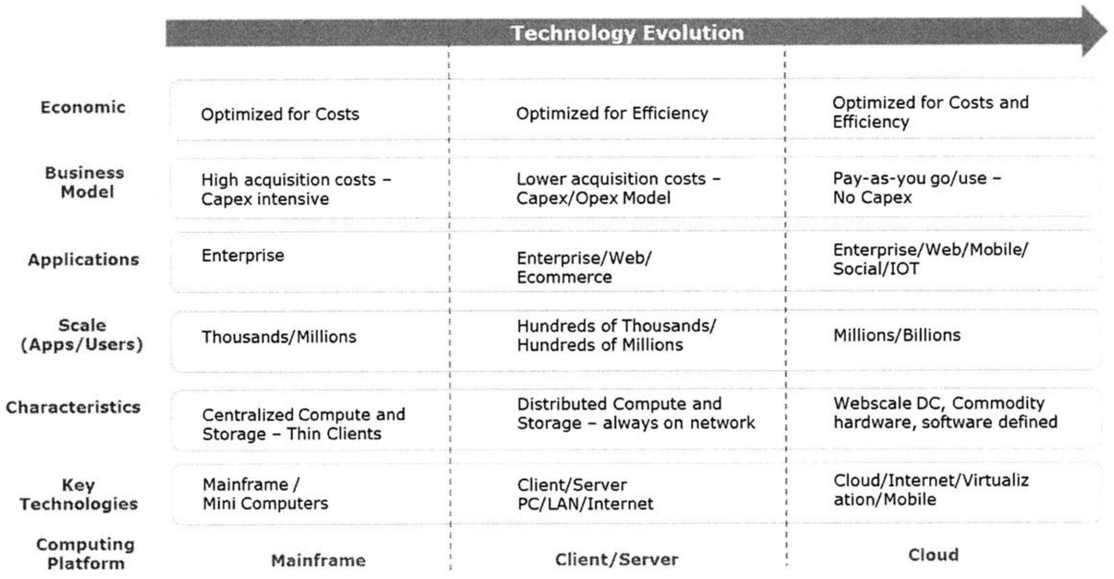
The research approach centers around literature review, case studies, technical analysis and business analysis to understand and evaluate the impact of the emerging trends on players in this space.

2. Technology Evolution

Over the last 35 years, the technology landscape has undergone some significant changes and shifts. We classify the evolution of computing technology broadly into 3 major episodes

- Mainframe Computing,
- Client-Server Computing and
- Cloud Computing.

The figure 2 below provides an evolution map across the three computing generations along several different dimensions.



Technology Evolution			
Economic	Optimized for Costs	Optimized for Efficiency	Optimized for Costs and Efficiency
Business Model	High acquisition costs – Capex intensive	Lower acquisition costs – Capex/Opex Model	Pay-as-you go/use – No Capex
Applications	Enterprise	Enterprise/Web/ Ecommerce	Enterprise/Web/Mobile/ Social/IOT
Scale (Apps/Users)	Thousands/Millions	Hundreds of Thousands/ Hundreds of Millions	Millions/Billions
Characteristics	Centralized Compute and Storage – Thin Clients	Distributed Compute and Storage – always on network	Webscale DC, Commodity hardware, software defined
Key Technologies	Mainframe / Mini Computers	Client/Server PC/LAN/Internet	Cloud/Internet/Virtualiz ation/Mobile
Computing Platform	Mainframe	Client/Server	Cloud

Figure 2: Technology Evolution

The first generation of computing gave rise to extremely powerful mainframe computers based on a centralized computing methodology. These were very large and expensive computers capable of supporting hundreds, or even thousands, of users simultaneously. They were characterized by -

centralized compute resources and thin-clients serving Enterprise applications. The high price of acquisition and maintenance of these systems has led to optimizations along the cost dimension.

The second generation of computing used a decentralized model to address some of the inefficiencies of the mainframes. Technologies like the Client-Server architecture and networking led to the rise of distributed compute and storage that hosted thousands of applications serving millions of users all optimized around efficiency.

The third generation of computing platforms referred to as the Cloud - is the result of evolution and adoption of existing technologies and paradigms. The Cloud uses massively scalable data centers in order to serve millions of application to billions of users. It tries to achieve efficiencies due to technologies like virtualization while optimizing for costs via scale and commodity hardware. The goal of cloud computing is to allow users to benefit from all of these technologies, without the need for deep knowledge about or expertise with each one of them. The cloud aims to reduce costs and help the users concentrate on their core business instead of being impeded by IT obstacles.

Over the past decade, a commoditization and standardization of technologies, virtualization and the rise of service-oriented software architectures along with the dramatic growth in the adoption of the Internet has led to an evolution from legacy, hardware-oriented datacenters, through adoption of virtualization to private cloud infrastructures.

This is a significant structural shift towards the next generation of computing platform that incorporates other co-evolving technologies like Mobile, Social and Big Data.

2.1 Application Evolution

The table below highlights the differences between Cloud Applications and traditional Enterprise applications. The traditional Enterprise applications are built assuming a scale-up highly reliable infrastructure underneath. They are designed for performance and have low elasticity.

Characteristics	**Enterprise Applications**	**Cloud Applications**
Development Paradigm	Design for Reliability/Performance	Design for Fail
Resiliency	Infrastructure Based	Application Based
Elasticity	Low	High
Scale	Scale-up – Vertical	Scale-out - Horizontal
Security	Stringent security requirements	Lower Security requirements
Compliance & Governance	High	Low
Application Life time	Long lived, Static	Short lived, Dynamic
Complexity	Embedded in the Infrastructure	Embedded in the Application
Examples	ERP, HRMS, SAP, Oracle	Social, Web 2.0, Big Data

Figure 3: Application Characteristics

On the other hand, Cloud applications are designed for fail, provide application-based resiliency and have high elasticity and are built on infrastructure that is designed to scale out. Enterprise applications have stringent security and compliance requirements. Since these applications handle sensitive information they need to be secure as well handle privacy, compliance and governance requirements. Cloud applications on the other hand have lower security requirements. The thought process for most enterprises is if it is a security or compliance concern; do not put it in the Cloud. Overall the complexity for Enterprise applications is embedded in the infrastructure as it takes care of providing the reliability and resiliency while in the case of Cloud applications each application is architected to handle those characteristics thus making it complex. However, by embedding those characteristics in the application it makes it agnostic to the underlying architecture thus increasing the agility and flexibility of deploying these applications. The Hyperscale providers primarily use and develop Cloud applications.

2.2 Data Center Architectures

Today's Enterprise Data centers are extremely complex systems built on top of best of breed products. The IT organizations have been responsible for integrating these products build by different vendors and delivering an infrastructure that can meet the requirements of the Business. The traditional Enterprise Data Center is made up of several discrete components.

- *Physical Data Center* – This involves the physical infrastructure like a building or facility to house the infrastructure along with the power, cooling and requirements.
- *Network* – The networking component includes the hardware and the software required to provide connectivity to the Data Center. The network comprises for a Local-area network (LAN) to provide connectivity between compute instances as well as networked storage devices. It also includes Security appliances and other products (like load balancers) required to effectively operate and manage the connectivity. In addition to the LAN there is also the Wide-area network (WAN) connectivity required to connect to the outside world and to other locations belonging to the business.
- *Compute* – The Compute component in traditional Data Centers is usually a non-virtualized physical Server that runs applications and is connected via the LAN to a shared storage used to host the application data.

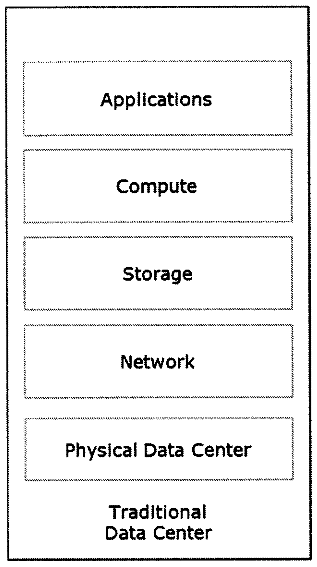


Figure 4: Traditional Data Center

- *Storage* – The Storage component for a traditional data center is either file or a block storage system that offers shared storage for the applications running on the Compute nodes or as storage for user generated content.
- *Applications* – Finally applications are the programs that use the infrastructure in the Data Center to provide services and generate business value.

Traditional Enterprise computing is built on the principles of Reliability, Resiliency and Serviceability. This means that the underlying infrastructure must provide reliable and resilient components that are highly available and designed in a redundant manner. There is an expectation of at least N + 1 component so that in the event of a failure there would be another stand by component to take over the operation without any degradation of service. From a maintenance and service perspective it means that the components that suffer a down time need to be serviced and brought back up to get the system back to its healthy state.

With the advent of Virtualization the Enterprise Data Center Technology stack added an additional layer of virtualization that sat between the Compute and Application layers. Virtualization enabled abstracting of the Server hardware resources so multiple applications running different operating environments could share the same infrastructure. Virtualization has significantly improved server utilization and has hidden many of the complexities of dealing with physical systems from the operating systems and applications reducing both capital and operational expenses. As a result of Virtualization and the disaggregation of Software from the hardware it enabled a customer to rip out server hardware belonging to company A in favor of hardware with identical specifications from company B. This resulted in accelerating the commoditization curve for servers as all the differentiation moved to the software than the server hardware itself.

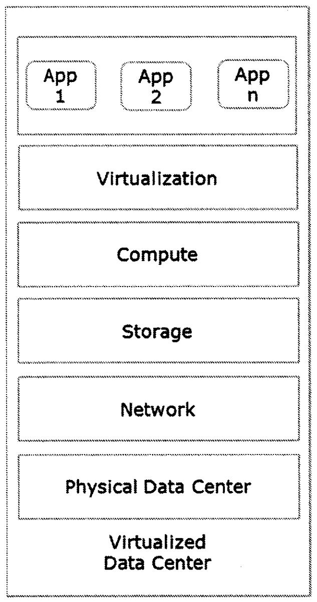


Figure 5: Virtualized Data Center

Converged infrastructure (CI) is an integrated set of compute, storage, and networking components with infrastructure management software that provides a single logical chunk of hardware and software that is either specifically engineered together or at the very least tested and proven in a variety of configurations and applications.

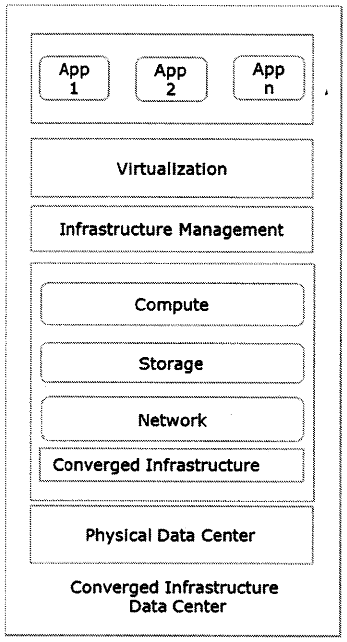


Figure 6: Converged Infrastructure Data Center

Converged infrastructure simplifies hardware and software management and accelerates the deployment of infrastructure for private clouds.

2.3 HyperScale Computing

Hyperscale is defined as –

"The ability of an architecture to scale appropriately as increased demand is added to the system. This typically involves the ability to seamlessly provision and add compute, memory, networking, and storage resources to a given node or set of nodes that make up a larger computing, distributed computing, or grid computing environment. It is the foundation necessary in order to build a robust and scalable cloud, big data, map reduce, or distributed storage system and is often associated with the infrastructure required to run large distributed sites such as Facebook, Google, Microsoft or Amazon." (4).

Companies like Amazon with their Amazon Web services offering deal with a scale that is order of magnitude more than any traditional Enterprise. AWS alone in 2014 had 1 million active customers, they added as much new server capacity as needed to support a company the size that amazon was in 2004 -$7B every day (5). That is an incredible amount on computing capacity that comes online every day. This is AWS alone and there are similar statistics for Facebook, Google and Microsoft for both their public customer offerings as well as internal infrastructure needed to support those.

Given this scale, Hyperscale architectures require a fundamentally different mindset than the one taken with traditional enterprise IT systems. The services that Hyperscale providers offer require significant amount of computing power and resources. A single search query may require hundreds of megabytes of data and billions of processing cycles in order to fulfill it. Building giant machines to process these can quickly become cost prohibitive. Hyperscale involves abstracting the intelligence that resides in the hardware in traditional enterprise IT systems in software that can be then decoupled from the underlying hardware and deployed independently. Even traditional hardware characteristics like resiliency, reliability and performance are implemented in software. This means that the Software needs to do much more than previously in order to maintain characteristics like high availability and reliability. These typical hardware characteristics may now be measured via a different yardstick based on the applications using them.

Hyperscalers built their applications that did not assume the same level of reliability and resiliency from the underlying infrastructure as enterprise application providers did. This enables them to optimize the infrastructure while having applications take on the responsibility to managing the consistency and reliability. Since the intelligence in the infrastructures is itself abstracted in the software, applications can now tap in to the infrastructure software to make decisions on the reliability of their operation.

This move by the web giants to build a new agile infrastructure is driven by their business needs. Companies like Google, Amazon, Microsoft, and Facebook operate in an environment where agility and speed are competitive advantages. Their entire business rests on the ability to execute faster. Add to it the cost factor which means they can't afford to spend at the same level as Enterprise IT because of the massive scale that they deal with and it quickly leads to the realization that traditional IT isn't a sustainable model for hyper-scale environments. By cutting costs through the use of commodity hardware and investing the savings back in their infrastructure by engineering software to provide them the required speed and flexibility, these companies are enabled to be as successful as they have been over the past decade, and more. The other important element apart from commodity components from a costs perspective is the energy consumption of their data centers, which is generally one of the biggest expenses associated with running one. By optimizing hardware, hyper-scalers have been able to make their operations green and significantly reduced the power usage. They have achieved this by a variety of techniques such as – hot air containment of the exhaust coming out of computing products, ambient air cooling as well as building data centers in areas of lower energy costs or cooler climates where they can take advantage of natural cooling instead of conditioning air in order to cool the equipment. From an equipment perspective, hyper-scalers are the ones who have been driving towards standardization of components and skinless infrastructure elements like servers to reduce energy consumption further.

Infrastructure hardware is optimized for efficiency by removing components that are either not needed because of the specific usage of these devices or applications where hardware functions have been subsumed by equivalent functionality provided by the Software layers above. This makes the implementation at the hardware layers pretty straightforward using commodity components while pushing the complexity to the layers above.

As an example – Enterprise IT organizations are very concerned about high availability and reliability. High availability refers to the on-demand availability of a system when one or more components of a system go down. Example of these components include - power supplies, network cards, disk drives, processors, memory etc. In traditional IT infrastructure redundancy is provided at the hardware level which means there is two of every component in the system so that if the primary component fails then the software running can fail over seamlessly to the secondary or stand-by component without the applications even being aware of the failure. Enterprise applications rely on this underlying resiliency of infrastructure.

As web scale companies starting looking at their infrastructure they realized that deploying Enterprise grade infrastructure at their scale isn't going to work. So they started building in the resiliency in software itself so that they would not require special purpose hardware to build their computing infrastructure on. By moving the resiliency from a hardware resiliency model to a software resiliency model the hyperscalers are able to leverage general purpose hardware components that are managed by specialized software in order to provide the same level of resiliency that hardware provides. Since the hyperscalers are no longer reliant on enterprise grade hardware they can go to cheaper ODMs to get commodity hardware or build their equipment themselves using commodity components. Facebook open-sourced its hardware designs under the Open Compute project in order to share their best practices and design for data center components.

Homogenizing the hardware makes it easier to manage and deploy systems using Software. By applying different Software personalities to the same piece of generic hardware, the hardware can be transformed into different elements of the infrastructure, replacing functions from the compute, network and storage components. The other benefit of homogenizing the infrastructure is to make it easier for application development. The application developer can treat the infrastructure as a black box and not worry about the network connections or the storage capacity required as it can be all managed behind the scenes by the Software layer of the infrastructure. Removing the dependency on specialized components and homogenizing hardware however makes the software and tools required to run and automate the infrastructure complex.

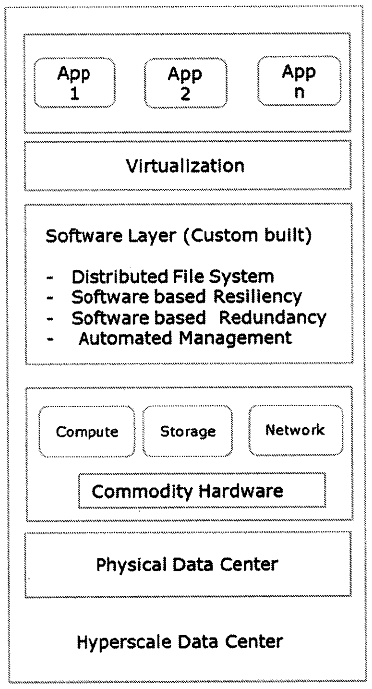


Figure 7: Hyperscale Data Center

Hyperscale Computing has successfully made the transition from a hardware-defined infrastructure to a more agile, flexible and lower cost Software Defined Infrastructure.

In the next section, we look at some examples of hyperscale architectures that been successfully implemented by the HSPs.

2.3.1 Google Architecture

Google is one of the largest technology companies in the world providing search, advertising, Cloud Computing and Software to its customers. Google is based on a radically different business model. It gives away several of its services for free. Those services like Gmail, Google Drive require huge amount of storage on the order of petabytes and exabytes. Google figured out that if they went the traditional equipment route they would not be able to offer those services and monetize them as they desired. The infrastructure that Google was built on would need to become their competitive differentiation, enabling Google to offer 'free' services while monetizing them through advertisements. Google figured that by hiring the smartest minds around it could build an infrastructure that eliminated waste and leveraged common components, thus driving the cost of hardware down while at the same time homogenizing the hardware so that the Software controlling it can abstract it.

The approach that Google took was to parallelize its infrastructure so that they can use thousands of commodity-class computers. By building an intelligent software layers that can manage and program these computers it enables creation of system that is comparable to high performance computing system albeit at a much lower costs.

Google's Computing Architecture is based on two fundamental characteristics.

- Reliability through Software
- Optimization for best price/performance.

These characteristics result in certain key design principles that Google's architecture follows.

The characteristics of Software reliability means that the infrastructure cannot rely on traditional reliability and fault tolerance features like multiple componentry available for failover (power supplies, memory, disk drives) or protocols like redundant array of inexpensive disks (RAID). The infrastructure software itself needs to focus on handling these faults and not rely on the underlying hardware. Using commodity general-purpose hardware also means that each component used in the infrastructure is stateless (instead of shared memory purpose-built appliances) and can leverage techniques like data replication for both fault tolerance and high availability as well as a means for greater aggregate throughput.

Optimization for price/performance dictates the choice of hardware being used since they need not use the hardware optimized for peak performance but can purchase a previous generation hardware that can provide the best price/performance metric. Typically for hardware due to Moore's law (expand) and following Intel's processor curve results in an EOL of the hardware in about three years which is a very short amortization schedule. This relative high weightage of hardware costs make traditional IT infrastructure less appealing and viable. Even though the special purpose equipment can provide higher peak performance the total cost of ownership for the hardware required for that level of aggregate performance is very high. Thus the actual equipment costs disproportionately impacts the overall price/performance ratio. By leveraging commodity hardware to reduce costs the upfront costs and investing in software efficiencies enables them to create significant value for their customers and positions them to capture the value too.

Commodity hardware comes with its own cost of managing and administering those devices. By homogenizing the hardware the hyperscale players can leverage orchestration and automation to significantly reduce operating costs while at the same time improving the operating efficiencies by making deployments agile and flexible.

The other benefit of using multiple commodity components is to reduce the failure domains and increase resiliency. Enterprise grade systems over fault tolerance, however, they are still limited by the number of fault-tolerant components. They usually provide a n-1 fault tolerant domain where n is the number of components supporting fault tolerance. Usually n = 2 and when one component fails the system goes in degraded mode and the failure of both the components can result in significant down time. The approach taken by the HSVs of deploying multiple smaller computing units can contain the failure domain to just one instance of that component. So for example if a disk on a server goes down or if a server itself goes down the data that is replicated to the other nodes in the system can still allow the rest of the system to function normally. Thus the cluster achieves the two main requirements - those of availability (software reliability) and the best aggregate price/performance.

By customizing their equipment and going to directly to the ODMs Google can eliminate unnecessary components like a graphics card for servers - most of the servers were never meant to power a display in order to accept commands but were remotely managed. Similarly enclosures for the motherboards and disks drives weren't needed as they slot directly into the racks.

Hyperscale architecture isn't only optimizing for hardware, it also includes changes to software to adapt to distributed architectures so for example software databases evolve to newer technologies like Hadoop which leverage distributed programming models like MapReduce.

With the massive growth in Cloud and online services Google had their largest spending quarter on the Hyperscale infrastructure (included plant, equipment and data center infrastructure) in its history (6). Google spent almost $3.5 billion in the fourth quarter of 2014 and $11 billion for the year of 2014.

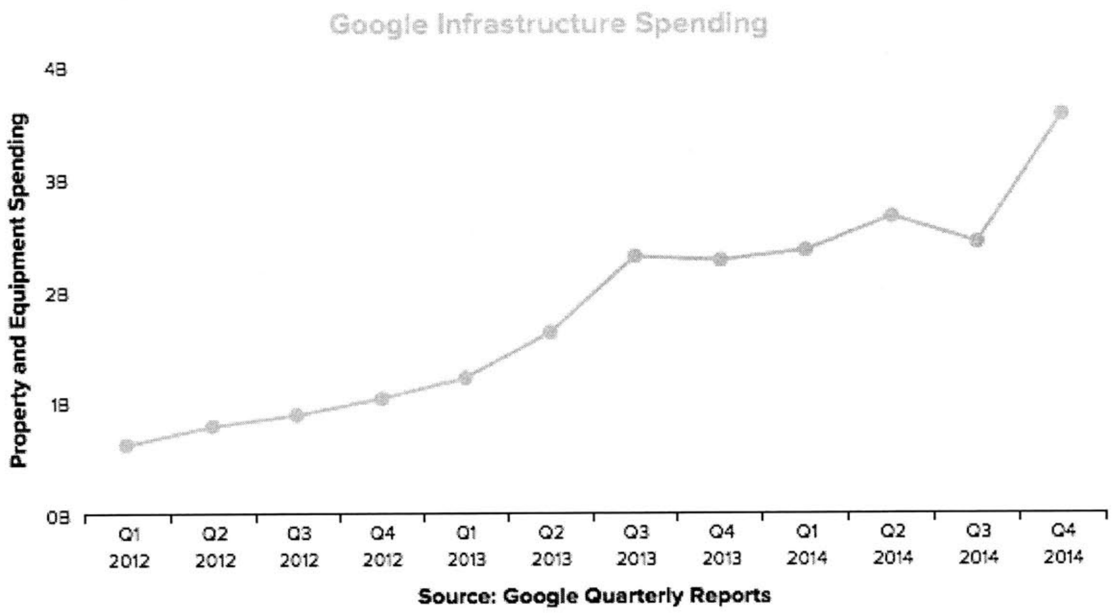


Figure 8: Google's Infrastructure Spending **(6)**

Figure 9: Hyperscale Infrastructure spending over the last four quarters by the Web giants shows the spending of the Web giants over the last four quarter and each of them spent well over a billion dollars in order to build their Hyperscale data centers. This represents a significant investment in the Hyperscale Computing model and represents dollars that would have otherwise been spent on the traditional hardware defined infrastructure. In a way the cumulative figure of more than $20 billion dollars represent the lost revenues for enterprise equipment vendors that are lost to the Hyperscale or Software Defined deployments

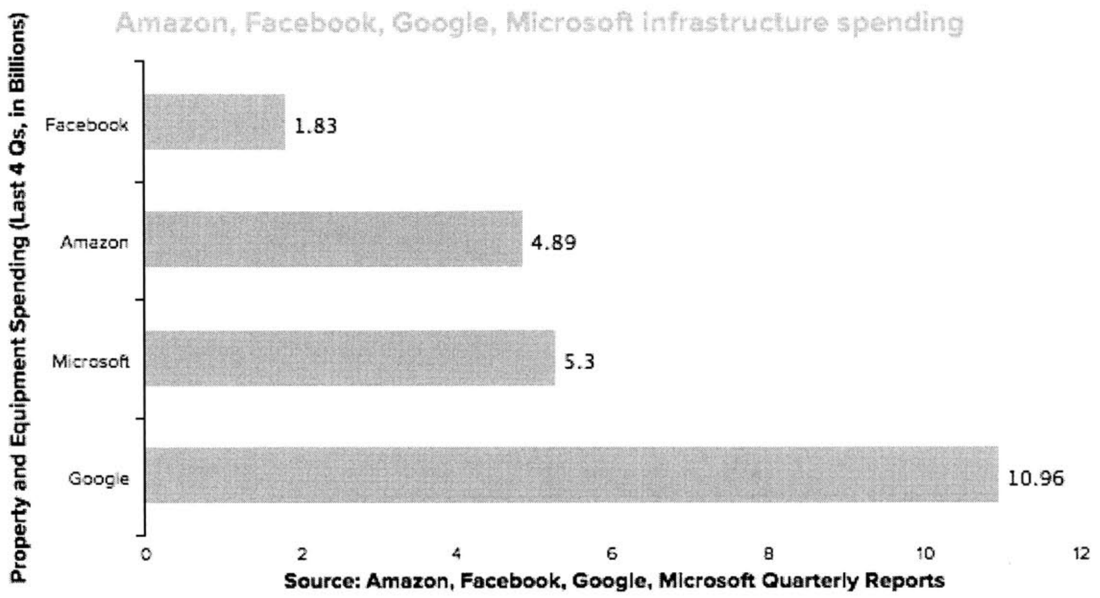


Figure 9: Hyperscale Infrastructure spending over the last four quarters by the Web giants **(6)**

2.3.2 Data Storage in Hyper Scale Environments

In this section we look at the changes to the architecture and design of Data Storage solutions in the hyper scale environment.

2.3.2.1 Google File System

The Google File System is an excellent example of the needs of web scale companies to build infrastructure that meets their requirements and satisfies their use cases and workloads. A File system is a mechanism to store and retrieve information on digital storage media. A file system enables the operating system to logically store data so that it is not merely a binary blob stored on the drives and can be read and written to in an intelligent logical manner. The system that manages the data and its structure is called the file system. A file system stores data usually via a storage block interface which does the raw reads and writes to the storage media. Storage media can vary from hard disk drives to solid state drives to magnetic disks to even Cloud storage objects. There are several different kinds of file systems each of them specifically targeted towards workloads and use cases they address better. The Networked File system or NFS is one of the most widely deployed file systems.

NFS uses a client server approach to file storage. The figure below describes this networked storage model. Network attached storage (NAS) system consists of a file server that is responsible for serving out the files stored on the server. It maintains the file metadata, versioning as well as the namespace for the files. It controls concurrency and secure access with credentials to the files.

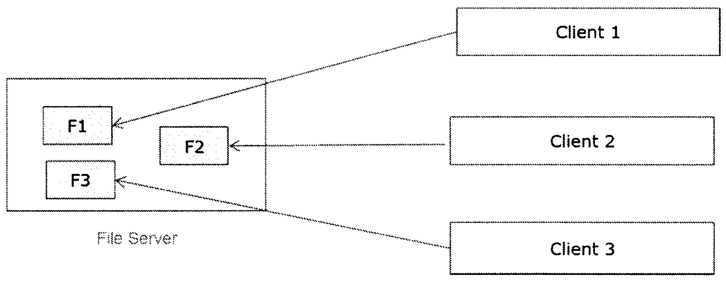


Figure 10: Networked File System

Each client that needs to access the file on the server makes a remote procedure call to the file server over the network in order to get a file handle to access the data in the file that it requests. This architecture works very well for traditional and transactional network access storage use cases. Google's initial approach to file data storage was to use NFS as their primary file system for data storage. However, based on their application and workload input output patterns as well as their unique technology architecture it was obvious that Google needed a file system that was significantly more scalable and distributed than what was available either commercially or in the open source community. Thus was born the Google File system based on two important design points, which are also the ones behind the overall Google architecture.

- The first main design consideration is to provide fault tolerance since the file system itself and the data stored is on inexpensive commodity hardware that can fail. All hardware components fail and that failure must be considered the norm than the exception. The data stored on Google's system spans hundreds or thousands of machines built on commodity components and accessed by commodity client hosts. So there is a high probability of failures. In such cases the distributed file system that spans those be fault tolerant and self-healing in order to maintain the aggregate performance throughputs.

- The second difference in the way traditional file systems or commercial file systems are used and the way Google's unique requirements are. Enterprise file systems mainly serve out a traditional NAS environment or a transactional NAS environment for both small sized data chunks must be read and written frequently. This is in stark contrast to the application and workload patterns for Google, which center around large files of the order of terabytes with large streaming reads and large sequential writes. Thus the design considerations around input output operations and block size need to be optimized for such workloads.

The access pattern involves essentially appending data to a file rather than overwriting existing data, which leads to a sequential write access pattern for, both reads and writes. Thus a sustained throughput becomes extremely critical than peak response time for low latency.

Lastly, the ability to co-design applications and file systems APIs together enables to optimize the file operations based on the needs of the applications (7). It helps Google and other Hyperscale providers that they have a finite number of applications with more of less well-defined access patterns that can be used to customize the underlying infrastructure like file systems.

The Google File System offers a traditional file system interface to applications. It organizes files in an hierarchical fashion in directories and can be addressed using Linux like file path names.

Standard file system operations like creation, deletion, opening, closing and reading and writing of files is permitted in addition to offering capabilities like Snapshotting and Appending. The Snapshot capability enables fault tolerance by creating a copy of the file or directory at a low cost while appending allows multiple concurrent accesses to append data to the file (7).

Figure 11 shows the Google File System Architecture.

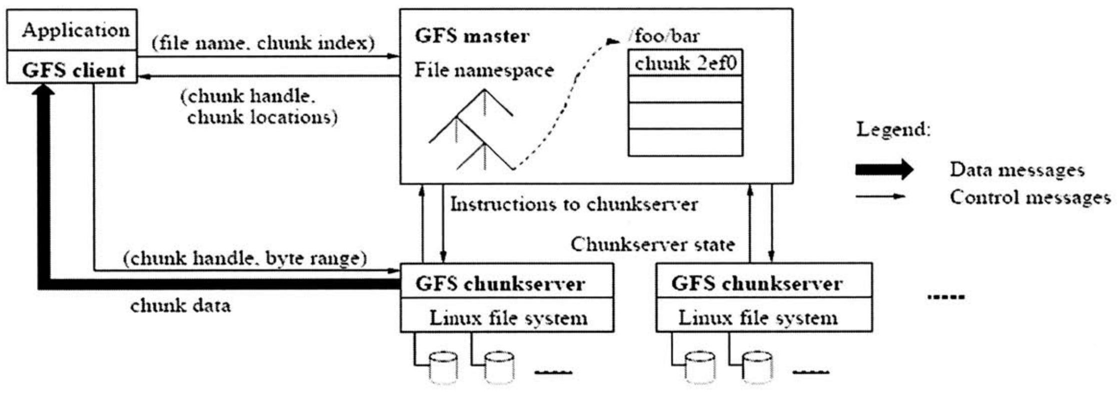


Figure 11: Google File System Architecture **(7)**

The GFS cluster is comprised of three main components -

- GFS Master
- GFS Chunk Server and
- GFS Client

Each of the components is running on commodity hardware running Linux as its operating system. All files are divided into chunks of fixed size and each chunk can be uniquely identified by a chunk handle. To provide reliability each chunk is replicated on at least three chunk servers. The GFS

Master is the one that manages the entire operation. It stores the metadata for the files and knows which chunk servers the various chunks of a file are located in. IT also makes policy decisions around security, access control as well as managing the lifecycle of those chunks. As shown in the figure the GFS master processes control messages and takes part in the initial setup and is not part of the data plane processing. Since there is a single master it is important that the data path does not go through the GFS master otherwise it becomes a bottleneck for the entire system.

Figure 12 explains at a high level the operation of the Google File System. GFS enables multiple concurrent accesses to the file. The figure below is called a GFS Cluster and it consist of one GFS master and many GFS chunk servers. Each file is itself divided into chunks of smaller size with each chunk of 64megabytes in size. This is a much bigger chunk size than typically used to account for the large size of the file itself otherwise the management of chunks itself becomes extremely inefficient for the file system. The chunks are then distributed across multiple chunk servers with the GFS master keeping the metadata of the chunks and their placements across chunk servers.

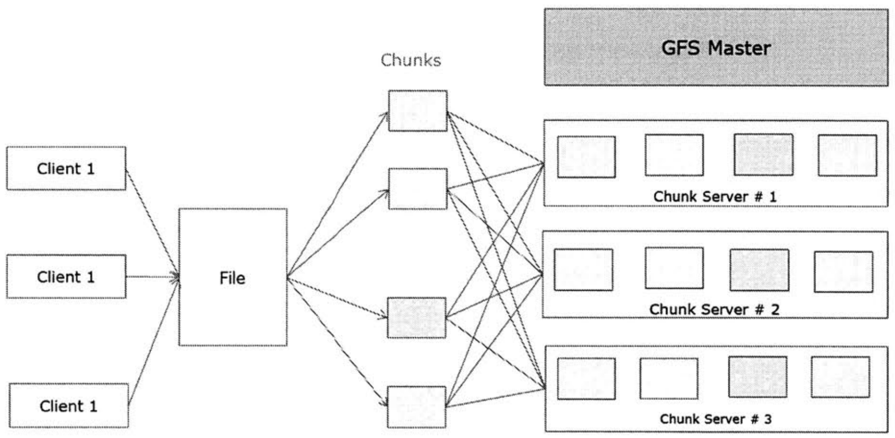


Figure 12: Google File System Structure

Google has deviated from the storage approach taken in large Enterprises where data is store on large dedicated Storage arrays. The redundancy is provided by protocols like RAID (Redundant array of Independent Disks) which ensures data availability in spite of drive failures. Google built their storage on the premise that drives are inherently fail and adding the resiliency in the software via replicas that span multiple systems worked better for their use case.

2.3.2.2 Hadoop Distributed File System (HDFS)

The Google File System inspired other file systems and mechanisms to make efficient use of commodity hardware and implementing software resiliency. One such example is the Hadoop Distributed File System or HDFS.

HDFS is a distributed, scalable and portable file system. Unlike GFS one of the main design points was portability and hence it was written in Java. HDFS also provides a familiar file system interface to its applications but does not support Linux-like or Windows interfaces.

Similar to the GFS, the HDFS consists of a

- Hadoop Cluster
- NameNode – this is equivalent to the GFS Master Server
- DataNode – equivalent to the GFS Chunk Server

The file itself is broken down into smaller pieces called blocks similar to the chunks in the GFS. HDFS provides resiliency by replicating content across multiple data nodes. Additionally the NameNode knows the rack and data center a DataNode belongs to and is thus able to provide availability across racks or data centers. Like GFS, HDFS provides data integrity via checksums. Webscale companies like Facebook and Yahoo are major users of Hadoop. The Hadoop cluster at

Facebook is rumored to be the largest cluster with over 100 PetaBytes of data being stored across thousands of machines and one that grows at the rate of roughly half a petabyte a day (8).

2.3.2.3 Facebook Storage - HayStack

Facebook is the world's largest online social network with over 1.3 billion users as of June 2014 (9). Given their massive user base and services like Facebook Photos while involves users uploading and sharing their photos with their friends – Facebook requires a massive infrastructure to support its business. As Facebook started building their infrastructure they ran into a similar problem like Google when it came to its infrastructure needs and ended up building its own infrastructure in much the same way Google did. Facebook went one step further and open sourced its data center designs as part of the Open Compute Project.

Like Google, Facebook faced challenges on the data storage front due to the unique characteristics of its applications. Facebook stores more than 260 billion photos, which roughly translates to more than 20 petabytes of data storage. Traditional data storage equipment would become expensive very quickly. So Facebook needed an inexpensive way to store these pictures. To achieve that Facebook built and object store called Haystack to store photos. Just like Google, Facebook's original solution involved using the enterprise networked storage arrays running the NFS protocol.

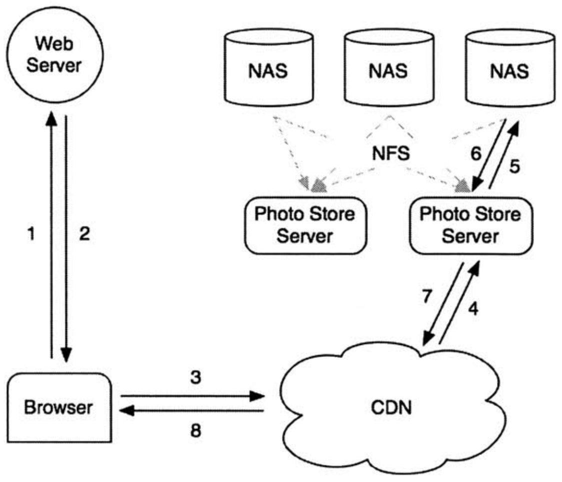


Figure 13: Facebook's Original Photo Storage Architecture (10)

The key issue for using NFS was the multiple numbers of disk input output operations that were required to read a single photo made the operation expensive and a bottleneck. Thus Facebook had a clear business need since none of the existing solutions actually worked for them. Facebook had very similar design points for the development of their Photo storage as Google. They centered on -

- Software resiliency (Fault-tolerance)
- Cost-effective
- High throughput/Low latency
- Simplicity.

These design considerations directly flow from the unique characteristics of the application they were running which in this case was Facebook Photos.

From their customer usage data, Facebook was able to identify the access patterns for their workload, which were very different from other workloads and also were very time specific. The typical action of a user that interacts with Facebook Photos is to upload a photo or retrieve a previously uploaded photo. A user that uploads a photo to Facebook would want to share it - as more friends of the user view or comment on a picture it generates more data to be read and written to the data storage system. This process is extremely time sensitive and the requests for access to those pictures reduce over time. The data is in high demand on its first few days and then the demand for its access falls off significantly. However, even though the data may not be accessed frequently after some time it is important to give users a seamless and real time experience to retrieve old pictures otherwise the experience of the user in interacting with the application isn't favorable anymore.

The architecture of Facebook's Haystack Storage system is as shown in Figure 12 (10).

The architecture consists of three main components

- Haystack Store
- Haystack Directory
- Haystack Cache

The Haystack Store is the management layer that handles the file system metadata for photos. The Haystack directory is the one that maintains the logical to physical mapping for the files being stored while the Cache provides faster access to recently retrieved data.

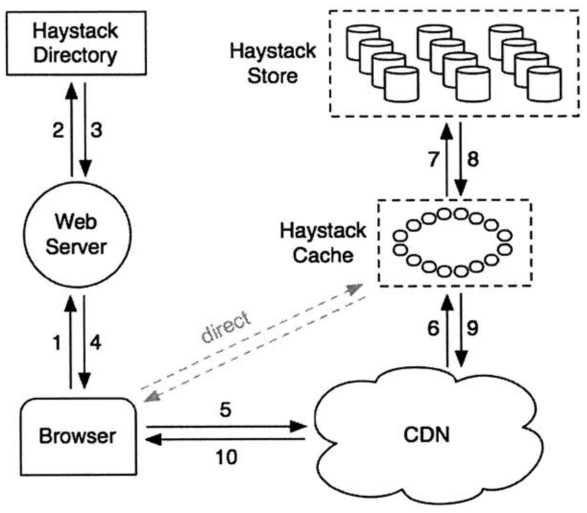


Figure 14: Facebook's Haystack Storage **(10)**

When a Facebook user visits a webpage with photos, the webserver queries the Haystack Directory to construct an URL for the photo (Steps 1-4). The URL contains information on the CDN, Cache and the machine in the Store from where to retrieve the photo.

The URL looks like this –

Http ://(CDN)/ (Cache)/ (Machine ID)/ (Logical Volume, Photo)

One the photo retrieval request is made the system checks the Cache first for the data, if it is a Cache hit then data served out of the cache, on the other had if it is a cache miss, the Directory strips the cache header and retrieves the data from the machine in the Store.

These two examples demonstrate the challenges HSPs face using traditional IT infrastructure in their data centers and shows the design considerations for the infrastructure being built to satisfy their requirements.

3. Technology Architecture

This chapter focuses on the technology architecture of hardware and software defined infrastructures and, in particular, analyzes the architecture of Software defined infrastructure in detail.

3.1 Hardware Defined Infrastructure

Traditional or Hardware defined infrastructure (HDI) is a tightly coupled and often proprietary infrastructure that is built by IT vendors to serve a specific role in the IT computing environment. Infrastructure components can be divided into two main building blocks:

- Hardware – the physical form of the infrastructure component and
- Software – the logical form or function of the infrastructure.

The Software runs on the hardware to deliver a value-added function. The hardware is specialized or purpose-built by the IT vendors and often serves as a competitive differentiation. In the networking domain, IT vendors like Nortel, Lucent and Alcatel utilized their own manufacturing while they designed and constructed purpose-built hardware. The Software teams in these companies then build customer software to take advantage of this proprietary hardware in order to deliver a reliable, resilient and a high performing system. With a significant investment in research and development for both hardware and software and the advances in technology innovations, the IT vendors continued delivering high performing hardware defined systems. The main characteristic of these systems was defined by the hardware they used - hence the classification as hardware-defined.

Networking routers and switches, for example, had custom ASIC forwarding substrates designed in hardware to enable fast packet processing at the rate of the transmission media. The equipment vendors did an excellent job of tightly coupling the various components and building reliability and resiliency into the products because of that tight coupling. These hardware-defined systems are expensive and being built on custom hardware that is extremely rigid. There is no way to change the function or the personality of the system after it has shipped from factory. Thus, if a customer wants to change the function of a piece of equipment it is impossible to do so with the existing hardware defined infrastructure model. In addition, the development cycle to build one of these was usually a multiyear cycle, thus stifling innovation.

Figure 13, below shows a generic hardware defined infrastructure element. It consists of special purpose hardware tightly coupled with the software-operating environment. This software environment can itself be further decomposed into two distinct abstractions – the control plane and the data plane.

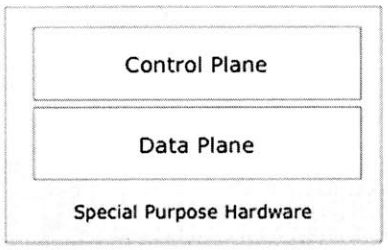


Figure 15: Hardware Defined Infrastructure

The Control Plane is the Software that as its name suggests controls the behavior of the data being processed by the system while the Data Plane is the Software that performs the actual transformation of the data. The transformation function itself varies by the actual purpose of the system. For example in a network switch, the data plane is responsible for forwarding the data to the next hop element. In case of storage appliances or arrays the data plane is the one that performs the core function of storing the data on the storage media in the array.

The main characteristics of hardware-defined infrastructure include:

- Reliability
- Resiliency
- Performance.

In addition, they are also criticized as:

- Expensive
- Vendor dependent (lock-in)
- Static
- Not flexible or nimble.

Detailed system architecture of a network router is as seen in Figure **16**.

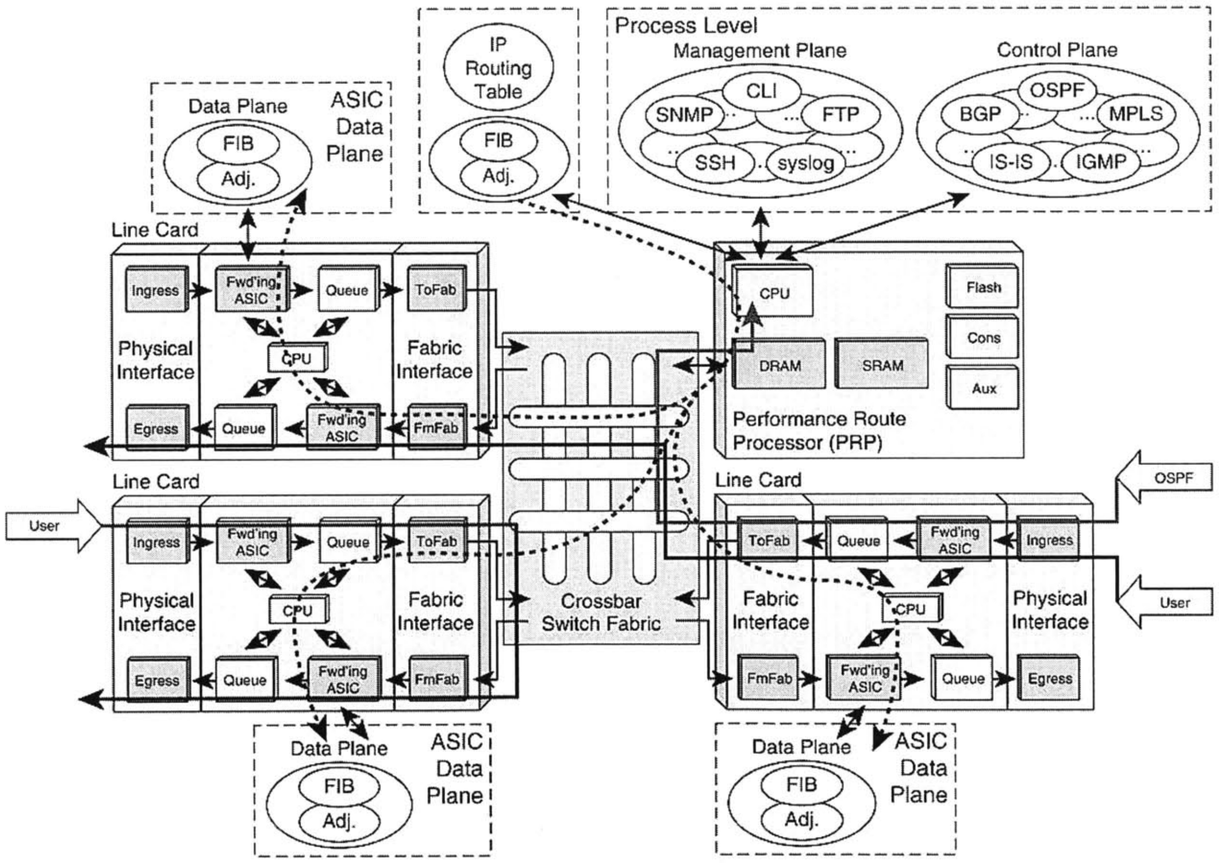


Figure 16: System Architecture of Cisco 12K **(11)**

The complexity and customization of the hardware is evident from the different components of this architecture. These hardware designs are proprietary and carry the vendor's 'secret sauce' that enables them to offer significant differentiation along the performance curve. The above system architecture shows two major components—the ASIC Data Plane on the line cards that perform the actual packet switching - these line cards are specifically built to offload the forwarding function from the CPU to dedicated ASICs that operate independently. This independent region facilitates secure performance at the same rate as the speed on the links connecting the router. The second component is the Control Plane that programs the forwarding entries on the line cards based on a routing or a switching algorithm applied to their view of the network and the information exchanged with their neighbors.

3.2 Software Defined Infrastructure

Over the years, with advances in chip technology, and thanks to Moore's Law, the computing power of general purpose off the shelf hardware started becoming good enough for a majority of the use cases to replace at least portions of the infrastructure - infrastructure that did not require the highest performance or wasn't mission critical. This, coupled with advances in Open Source Software like Linux, led to a decoupling of the software from the hardware. Around the same time frame a technology called Virtualization, which abstracted the Operating System from the underlying hardware, was gaining significant momentum. This led to the creation of a Software Defined or a Virtualized Compute model where the Compute resources could be virtualized and run on a common general-purpose hardware (server). No longer was there a need for a specific server platform but any server platform that met certain minimum specifications could be virtualized to run applications that may not have previously run on that platform.

As we described in Chapter 2 of this thesis, the web giants like Google, Facebook, Amazon and Microsoft saw this as a way to build massively scalable data centers at significant lower costs. These companies needed to build huge data centers in order to enable their business. In their case, the data center wasn't just a means to support the business; it was their main business to offer applications running in their data centers. Recognizing that fact, they quickly realized that traditional data center equipment was too expensive as well as too rigid for their purposes. They wanted something that was agile and cheap—so that they can buy it in huge quantities directly from the source, and write software that can programmatically manage and control them. This led to the creation of the concept of Software Defined Infrastructure.

This new model of building a Computing infrastructure, and the success the web giants achieved in doing so, has certainly captured the mind share for Enterprise IT for a variety of reasons.

Software Defined Infrastructure enables delivering infrastructure with different service level agreements or classes of service. This enables an application to utilize the most efficient service level required of the infrastructure. Typically, every application goes through a lifecycle ranging from test and development to a pilot, to an initial rollout to wide-spread adoption. This in the traditional infrastructure meant different physical components to support these different stages. But in software defined infrastructure these phase changes only the services requested delivered on top of a virtualized infrastructure. Thus the application can meet changing usage patterns without disruptions due to the elasticity offered by the underlying infrastructure – the ability to scale up, down or scale out on demand.

In the software defined world as shows in Figure 17: Software Defined Infrastructure – the infrastructure component splits up into two elements. The Control and the Data planes become two distinct physical entities in the software-defined infrastructure with the software that delivers the control or management functionality and the data plane functionality running on general purpose hardware rather than a special purpose hardware system.

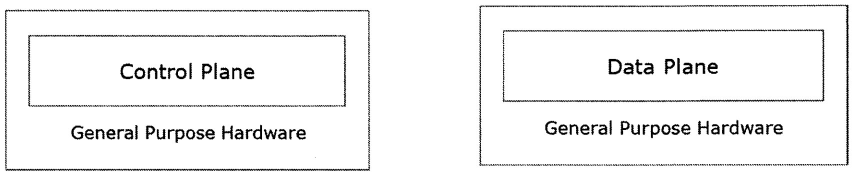


Figure 17: Software Defined Infrastructure

Another closely and often interchangeably used term is Software Defined Data Center, which mainly consists of Software Defined Infrastructure that is used as the foundation of a Software Defined Data Center. A Software-defined data center is a data center built on Software-defined Infrastructure where the underlying infrastructure is virtualized and delivered as a service, and is entirely managed and orchestrated by software.

Software-Defined Data Center automates the orchestration of services across all software-defined components– server, storage, network, and security. In the Software-Defined Data Center, software and hardware components are decoupled from each other. Software components abstract their functionality through policy-based services and, through virtualization, can move in an agile manner.

The main characteristics of software-defined infrastructure includes –

- Agility
- Flexibility
- Low Costs
- Vendor agnostic

Software Defined Infrastructure consists of the following major components –

3.2.1 Software Defined Compute

The trend of Software Defined Infrastructure started in the Compute domain. Traditional hardware defined servers operate at less than 15 percent of capacity. This is because of the need to plan for peak capacity. Virtualization rewrote the entire equation. CPU and memory are decoupled from physical hardware, creating pools of resources for use wherever needed. Each virtualized application and its operating system are encapsulated in a separate, fully isolated software container called a virtual machine (VM) (12). Thus applications that previously ran on different operating systems and were required to have an independent server setup could all be co-located on the same server.

Figure 18: Compute Evolution – From Hardware Defined to Software - shows evolution of compute from the hardware-defined model to software defined one. At the top of the diagram is the hardware-defined model where each application was run on a specialized server. An application running in the Windows environment or a Linux environment would need those operating systems to be run natively on the hardware with the application running in the context of the operating system.

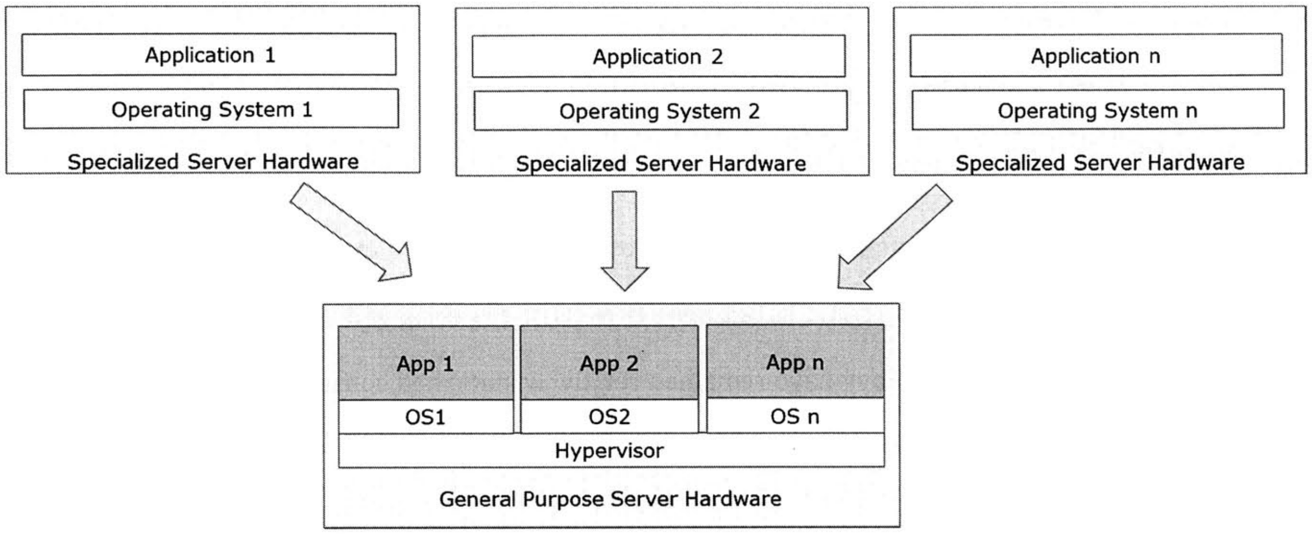


Figure 18: Compute Evolution – From Hardware Defined to Software

However, virtualization changed that—now with a hardware abstraction layer like the hypervisor, many different operating systems could run on the same server hardware with applications running in the context of their respective operating systems. This significantly reduces the dependence on purpose-built server hardware from vendors like Sun Microsystems and leveraged the commoditized server hardware running on Intel's x86 based processing platforms.

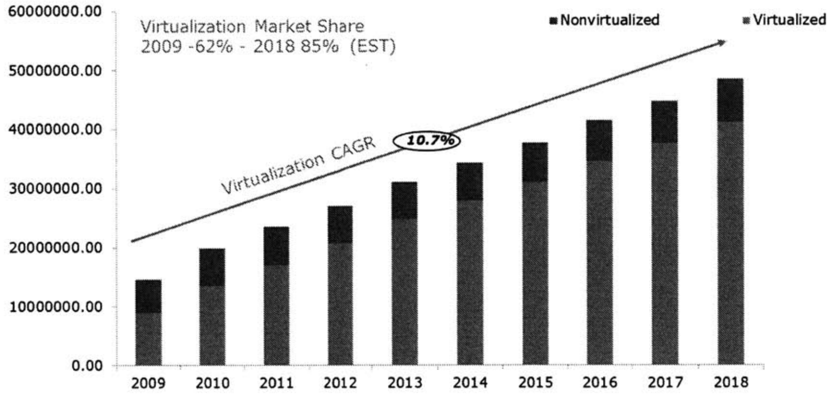


Figure 19: Virtualized Server Market Share

Figure 19 above captures the significant growth in virtualized servers over the last five years and forecasts the growth going into 2018. The virtualized server category is forecast to grow at a compounded annual growth rate of 10% for the remainder of the decade. That is a significant growth, as the fraction of virtualized to no-virtualized server goes up from 60% 2009 to nearly 85% in 2018.

While Software Defined Compute or Server virtualization has seen significant growth and strong adoption over the years, the same hasn't been true for networking and storage components of the computing infrastructure. They have remained relatively static and consequently have become the inhibitors for a more agile computing environment.

3.2.2 Software Defined Networking

Software-defined networking (SDN) is an approach to computer networking that enables separation of the control and data plane referred to in the previous section so that networking services could be efficiently managed and deployed via software abstractions.

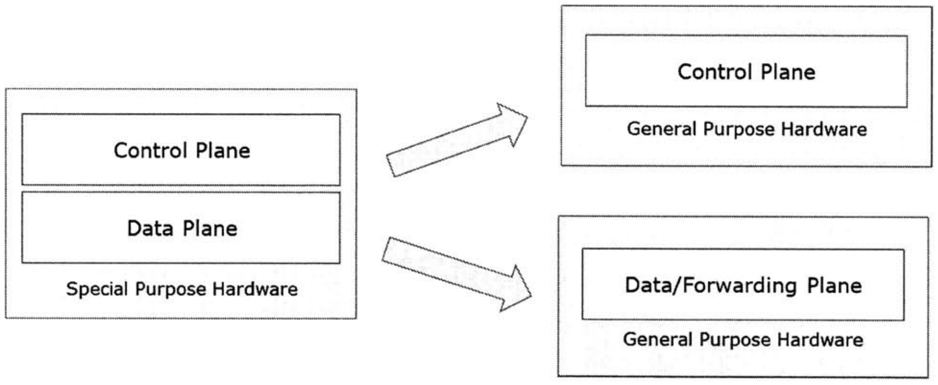


Figure 20: Software Defined Networking

The Figure 20: Software Defined Networking - shows the transformation of hardware defined networking router or a switch to software defined one. In the hardware-defined switch on the left of figure x, the control plane and the data or the forwarding plane is tightly coupled in a specialized hardware complex. To transform the hardware-defined switch to software defined, the control plane and the forwarding plane logic is separated. By isolating the two components the processing requirements reduce to be able to take advantage of commodity server hardware to deploy the two components separately.

Figure 21: Software Defined Switch – shows the architecture of a Software Defined Switching element. The SDN controller controls the protocol logic as well as programming of the forwarding table for the commodity switch elements that enable the any-to-any connectivity network.

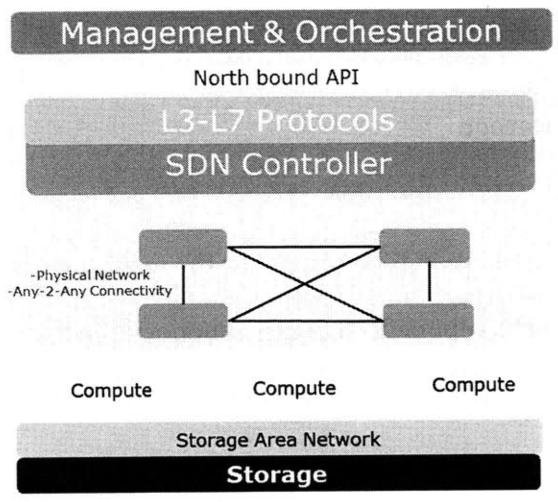


Figure 21: Software Defined Switch

The northbound APIs exposed by the SDN controller can be manipulated by the management and orchestration layer through an application to change the characteristics of the network programmatically.

In the networking domain, the move towards software defined has been driven by the evolution of technologies like mobile, server virtualization and Cloud. Traditional hardware-defined networks were typically hierarchical, which worked very efficiently in the client-server phase of computing evolution. The network architecture mirrored the computing infrastructure in that both were static and the traffic followed a well-defined north-south pattern between a client and server. However, virtualization and Cloud has generated significant cross or east-west traffic patterns for which the traditional networking architectures were ill suited. Comparing the Software Defined switch architecture in figure 20 with the hardware defined switch architecture in figure X brings out a striking contrast to the hardware complexities of hardware defined systems. This hardware complexity is directly reflected in the cost of the components and in turn on the price of the system sold by the networking vendors.

Comparing this with Figure **16**- brings out a notable difference and cost savings between hardware and software defined infrastructures.

3.2.3 Software Defined Storage

Software-defined storage (SDS) describes storage technologies that disaggregate the storage software from the hardware enabling the control or the management of the underlying stored data to be managed by a separate component than the component that actually stores the data.

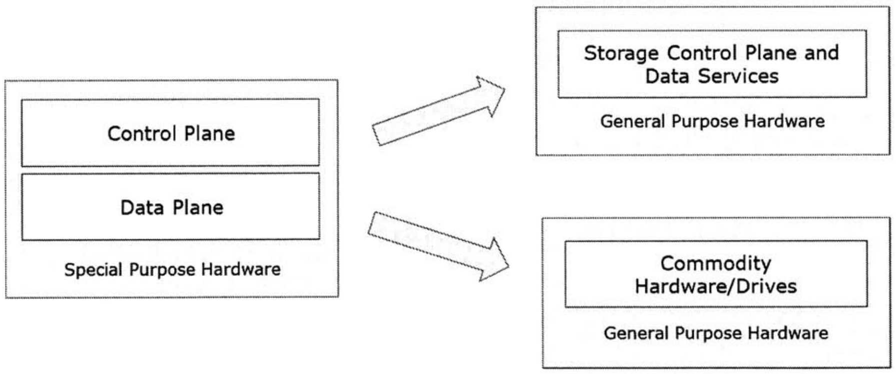


Figure 22: Software Defined Storage

Figure 22: Software Defined Storage - above shows the differences between a hardware defined Storage system and a software defined system. A hardware defined storage system like other hardware defined infrastructure components is tightly coupled and purpose built. This enables a high performance and resilient storage experience. A Software Defined Storage system decouples the storage software from the hardware and enables using various different storage media and architectures to store the data using the same set of storage control plane and storage data services like data replication, data encryption, and data compression for all the heterogeneous storage end points. This disaggregation reduces cost significantly while also improving agility and flexibility of storage deployments. Software Defined Storage has also given rise to a new kind of architecture called the Hyper Converged infrastructure where the Storage and the Compute elements are co-located on the same hardware and thus providing a building block for the data center that can be expanded one brick at a time.

3.2.4 Software Defined Management

Managing the Software Defined Infrastructure is one of the most critical aspects of the new architecture. Cloud Management Platforms or CMPs have sprung up to undertake the role of the management and orchestration layer that can automate the life cycle management of the infrastructure. Since the infrastructure is software defined, it enables the management platform to spin up assets dynamically and enable giving the hardware a different software profile by installing the appropriate software version on it. OpenStack is an open-source CMP that provides capabilities necessary to build a cloud IaaS offering, including virtual infrastructure management and self-service provisioning. It is not a full-featured enterprise CMP; rather, its focus is on capabilities for developer enablement and programmatic access to infrastructure. OpenStack is under the governance of the OpenStack Foundation, a nonprofit consortium sponsored by numerous IT vendors including Cisco, Dell, HP, IBM, Rackspace, Red Hat and VMware (13). The figure X below shows a typical Software Defined Management layer and it's various components that enable the management and orchestration as well as the telemetry and analytics needed to operate the software defined infrastructure.

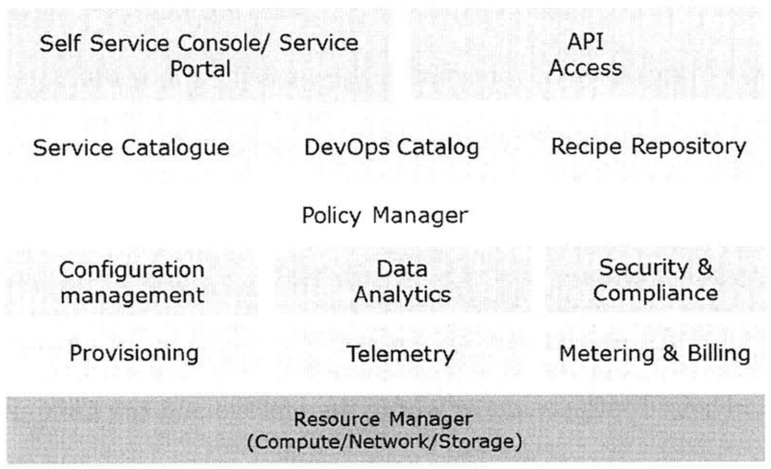


Figure 23: Software Defined Management

The most common use cases for OpenStack are around building

- Inexpensive, elastic platform for batch computing.
- AWS-like private cloud.
- Infrastructure foundation for "private platform as a service.
- Hardware agnostic private Clouds
- Cloud-enabled service or cloud IaaS offering for customers.

An Open Source CMP like Openstack has a very important role to play as the Cloud Operating System of a Software Defined Infrastructure.

In summary, the Software-Defined Data Center truly represents major changes in the way infrastructure resources will be built, delivered, consumed, connected and managed to attain maximum agility, operational efficiency, and extensibility. Software defined infrastructure extends the benefits of virtualization - which most customers have already realized from the compute side – to other data center infrastructure. It abstracts leverages and pools assets and asset intelligence, integrating that with the intelligence at the virtualization layer to facilitate policy-based automation. This automation lays the foundation for infrastructure as a service by enabling our clients to provision in a very rapid and dynamic way. It provides a single plane of glass, with which the customer can manage existing heterogeneous infrastructure as pools of infrastructure resources, enabling them to decouple the data plane from the control plane, and provide resource access via APIs, as opposed to hard coded integration.

The architecture diagram of a Software Defined Data Center is as shown below.

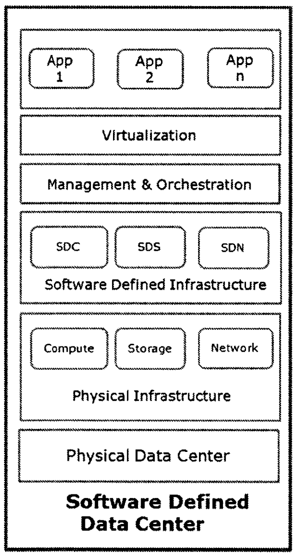


Figure 24: Software Defined Data Center

Figure 25: Data Center Characteristics - below highlights the main differences between traditional data centers and hyper-scale data centers. Hardware Defined Data Centers utilize shared resource

architectures, are built on proprietary technology stacks and have large failure domains. They are built on infrastructure that provides reliability, resiliency and can scale up vertically as needed.

Characteristics	Traditional Data Centers	Hyper Scale Data Center
Key Drivers	Consolidation and Optimization	Elasticity, agility and lower TCO
Architecture	Shared resources architecture	Shared nothing architecture
Elasticity	No elastic compute or elastic storage	Elastic compute and storage
Agility/Flexibility	Low	High
Applications	Traditional customized enterprise applications	Cloud applications written to the 'design for fail' paradigm
Provisioning	Manual Provisioning	Automated Provisioning
Failure Domains	Large failure domains	Smaller failure domains
Technology Stack	Integrated and Proprietary technology stack	Distributed and open technology stack
Product type	Proprietary	Open

Figure 25: Data Center Characteristics

Hyper-scale Data Centers, on the other hand, are built for agility and flexibility. They utilize 'shared nothing 'architecture, have smaller failure domains and are built on scale-out and often open technology stacks.

3.3 System Architecture

Software Defined Infrastructure defines new system architecture for computing infrastructure where the software component of the infrastructure is decoupled from the underlying hardware and virtualized to enable delivering infrastructure as a service in an automated manner. This signifies a paradigm shift in the way we think of some of the most critical elements of our computing architecture. The Software Defined aspect of it refers to the capability to define the role and purpose of an infrastructure element based on the Software personality it is given.

The System Architecture for a Software Defined component is as shown in the Figure 26: SDI System Architecture. The System architecture shows the interactions between the various logical components in a Software Defined system. The Control layer would be physically on a different hardware than the Data Layer and the Management layer. It controls the behavior of the Data layer and exposes application-programming interfaces Northbound to the management application in order to programmatically configure and manage these elements.

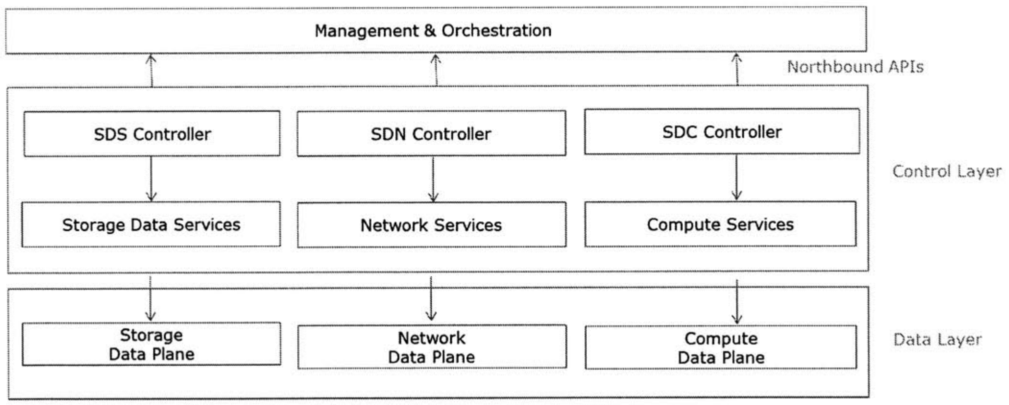


Figure 26: SDI System Architecture

The System Architecture of a Software Defined Infrastructure component can be decomposed as follows

- System Problem Statement: To *enable* agility, flexibility and reduce costs *by disaggregating* the software from the underlying hardware and using software defined infrastructure *to deliver business value* and reduce costs for customers.
- Description: A flexible and agile infrastructure component that can be deployed dynamically as a software unit on commodity hardware.
- Concept: The concept of a Software Defined Infrastructure component is to abstract the hardware and disaggregate the software layer from the hardware layer in order to deploy the software function on generic hardware to reduce complexity and increase agility and flexibility.
- Form: The form consists of the software code written to deliver value to the stakeholders.
- Function: The function is the abstraction and virtualization of the underlying hardware for the layers above; so that the software can be deployed in a hardware agnostic manner.

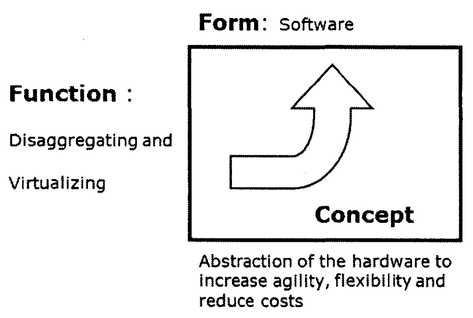

Form: Software
Function :
Disaggregating and Virtualizing
Concept
Abstraction of the hardware to increase agility, flexibility and reduce costs

Figure 27: System Architecture -Function, Form and Concept

- Direct Beneficiary: The consumers or the builders of the Computing infrastructure are the main beneficiaries.
- Primary Benefit: Agility, Flexibility and reduce costs

- Losing stakeholders: EEV's – the traditional hardware defined equipment players are the losing stakeholders.

As a next step to analyzing the architecture we look at the different layers of the system. The Figure 28: SDI System Architecture Layers - below shows the SDI architecture in layers. It shows the operand, the hardware on which the system acts the solution neutral transformation of flexible and agile deployment by abstracting the hardware and disaggregating the software to run independent of the hardware beneath.

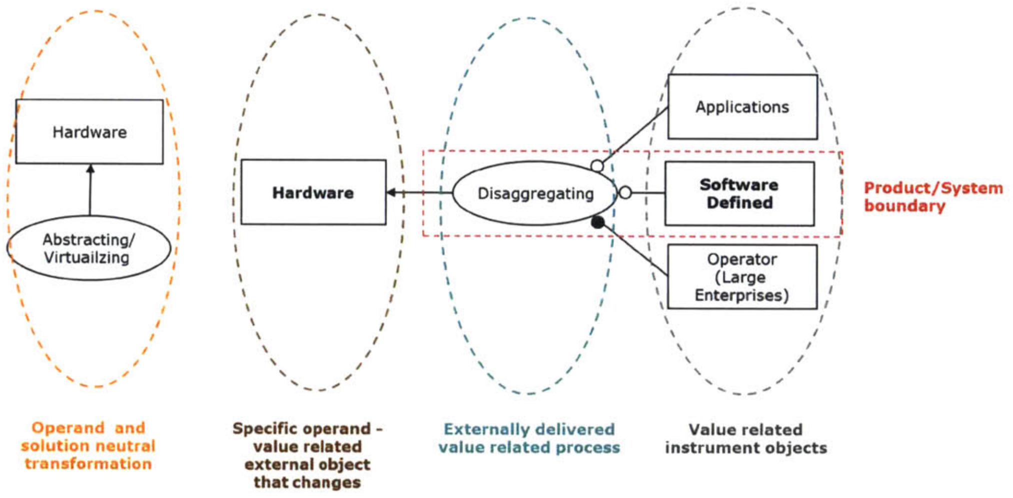

Hardware
Abstracting/ Virtuailzing
Hardware
Disaggregating
Applications
Software Defined
Operator (Large Enterprises)
Product/System boundary
Operand and solution neutral transformation
Specific operand - value related external object that changes
Externally delivered value related process
Value related instrument objects

Figure 28: SDI System Architecture Layers

The Operand or the component that is acted upon by the architecture is more generically the hardware. The solution neutral transformation is to abstract or virtualize the hardware so that the linkage between the software and special purpose hardware can be broken thus enabling the software to be loaded on commodity hardware without any special characteristics. The externally delivered value related process is this disaggregation of the software from the hardware. This critical process of separating the tight coupling between hardware and software is responsible for the externally delivered values of agility, flexibility, lower costs and avoiding vendor lock-in.

3.4 Design Principles

The computing architecture has under gone a significant change in its design principles with the advent of Software Defined Infrastructure. Software Defined enables Hyper Scale - the ability of architecture to dynamically scale as the demand changes. It involves the ability to seamlessly provision and adds compute, memory, networking, and storage resources to a given node or set of nodes that make up a larger computing infrastructure. As we have seen these capabilities are crucial in order to build a robust and scalable cloud, big data, map reduce, or distributed storage system and are often associated with the infrastructure required to run large distributed sites such as Facebook, Google or Amazon (6) (7) (8) (9). We identify four key design tenets for the system architecture of a Software Defined Infrastructure

3.4.1 Design Principle # 1 Software Defined Everything

This is where Software Defined Infrastructure meets Cloud or Hyper Scale Infrastructure. One of the main principles for Hyper-scale is that everything should be available as software. A virtual instance of a server, a storage device or a network router should all be abstracted and be available as a software component so that they can be spun up dynamically as needed. All Software must run on standard x86 based hardware, with no special purpose dedicated components. Customers must be able to purchase software and hardware independently of each other and then put them together based on the software specifications.

3.4.2 Design Principle # 2 – Disposable Components - The Pets versus Cattles Metaphor

Hyper-scale architectures are based on a shared-nothing architecture that heavily relies on Virtualization. Shared-nothing architecture is a distributed computing architecture in which each node is independent and self-sufficient and the nodes within the system do not share memory or disk storage. This is in contrast to shared-memory or shared-disk architectures where there is a common memory or disk element shared across multiple nodes. Prior to the advent of virtualization application servers and other infrastructure components were managed closely by IT. The reliability of the applications and the data center in general depended on the reliability of the infrastructure below it. In the Enterprise world a reliability of 5 9's has become the gold standard. Infrastructure vendors pride on the ability of their products to provide 5 9's of availability in order to ensure uptime goals, which would meet SLA requirements of mission-critical high performance applications. 5 9's of reliability or 99.999% uptime roughly equates to about 5 minutes of down time (planned or unplanned) throughout the year.

Enterprise applications are designed for this reliable infrastructure and relied on sub millisecond latency. These requirements on applications led to faster connections between infrastructure components as well as high Input Output processing (IOPS) metrics for storage environments. With the advent of solid state drives and the move to an all flash storage array more a million IOPS at a sustained low latency as workloads scaled out has become an expected performance metric of the infrastructure. In addition, Enterprise IT departments fine-tune the infrastructure for the best performance and several develop in-house monitoring tools in order to alert on failures and take remedial action. If an infrastructure component fails, the IT teams and their suppliers work together to dispatch service personnel in order to fix the hardware component. Infrastructure vendors sell maintenance and service contracts to support their products and it has become a significant part of their business model.

Virtualization changed this dynamic. Now instead of a hardware component, the infrastructure elements were virtual machines running in a container. If the virtual instance of the infrastructure went down it could easily be deleted and another one cloned in its place. All the virtual instances ran on homogeneous x86 based servers. If the server itself went down another identical server replaced it with the same VM templates applied. Hyper Scale and Public Cloud providers automate the process of decommissioning a failed server and bringing another bare-metal server online – all done via software.

These contrasting approaches were captured in the Pets versus Cattles analogy presented at the 'CERN Data Center Evolution' in 2012 and is attributed to Microsoft Distinguished Engineer Bill Baker (14). The service model in the traditional enterprise data center can be compared to that of taking care of a Pet. Pets are given names like – Pluto-the-dog - they are unique, lovingly raised and cared for. When they get sick they are carefully nursed back to health. Cattles on the other hand are given numbers like vm.app.4301010.mit. They are almost identical to one another or at least treated as identical to one another. When they get ill, they are discarded and replaced with an identical one.

The above analogy captures the significant differences between the service models of the traditional IT infrastructure and Cloud infrastructures. IT applications are designed from an enterprise perspective – on a reliable infrastructure with services that are tightly coupled and rely on scale-up systems and relational databases. They are lovingly and expensively cared for, with great attention and affection. On the other hand in hyper scale environments infrastructure components are treat like cattle, if one goes down it is decommissioned and replaced with another in a matter of no time. Low cost commodity general-purpose hardware enables hyper scale providers to take that action because often the cost to repair a failed system both in terms of equipment and manpower is more than the hardware that could replace it.

3.4.3 Design Principle # 3 - Design for Failure and Design for Reliability

There are two main principles that application architectures can be based on–

- Design for failure and Design for Reliability.

These two are diametrically opposite paradigms for building an application. One assumes everything will fail eventually and that every application must have resiliency built into it while the other passes on the onus of reliability to the underlying layers. One of the most important design principles for hyper scale applications is the need to design for failure. Given that most of the infrastructure in the hyper scale environments is more cattle-like, applications need to be built such that they do not expect resiliency from the infrastructure but can continue functioning in the event of a failure.

Traditional IT applications can be classified as hardware resilient applications. These applications like Oracle and SAP databases rely on the underlying hardware to provide redundancy. The applications are built assuming a highly reliable infrastructure. This means they are less tolerant to infrastructure failures and have extremely low latency requirements. Applications usually time out and fail if the underlying infrastructure does not provide the kind of resiliency they are looking for. However, Cloud applications are built based on an infrastructure that could be unreliable. Hence the resiliency must be built into the application to account for unreliable infrastructure underneath. As we discussed in Chapter 2 – hyper scale providers like Google and Facebook have redundancy built into their applications so that failure of a single infrastructure component does not lead to a degraded application experience.

3.4.4 Design Principle # 4 – Fail fast and frequently

As explained above, for the next generation of applications, the resiliency expected of the infrastructure will be built in to the application. This is true of massively distributed application built in the Cloud today. Netflix for example, runs a highly decentralized application comprised of independent services that are aggregated to provide specific functionality. Each of the services operates separately, and the resulting application is unique for every user. The best way to test the resiliency of such applications is to fail fast and fail often. By frequently causing failures, applications are forced to be built in a way that is more resilient.

Companies like Netflix whose entire business and customer satisfaction depends on accessibility of their services anytime of the day and from anywhere chose to build upon an unreliable infrastructure by building such resiliency into the application. Netflix developed a tool to randomly shut down instances within the underlying Amazon Web Services infrastructure to ensure its applications are robust enough to handle failures of the underlying resources. They achieve this through a homegrown tool called Chaos Monkey and have continued development of other tools to improve resiliency. (15) . The basic principle behind these tools is to fail fast and fail often thus enabling a robust and deterministic failure remediation process. This is a completely new approach to managing failures for a production environment.

4. Business Analysis

The emergence of Software Defined Infrastructure will have a significant impact on the business model of traditional IT vendors. This chapter analyzes that impact by looking at the technology adoption, business ecosystem and the impact of SDI on the value chain in the industry.

4.1 Architectural Innovation

Software Defined Infrastructure can be classified as an Architectural Innovation based on the work of Henderson and Clark in their paper on the topic of Innovation (16). Their research describes an architectural innovation as one in which an established system is reconfigured to link together existing components in a new way. It is triggered often by a change in a component such as size or other basic parameters of its design that creates new interactions and new linkages with other components in the established product. However, the core design concept behind the components and the associated scientific and engineering knowledge remains the same.

The Henderson and Clark paper (16) on Architectural Innovation classifies innovation in four different categories based on two factors

- Core Concepts and
- Linkages between the Core Concepts and Core Components.

As shown in Figure 29 – If the linkages remain unchanged but the concepts are reinforced then it is called an incremental innovation. If the linkages remain unchanged but the core concepts are overturned then that forms a modular innovation. When linkages change but the core concepts remain the same then it is architectural innovation while when both the linkages and the core concepts change then it is classified as radical innovation.

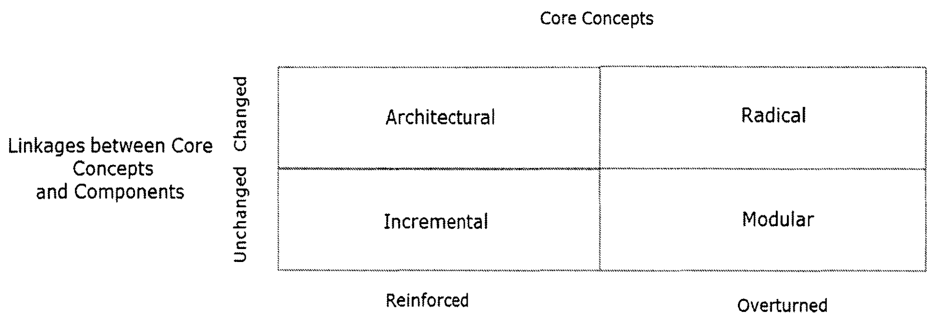


Figure 29: Innovation Framework **(16)**

From the description of Software Defined Infrastructure in chapter 3 it is clear that the components in a system are linked in different ways; however their core concepts and functions remain the same. Hence we consider SDI as an architectural innovation.

(16) Theorizes that - architectural innovations destroy the usefulness of the architectural knowledge of established firms. Given that architectural knowledge becomes embedded in an organizational structure and often becomes the core competency of a firm it gives rise to organizational rigidities which makes it extremely difficult for incumbent firms to identify these trends and shift focus before it's too late. Thus architectural innovation often has significant competitive and business implications. Much of what the firm knows is useful and needs to be applied in the new product; however the rest of its knowledge might actually prove detrimental to its progress and recognizing those differences may not be particularly easy due to the way such learning's are organized and managed within the firm.

4.2 Disruptive Technology

Disruptive innovation, a term coined by Harvard Business School professor Clayton Christensen, describes a process by which a product or service takes root initially in simple applications at the bottom of a market and then relentlessly moves up market, eventually displacing established competitors (17).

Christensen defines a disruptive innovation as a product or service designed for a new set of customers.

> "Generally, disruptive innovations were technologically straightforward, consisting of off-the-shelf components put together in a product architecture that was often simpler than prior approaches. They offered less of what customers in established markets wanted and so could rarely be initially employed there. They offered a different package of attributes valued only in emerging markets remote from, and unimportant to, the mainstream." (18).

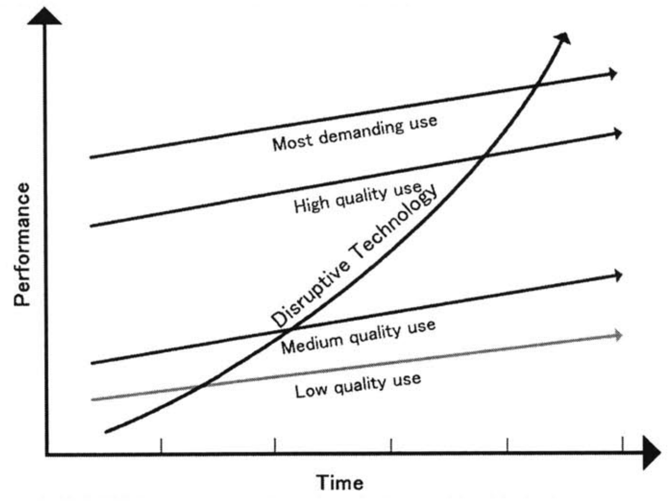


Figure 30: Disruptive Innovation (12)

An innovation that is disruptive allows a whole new population of consumers at the bottom of a market access to a product or service that was historically only accessible to consumers with a lot of money or a lot of skill.

Disruptive Innovations arise when incumbents innovate faster than their customers' needs evolve. Disruption at the low-end occurs when the rate of improvement of products along a generally accepted metric (like performance) continues improving incrementally and eventually exceeds the rate at which customers can adopt the new performance. At some point along that trajectory of product evolution, the performance of the product overshoots the needs of certain customer segments, which means that the product has become too complex or sophisticated for that market segment and that a simpler and more likely cheaper alternative becomes a viable business opportunity. Usually, companies pursue these sustaining innovations at the higher tiers of their markets – because that's where their most demanding customers are who contribute to the highest margins of their business.

This can result in an opportunity for a disruptive technology to enter the market and provide a product which has lower performance than the incumbent but which exceeds the requirements of certain segments, thereby gaining a foothold in the market. In low-end disruption, the disruptor is focused initially on serving the least profitable customer, who is happy with a good enough product. This type of customer is not willing to pay a premium for enhancements in product functionality. Once the disruptor has gained a foothold in this customer segment, it seeks to improve its profit margin. To get higher profit margins, the disruptor needs to enter the segment where the customer is willing to pay a little more for higher quality. To ensure this quality in its product, the disruptor needs to innovate. The incumbent will not do much to retain its share in a not-so-profitable segment, and will move up-market and focus on its more attractive customers. After a number of such encounters, the incumbent is squeezed into smaller markets than it was previously serving.

Finally, the disruptive technology meets the demands of the most profitable segment and drives the established company out of the market.

There have been various examples of disruptive innovation over the years, such as the personal computer disrupting mainframes and minicomputers. The well-documented disruption in the disk drive industry across the various different drive types, and the disruption of circuit switching by packet switching technology in the networking world. I believe that Software Defined Infrastructure is proving to be a classic case of a disruptive technology. When Web scale companies like Facebook, Google, Microsoft and Amazon started evaluating using Enterprise data center equipment they ran into the classic problem of tradeoff between performance and costs. With the massive scale of their data centers it was necessary to have their computing environments architected with different characteristics than the typical enterprise data center. This meant that they traded off performance for costs and used parallelization to gain some of the performance benefits. In that sense they were the low-end or the least demanding of customers for EEVs.

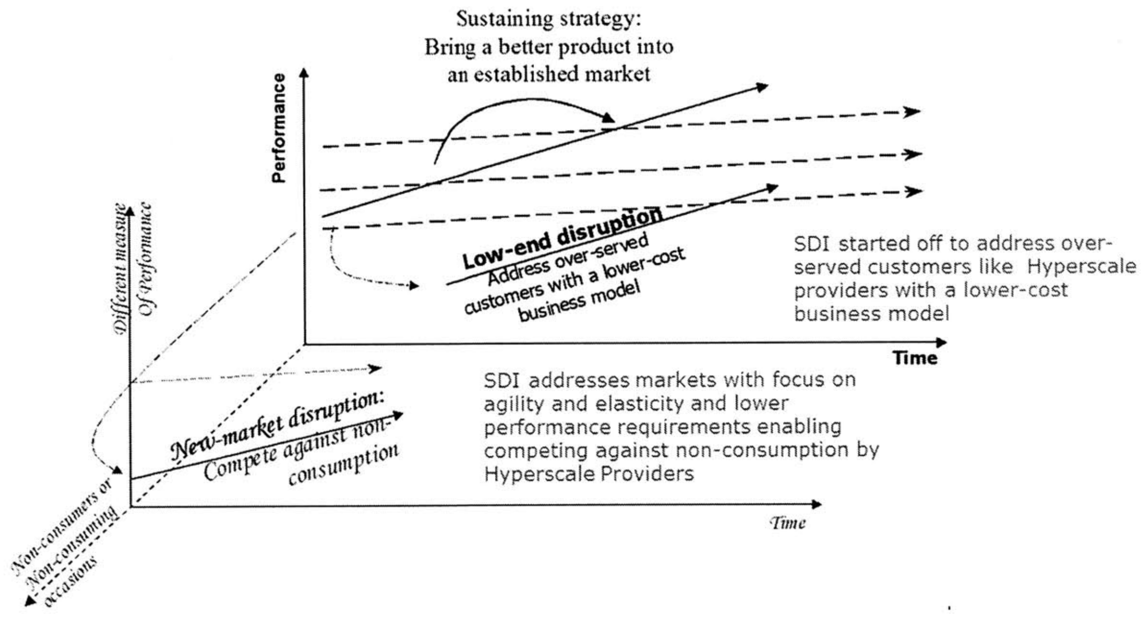

Sustaining strategy: Bring a better product into an established market
Performance
Low-end disruption
Address over-served customers with a lower-cost business model
SDI started off to address over-served customers like Hyperscale providers with a lower-cost business model
Time
Different measure of performance
New-market disruption: Compete against non-consumption
SDI addresses markets with focus on agility and elasticity and lower performance requirements enabling competing against non-consumption by Hyperscale Providers
Time
Non-consumers or Non-consuming occasions

Figure 31: Disruptive Innovation and SDI

The EEVs spent their resources trying to improve the performance of the existing technology, which was approaching its technology limit on the S-curve. The investments made by these incumbents were largely in the incremental or sustaining innovations category. On the other hand, new players (such as Cloud Providers) emerged who had a problem that needed to be solved. The products from EEVs, which were previously hired to perform similar jobs, weren't sufficient anymore due to a change in the performance and cost profile. The EEVs who were trying to listen to their customers did not view the Hyper Scale model and SDI as a major threat because their customers weren't thinking of adopting those technologies given its characteristics. However, as the technology started to evolve and become good enough, the EEVs' customers started seeing the business benefits of the technology. The study by Christensen and Bower (1996) (19) on the disk drive industry highlights this dilemma. Established firms fail to invest in radically different technologies or business models since the resource allocation is based on the needs of the existing conditions. This has been a classic case of new entrants disrupting incumbents. Over a period of time suppliers to a market place will invest more resources into the then dominant design. Their core competency becomes ever more closely aligned to the dominant design, these core competencies serve it well as long as the dominant design continues to dominate. However when a new design emerges that has sufficient improvements over the dominant design, these core competencies become core rigidities.

The EEVs are faced with the 'Innovator's Dilemma' with the advent of SDI and need to strategize in order to jump on to the S-curve for the next generation technology.

4.3 Diffusion of Innovation

The theory of Diffusion of Innovation seeks to explain the process, the reasons and the rate at which new ideas and technology spread through cultures. Everett Rogers, a professor of communication studies, popularized the theory in his book *Diffusion of Innovations* (20) first published in 1962. In the book, Rogers explains diffusion as the process by which an innovation is communicated through certain channels over time among the participants in a social system. The origins of the diffusion of innovations theory are varied and span multiple disciplines.

The process of diffusion over time is typically illustrated as a classical normal distribution or "Bell curve." Based on their inclination to adopt new technologies the adopters can be classified as (20).

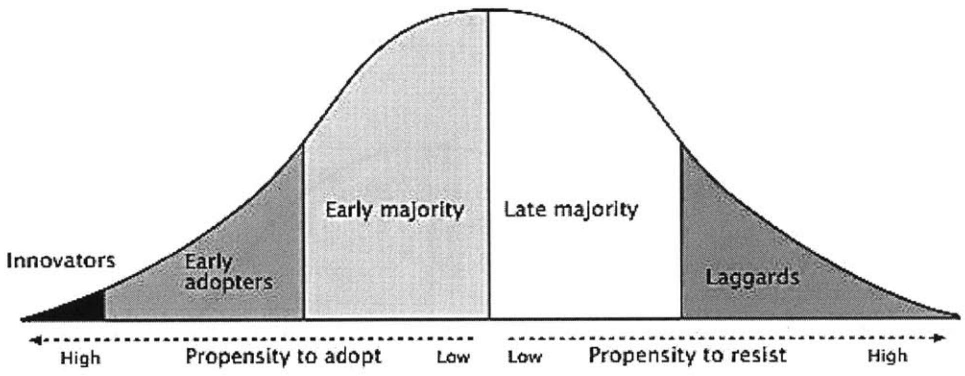


Figure 32: Everett Rogers Diffusion of innovation Model **(20)**

This model classifies adopters based on their willingness to use new technologies as well as their propensity for risk taking. The Innovators are the adopters who are at the bleeding edge of new technology and their high level of risk tolerance allows them to adopt technologies that may eventually fail. Early Adopters are the firms that are considered as thought leaders in their fields and are more deliberate in their adoption of new technologies but are more willing than the rest of the population to adopt a technology that can deliver value to their business. The next group of adopters are the early majority who start adopting technology just as it is about to become mainstream and a dominant design has emerged. The later majority is the organizations who would likely wait for the 2.0 version of the technology to come out so that all the flaws and shortcomings of the 1.0 version have been fixed by the time they adopt. Lastly there are the laggards who are extremely risk-averse and would likely skip a technology or product generation and are resistant to any change and upgrades.

Based on the Diffusion of Innovation framework described above, we classify SDI as still being in the "early adopters" phase according to Everett Moore's Technology adoption model. Web-scale companies widely deployed their versions of SDI because they needed to innovate in order to reduce their costs. These firms have been the innovators adopting this technology in its early stages.

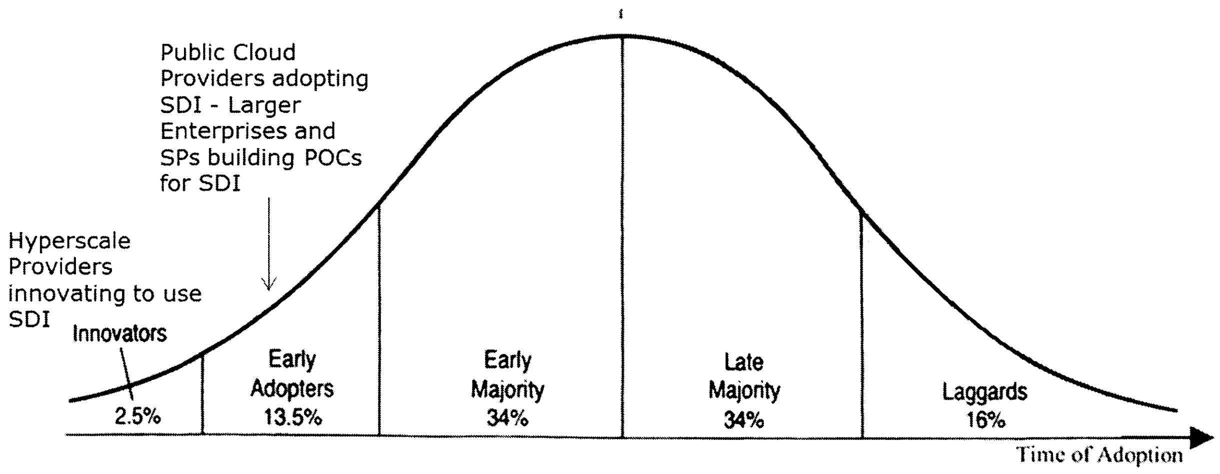

Public Cloud Providers adopting SDI - Larger Enterprises and SPs building POCs for SDI
Hyperscale Providers innovating to use SDI
Innovators
2.5%
Early Adopters
13.5%
Early Majority
34%
Late Majority
34%
Laggards
16%
Time of Adoption

Figure 33: Diffusion of Software Defined Infrastructure Technology

Everett Rogers identified five product-based factors that govern the rate of diffusion (20). These can be applied to predict the rate of diffusion for Software Defined Infrastructure.

- *Relative advantage*: The degree to which the product is better than the product it replaces: *HIGH (+).*

This is the degree to which an innovation is perceived as better than the idea it supersedes by a particular group of users, measured in terms that matter to those users, like economic advantage, social prestige, convenience, or satisfaction. The greater the perceived relative advantage of an innovation, the more rapid its rate of adoption is likely to be. There are no absolute rules for what constitutes "relative advantage"—it depends on the particular perceptions and needs of the user group. For the needs of the next generation of application, the Software Defined Products are significantly better in handling the requirements than the products they are displacing. Thus they have a significant relative advantage over the previous generation technology.

- *Compatibility:* Degree to which product is consistent with users' experiences: *HIGH (+)*.

This is the degree to which an innovation is perceived as being consistent with the values, past experiences, and needs of potential adopters. An idea that is incompatible with their values, norms or practices will not be adopted as rapidly as an innovation that is compatible. The most interesting aspect of Software Defined Products is their compatibility and consistency with users' experiences. Software Defined infrastructure elements (especially in the Storage domain) are often virtualized versions of the purpose built hardware components and hence provide consistent user experiences and workflows as the products customers are used to.

- *Complexity:* Degree to which a product is difficult to understand and use: *MED (+).*

This is the degree to which an innovation is perceived as difficult to understand and use. New ideas that are simpler to understand are adopted more rapidly than innovations that require the adopter to develop new skills and understandings. SDI as a product is fairly straightforward to understand since it leverages all the concepts of products from the previous generation. What is different with SDI is the dynamic spinning up and spinning down of resources. However since the basic concepts are still the same the products aren't difficult to understand or use.

- *Trial-ability: Degree to which an innovation can be tested or prototyped: HIGH (+).*

This is the degree to which an innovation can be experimented with on a limited basis. An innovation that is trial-able represents less risk to the individual who is considering it. Thus it is an important factor for any new technology. Due to the proliferation of Open source Cloud management Platforms like Openstack and Cloudstack the trial-ability of this technology high.

- *Observe-ability*: Degree to which an innovation can be observed in action: *HIGH (+).*

The easier it is for individuals to see the results of an innovation, the more likely they are to adopt it. Visible results lower uncertainty and also stimulate peer discussion of a new idea, as friends and neighbors of an adopter often request information about it. With Cloud service Providers and the benefits they offer with the Public Cloud the observe-ability characteristic of SDI is significantly higher.

4.4 Business Ecosystem

Business strategist James Moore pioneered the term "business ecosystem" and was central in developing an ecological approach to business and economic strategy. The concept of a business ecosystem is now widely adopted in the high tech community. The basic definition comes from Moore's book, *The Death of Competition: Leadership and Strategy in the Age of Business Ecosystems* (21).

Moore defines a Business Ecosystem as-

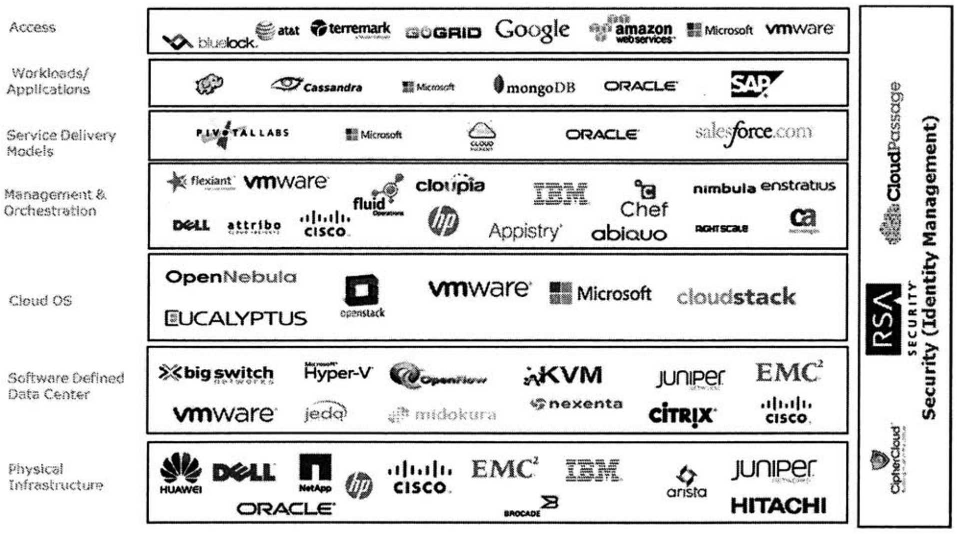
> "An economic community supported by a foundation of interacting organizations and individuals—the organisms of the business world. The economic community produces goods and services of value to customers, who are themselves members of the ecosystem. The member organisms also include suppliers, lead producers, competitors, and other stakeholders. Over time, they coevolve their capabilities and roles, and tend to align themselves with the directions set by one or more central companies. Those companies holding leadership roles may change over time, but the function of ecosystem leader is

valued by the community because it enables members to move toward shared visions to align their investments, and to find mutually supportive roles." (21)

The concept behind a Business ecosystem is the use of the term as an ecological metaphor to describe the coevolution of firms with others in the market and identifying the niche and key players in the ecosystem. As newly arriving species challenge niche players it becomes critical for the surviving species to be develop mutually beneficial symbiotic relationship with customers, suppliers, and even competitors.

Figure 34: Business Ecosystem – Software Defined Infrastructure captures the Business ecosystem for SDI.

The proliferation of Cloud and emergence of Software Defined Infrastructure is creating a rapidly evolving business ecosystem filled with diverse products, services, customers, competitors and collaborators. The profound ecological shifts are creating new partners and giving birth to new rivalries in this space. Disruptive trends from both the supply side (cost, complexity, competition, and commoditization) and the demand side (integration, bundling and content) are reshaping the players and relationships within this ecosystem.



Access
Workloads/ Applications
Service Delivery Models
Management & Orchestration
Cloud OS
Software Defined Data Center
Physical Infrastructure
Security (Identity Management)

Figure 34: Business Ecosystem – Software Defined Infrastructure

- *Key Niches:* The key niches (focused/specialists firms) within this business ecosystem include the Software Defined Networking, Software Defined Storage and the Management and Orchestration firms. Emerging SDI architectures like Hyper-converged infrastructure are disrupting the existing layers in this ecosystem. These are emerging technologies; which require specialist knowledge. As discussed by Christensen (18) the established players are organized to best serve the needs of their current customers. The real disruptive innovation comes from the niche players that are focused on a particular pain point in the industry. These players play an important role in technology innovation and adoption. The Software Defined Storage and Networking players are specialists in their domains in areas the established players have only recently started paying close attention too. Several of these players, especially those that bring complementarity assets like Management, Orchestration and automation, partner with the incumbents in a symbiotic relationship to offer a complete solution to their customers.

The size and scope of this market has offered other niches that have evolved in the overall business ecosystem including pure play software vendors that support processes around enabling effective operation of the data center or companies that bridge the gap across an on premise deployment to the off-premise Cloud deployment.

- *Key Players:* The key players in this space are the five main IT vendors – IBM, Cisco Systems, HP, EMC and Oracle, along with infrastructure Software vendors like Microsoft and VMware. These incumbent IT vendors address a broad market with IT solutions from compute, network to storage for both enterprise and service provider markets. The products used in this market require a high degree of technical sophistication as well as scalability and reliability. The key players typically dominate these segments of the market.

- *Leaders and Followers:* IBM, Microsoft, Cisco Systems, EMC, HP, Oracle have been the leading players in this business ecosystem over the last decade. The rest of the firms like Juniper, NetApp and Dell along with the international vendors like Huawei and ZTE play the role of followers competing either in niche markets or on cost by commoditizing the products in this space. The leaders rely on their significant R&D investments in order to develop innovative solutions to capture both mind and market share. They shape the architecture of the system and the development of new protocols.

  Leaders like Cisco and EMC have built platforms in the networking and storage spaces respectively that improve the overall system performance while providing a way to inter operate with both complimentors and competitors. This enables these firms to offer a highly integrated solution based on a modular architecture to their customers while maximizing the returns from in-house R&D activities by connecting them with external sources of knowledge.

The IT industry has undergone several waves of creative destruction where competence destroying, architectural and radical changes transformed the industry – right from the mainframes to the client server to the Cloud—during each of these waves the companies in the leadership position has changed.

To understand the impact of the leaders in this space *Figure 35: Key Players – SDI Business Ecosystem* shows the products that these companies have in the first layers of the technology stack. Companies like EMC, IBM, HP, and Cisco have a full complement of products that play across the entire stack. As firms adapt to emerging technologies it will be critical to offer a modular solution that can offer all the functionality needed to build a complete IT stack.

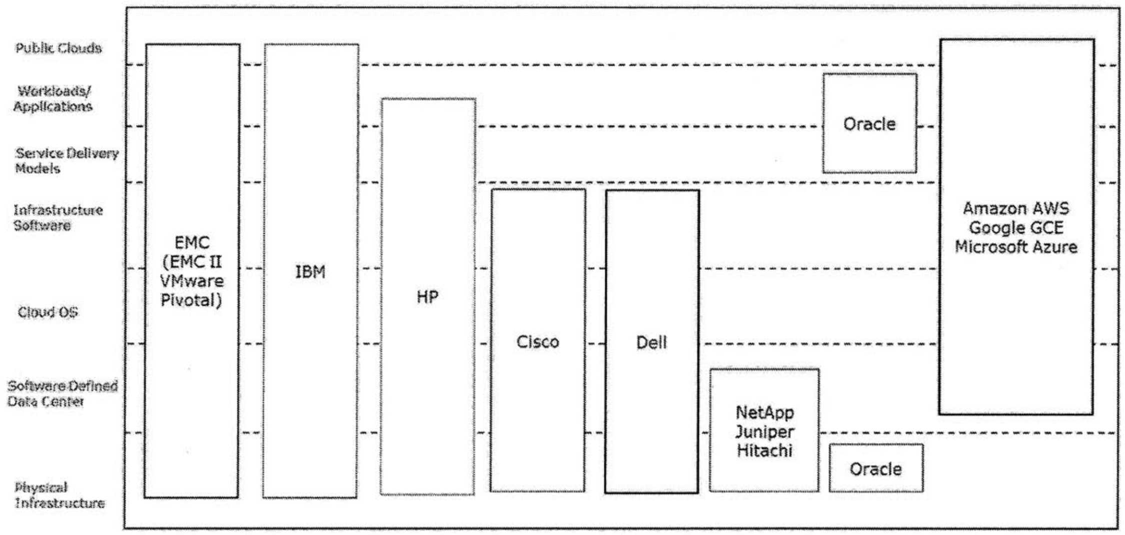


Figure 35: Key Players – SDI Business Ecosystem

4.5 Business Model

A business model can be simplistically defined as all the activities that a business needs to perform in order to make money. More formally – a business model describes the rationale of how an organization creates, delivers, and captures value in economic, social, and cultural or other contexts (22). The process of building a business model forms a core part of business strategy. The term is often used to describe a variety of aspects of a business ranging from its purpose, process, customers, product offerings, strategies and organizational structures as well as trading and operational policies. Business model innovation can help a company reinvent itself or, for a startup, disrupt a powerful incumbent.

The Software industry has seen a variety of business models. There is no 'one-size fits all' model for software businesses. Based on the product and the market, there are various different business models that have worked well.

- *Transaction Based* – This is the traditional license based model where a user pays for the right to use the license for a specific period of time. Examples include – Microsoft Windows, Office, and Paid Apps on Smartphones, Enterprise Software like VMware, SAP, and Oracle.
- *Advertising Based* – In the advertising model, often the Software itself is available for use free of charge however it is support by advertisements. Users are shown ads that are contextual and are usually displayed in a non-obtrusive manner. Privacy groups take a strong exception to this business model as it involves monitoring user behavior and content in order to display relevant ads. Due to similar concerns this business model is not used for Enterprise Software. The examples of advertising based business model include – Google (Gmail, Google Search, YouTube), Facebook.

- *Usage Based* – Usage based models have become extremely attractive and widespread with the advent of the Cloud. This model involves charging a fee for the period of time or computing or storage capacity used by the user. Examples include – AWS EC2, Microsoft Azure, SaaS Vendors like Salesforce, Cloud Storage companies like Dropbox, Microsoft, Google all use a usage-based model.

- *Service Based* – There are two variants to this business model. In one model, Software service and maintenance contracts are sold in addition to the licensed software as an additional revenue stream for the company. In the other business model, the software is offered for free but the service component is the one that is monetized. Most Enterprise software falls in this category. Service is either bundled or more likely sold separately with the product or in case of Open Source Software (a model popularized by RedHat, SuSe and other commercial Linux distributions) the Software can be used for free while the customer pays for the support of the product.
- *Appliance Based* - Appliance-based model is a hybrid Software hardware business model, where a company bundles software with a hardware appliance to sell a system. Additional software suites can be purchased. An appliance-based model heavily relies on monetizing the hardware than the software purely. EEVs like Cisco, IBM, HP, Dell, EMC mainly use the appliance-based model to sell their Software.
- *Freemium/Shareware* – A freemium service model is in which the Software with all or limited functionality is offered for free either perpetually or for a specific period of time. To use additional or premium features the customer needs to pay extra.

There are various strategic management frameworks and tools that exist in order to describe a firm's business model. The Business Model Canvas is a strategic management template for developing new or documenting existing business models. It is a visual chart with elements describing a firm's value proposition, infrastructure, customers, and finance (22). It helps firms in aligning their activities by illustrating potential trade-offs.

The figure below uses the Business Model Canvas as a way to describe the business model of an equipment manufacturer. The various component of the BMC are as described below –

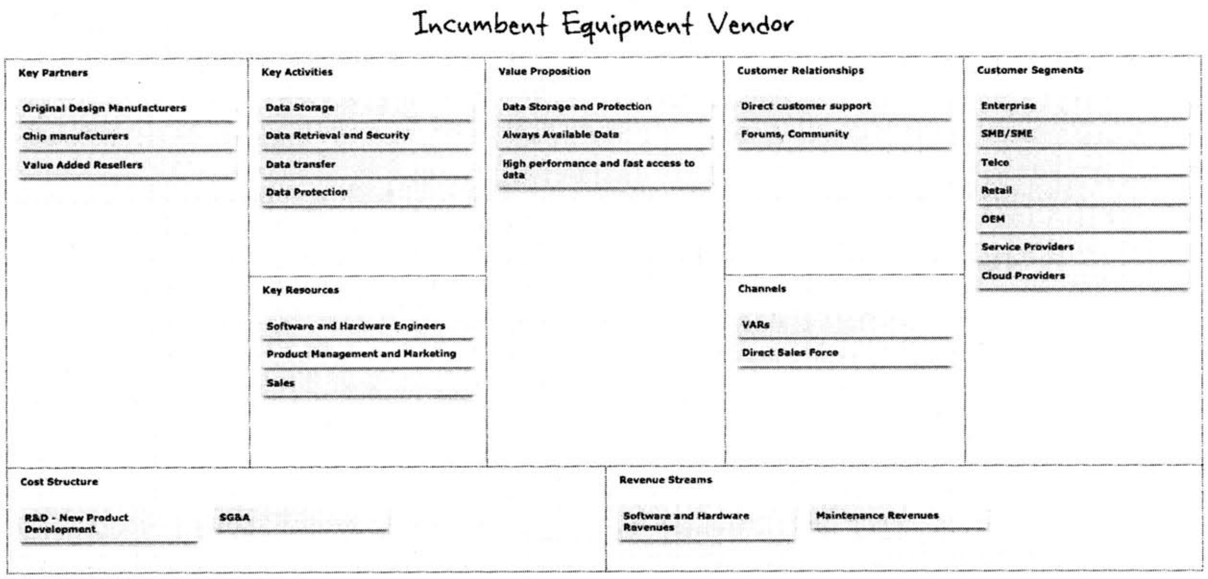


Figure 36: Business Model Canvas – Incumbent Equipment Vendor (Data Storage) [1]

- *Key Activities:* This captures the most important activities in executing a company's value proposition. For example for a Data Storage company – the key activities are mainly to create products that can securely, reliably store and protect data. Similarly for a network switch manufacturer; the key activities would be to securely transfer data from one end point to another.

[1] Generated using canvanizer.com

- *Key Resources:* These are the resources that are necessary to create value for the customer. These are the core assets of the company that are necessary to sustain and support the business. These resources could be human, financial, physical and intellectual.
- *Key Partners:* These are the key partners that an organization cultivates in order to optimize operations and reduce risks of a business model and focus on the core activities. These include component makers, ODMs as well as relationships with other vendors to resell or OEM their product.
- *Value Proposition:* The value proposition is the unique characteristics of a company's product or service offerings that distinguish itself from its competitors'. The value proposition provides value through various elements such as newness, performance, customization, "getting the job done", design, brand/status, price, cost reduction, risk reduction, accessibility, and convenience/usability.
- *Customer Segments:* To build an effective business model, a company must identify which customers it tries to serve. Various sets of customers can be segmented based on the different needs and attributes to ensure appropriate implementation of corporate strategy meets the characteristics of selected group of clients. The different types of customer segments relevant to equipment vendors include: Enterprise, SMB/SME, Mid-Market, Service Providers and Verticals like – Telco, OEM, and Federal.
- *Channels:* A company can deliver its value proposition to its targeted customers through different channels. Effective channels will distribute a company's value proposition in ways that are fast, efficient and cost effective. An organization can reach its clients either through its own channels (store front), partner channels (major distributors or value added resellers), or a combination of both.

- *Cost Structure:* This describes where a company's cost center lies and how its costs are divided.
- *Revenue Streams:* This is the revenue model, which describes how a company will earn revenue, and what type of revenue generation model it will adopt.

4.6 Impact on Existing Business Models

The adoption of Cloud, the hyper scale IT model and the emergence of Software Defined Infrastructure will have a significant impact on the business model of traditional IT vendors. Given the disruptive nature of SDI from both a technology and business perspective, it is clear that EEVs need to think innovatively about their business model. However, it is not easy for established companies to undertake business-model innovation. It is easier to create new products that disrupt competitors without fundamentally changing their own business model but changing and disrupting their own business model is very difficult to do while maintaining current business operations.

There are, however, technology and business discontinuities when adopting a new business model becomes necessary to survive in the long term. Research (23) identified strategic circumstances that often require business model change. In order to identify if a change in their business model is needed, we overlay the impact of SDI -

- Disruptive Innovation - Opportunity to address through disruptive innovation the needs of large groups of potential customers who are shut out of a market entirely because existing solutions are too expensive or complicated for them. This holds true for SDI. One of the reason Hyper Scale and Cloud Providers' have bypassed the EEVs to go directly to the ODMs is the high cost of the existing solution especially at the scale that CSPs need. SDI gives EEVs an opportunity to succeed in a market segment where they have traditionally struggled to offer a competitive solution.

- Emerging Technologies - Opportunity to capitalize on a brand-new technology by wrapping a new business model around it or the opportunity to leverage a tested technology by bringing it to a whole new market. SDI offers EEVs an opportunity to capitalize on the new Software Defined technology and adopt a new business model in order to address the next generation architectures.
- Defensive posture - The need to fend off low-end disrupters. SDI is also a defensive strategy for EEVs as they try to protect their turf from the effects of commoditization and being taken over by the ODMs.
- Competitive Pressure – Firms need to respond to a shifting basis of competition. Inevitably, what defines an acceptable solution in a market will change over time, leading core market segments to commoditize.

The IT vendors' business model is an appliance business model. These vendors, whether they handle networking switches and routers or storage arrays, sell an appliance that is a combination of purpose built hardware tightly coupled with software designed to run on that proprietary hardware and monetized via two sources of revenue - appliance revenue and maintenance revenue.

- *Appliance Revenue* – Revenue generated by selling the appliance. The appliance pricing is mainly the hardware costs plus margins along with a charge for the software. The Software component usually forms a smaller portion of the total unit revenues. Usually especially towards the end of a financial quarter or year the sales force will throw in additional software for free as long as they can make their quota for that period in terms of the total revenues from the appliance sales. This model works well when customers are looking for a tightly integrated appliance.

- *Maintenance Revenue* – The other significant revenue stream for IT vendors is the maintenance revenue stream where customers can sign up for service and support for a specific period of times like three or five years.

The components of the Business model that are significantly affected by SDI include –

- *Value Proposition* – The value proposition that EEVs offer their customers now changes from delivering a tightly coupled system to delivering software that can be used to provide the same level of functionality but leveraging commodity hardware. This is a big technical and business challenge for EEVs.
- *Key Partners* – There will need to be greater amount of collaboration across the different SDI platforms with Cloud Service Providers as also the component manufactures and ODMs.
- *Revenue Streams and Cost Structure* – The shift to software only model from an appliance model will have a significant impact on both the top and the bottom line of EEMs. That in turn will also have an impact of the cost structure and the go to market strategies of these firms.

The impact of SDI on the business model of EEVs is captured below

Characteristics	Impact of SDI
Modularity	Low -> High
Interoperability	Low -> High
Extensibility	Low -> High
Go To Market	High-touch -> Low-touch
Customer Base	Enterprise -> Enterprise, SPs, CSPs
Gross Margins	High -> Very High
Average revenue per unit	High -> Low

Figure 37: Impact of SDI on Business Model

SDI is forcing vendors to rethink the appliance model since they may not be selling an appliance anymore. This transition to software based business models from a hardware based business model will have a significant impact on vendor cash flow and top-line revenue growth. The high average selling prices that vendors are used to with appliances will not be available for the software model. This results in pressure on a company's top line revenue numbers. For Public companies, this may result in a decline in market value. Based on the software-licensing model chosen (up-front licensing versus subscription based) it may further impact the market value until the periodic revenue streams are stabilized and assured. On the other hand, by entering a software-only business model, companies can significantly improve their gross margins, since the marginal cost of producing a copy of the software is almost zero.

The challenge for companies in this new structure is to transition this shift in revenues from a high selling price appliance to a lower selling price but extremely high margin software business model. These business model changes will have an organizational wide impact across the various different departments in the company. With the up-front appliance revenue eliminated, vendors may not have access to the level of cash flow needed to invest in next generation products. Similarly, the sales organization will now need to be compensated accordingly. Most sales teams are compensated on a percent of their sales or meeting revenue targets for a specific time period. Their variable compensation is often tied to the total revenues that they bring in. With the lower selling price of a software only solution compared to an appliance it will significantly reduce their compensation and motivation of selling the product, hence the sales force needs to be compensated accordingly

Thus SDI can have the potential to impact EEVs along a couple of different dimensions.

- Deflation of Hardware Revenue – Significant impact to the top line
- Value migration to software only vendors on new architectures – erosion of customer base.

4.6 Value Chain and Value Capture

According to (24) "competitive advantage stems from two distinct, albeit related, activities; value creation and value capture". The value chain was first described by Michael Porter (25)—according to Porter; the value chain is a "system of independent activities, which are connected by linkages. Linkages exist if the way, in which one activity is performed, affects the cost or effectiveness of other activities". Linkages describe the relation between the different activities and illustrate the impact of a particular activity on others and thus serving an important source of competitive advantage and value adding. Products and services should pass through all activities of the value chain sequentially and, at each activity, the service gains some value and the chain of activities gives the product more added value than the sum of all single activities (25).

Value creation, involves performing activities that increase the value of a company's offering and encourage customer willingness to pay, is the heart of any business model. In traditional product companies, creating value meant identifying enduring customer needs and manufacturing well-engineered solutions.

The value chain for EEVs is as shown in the figure value chain diagram below. The various players in the value chain for data center equipment include –

- *Component Manufacturers* - These include firms that build the basic components that go into building a data center component. This includes Microprocessors, Memory, and Network Cards, Storage devices including disk drives.

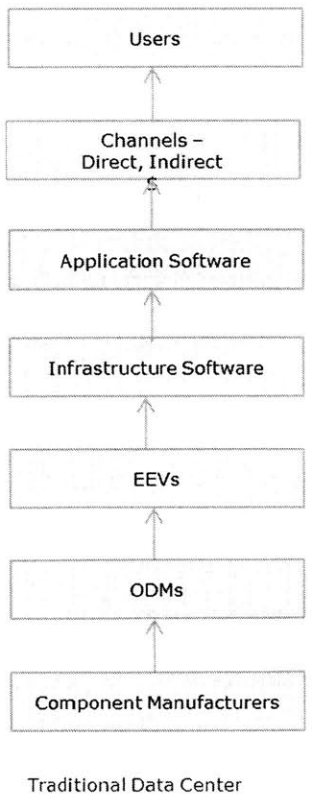


Figure 38: Value Chain for Traditional Data Center

- *ODMs* – Original Design Manufactures are firm that designs and manufacture a product as specified. Another firm for sale under a different brand name eventually rebrands their products. Such companies allow the branded firm or OEM to produce without having to engage in the organization or running of a factory.
- *Equipment Vendors (EV)* – Equipment Vendors are firms that consume or sometimes manufacture their own products and tightly couple it with software to give the hardware a specific device personality. These products are tightly integrated to offer the highest performance, reliability and resiliency.
- *Infrastructure Software* – Infrastructure Software vendors provide the Software management layers that can manage the infrastructure components of the system. This is one of the areas where the Cloud has brought in a lot of innovation. Before the advent of the Cloud, the Infrastructure Software was mainly used to manage the life cycle of compute resources (VMware vCenter, Microsoft System Center). The agility and flexibility that the Cloud provides makes this IS layer extremely critical. This is the layer in a Software Defined Infrastructure that can not only manage the virtual infrastructure but also automate and orchestrate their deployment and thereby providing Cloud-like agility and flexibility.
- Application Software – Application Software includes the vendors who build applications for the end user. Applications can range from payment processing, human resources, and analytics to business intelligence. Application Software vendors need to adapt their software to the new characteristics of the infrastructure.
- *Users* – The users here include the consumers of the infrastructure – Enterprises, Service Providers, Cloud Providers and End Users.

The value chain for hyper scale providers/Cloud vendors is as shown below. It is a much smaller value chain since it eliminates the EEVs from the value chain in favor of directly going to the ODMs to source their hardware and investing in-house resources to build the Software for the management and the functioning of the hardware.

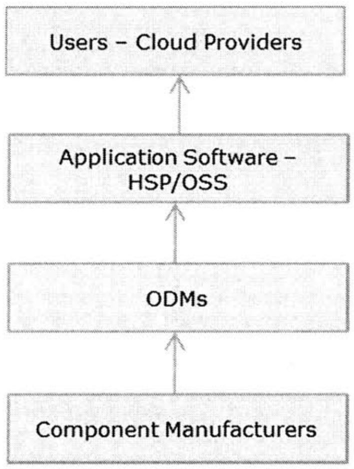


Figure 39: Value Chain – Hyper scale Providers

The SDI value chain is really a combination of the hyper scale and traditional IT value chain. The EEVs are trying to insert themselves back in the value chain by trying to add value to the bare metal products of the ODMs and providing a services layers on top to facilitate seamless deployment and management for web scale IT environments.

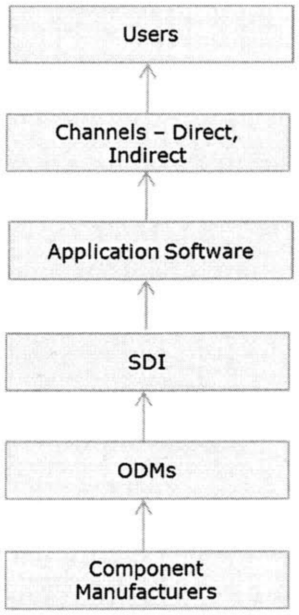


Figure 40: Value Chain – Software Defined Infrastructure

Figure 38: Value Chain for Traditional Data Center - shows the value capture in the value chain. As seen in the diagram the Equipment vendors are able to capture a significant portion of the value.

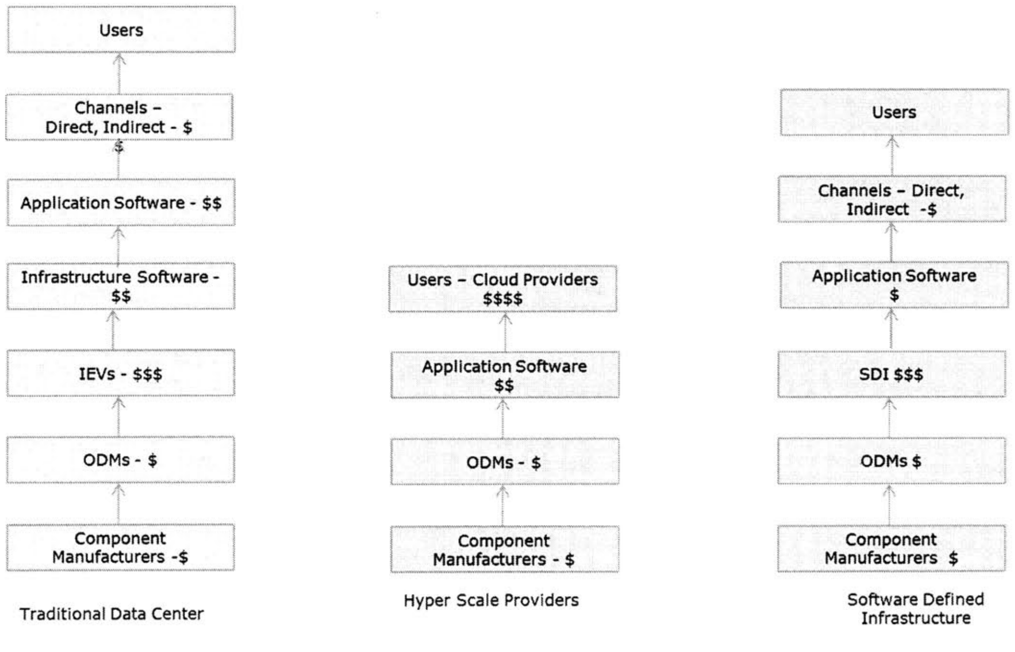

Users
Channels - Direct, Indirect - $
$
Application Software - $$
Infrastructure Software - $$
IEVs - $$$
ODMs - $
Component Manufacturers -$
Traditional Data Center
Users - Cloud Providers $$$$
Application Software $$
ODMs - $
Component Manufacturers - $
Hyper Scale Providers
Users
Channels - Direct, Indirect -$
Application Software $
SDI $$$
ODMs $
Component Manufacturers $
Software Defined Infrastructure

Figure 41: Value Capture – Traditional DC, Hyper Scale Providers, SDI

Figure 41: Value Capture – Traditional DC, Hyper Scale Providers, SDI - shows the value capture for Cloud Service Providers (CSPs). The Cloud Providers like Amazon AWS, Google, Facebook and Microsoft bypass the equipment vendors to directly go to the ODMs to source their hardware. For the CSPs every penny saved can amount to hundreds of millions of dollars in savings at the scale they operate in. So it is important for them to squeeze as much value as possible out of the hardware and the dedicated software stack that is optimized for their environment. Hence they bypass EEVs to directly go to the ODMs.

4.7 Double Helix Framework for Enterprise Infrastructure Value Chain

In order to further analyze the value chain and predict the evolution of the EEV industry dynamics we apply the Double Helix Framework to the Enterprise infrastructure value chain discussed in the previous section. According to Fine's model the periods of vertical and horizontal integration within an industry alternate based on the clock speed of the industry (26).

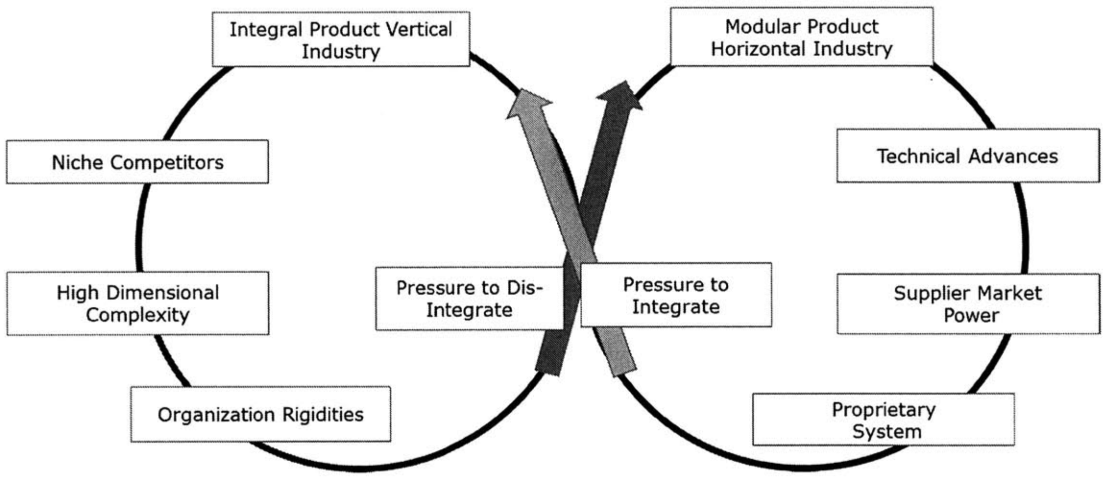

Integral Product Vertical Industry
Modular Product Horizontal Industry
Niche Competitors
Technical Advances
High Dimensional Complexity
Pressure to Dis-Integrate
Pressure to Integrate
Supplier Market Power
Organization Rigidities
Proprietary System

Figure 42: Double Helix Industry Structure

The Double Helix analysis (26) offers an excellent framework to analyze potential shifts as it emphasizes the technologies and critical capabilities can get commoditized over time based on the clock speed of the various components. This can lead to organization rigidities that put pressure on the integrated product and spur a shift to the other half of the double helix. The vertical industry and an integral product leads to leads to lower competitive intensity as only a few players can afford to build fully integrated systems. This gives rise to niche competition. This is similar to the Disruptive innovation that occurs at the low end of the market. As the niche competitors take hold, the high dimensional complexity of the large players gives rise to organizational rigidities, which in turn puts pressure on those firms to dis-aggregate. Following the double helix to the modular product and a horizontal industry phase – technological innovation and advances as well as the market power of suppliers put a strain on the profitability of the modular system and increasing the pressure to integrate and for the industry to transform to the vertical model.

This cycle of vertical to horizontal industry transformation repeats at intervals that are unique to each industry based on the clock speed of the various components of the system. The clock speed of the industry depends on the product clock speed, the process clock speed. The hypothesis (26) states that the industry clock speed that a firm faces increases the farther downstream it is located in the supply chain. Thus, enterprise equipment vendors experience faster clock speeds and consequently shorter product cycles than semiconductor manufacturers. Rapid technological innovation accelerates clock speeds while system complexity and organizational rigidities slow down the clock speed.

The Enterprise computing industry of the 80s and 90s was a highly integrated vertical industry. There were few players that built the mainframe computer that provided an integrated server and storage to the computing environment. The big giant of that era – IBM – had several forces working against it including constant pressure from niche entrants, particularly in software like Microsoft. There were competitors who innovated in niche markets like Intel with the microprocessor. As IBM grew successful and became bigger both from a business perspective as well as the employee headcount perspective it became more bureaucratic and thus losing the agility needed to battle these niche nimble competitors. At the same time, niche competitors changed the game by investing the Client-Server computing model, which completely decimated IBM's vertically, integrated model and have birth to modular computing systems and new competitors in servers, networking and storage. This pushed the industry from the vertical integrated loop into the horizontal loop. As Client-Server computing took off the modularity of the Enterprise computing elements led to significant technological advances. The three critical elements of the IT infrastructure – compute, network and storage each evolved long different technology trajectories getting to a point where there is significant pressure to integrate back into a vertically integrated system. The supplier market power with Hyper scale providers directly going to the ODMs and bypassing the EEVs is putting additional pressure on the firms in this space to provide an integrated product that can deliver significantly more value across all the layers of the value chain than the bare bones equipment provided by their suppliers.

5. Strategic Recommendations

The IT infrastructure landscape has undergone a massive change over the last several years, due to both technological innovation and business needs and drivers of next generation of computing applications. This change is impacting the entire IT stack as well as the ecosystem and value chain built around it. Chapters 1-4 of this thesis focused on this transformation. Based on that technology and business analysis this chapter focuses on the recommendations for firms to be successful in this new era of hyper-scale computing.

5.1 Market Opportunity

The market research firm International Data Corporation (IDC) coined the term 3rd Platform is order to describe these disruptive trends in the Technology industry. The 3rd Platform consists of technologies that are delivered through ubiquitous access through Cloud and Mobile, featuring massive data stores or lakes gathering information from the Internet of Things and that enable Big Data analytics to converge faster and facilitate informed decision making by the businesses. The basic building block of this 3rd Platform is the agile and flexible infrastructure described in chapters 2 and 3, which enables higher-level applications to deliver value to the end user. This is already resulting in a shift in the buying patterns of customers in terms of the infrastructure elements they are looking to purchase for their next generation of data centers. This shift in the buying patterns for infrastructure is disrupting the traditional server, storage and networking system value chain. Customers are moving away from buying systems to buying components or building blocks to build their own data centers. This is resulting in a change in the value chain as several large customers are now bypassing the traditional vendors (EEVs) to go directly to original design manufacturers to get their hardware right from the source and then building software organically on top of it. On the Software side the rise of OSS (Open Source Software) is putting similar pressure on Enterprise Software and infrastructure companies.

IDC predicts that this shift that already started in 2014 is expected to accelerate in the coming years. While the overall IT infrastructure spending grew by 5% in 2015 only 0.7% was a result of the traditional infrastructure while 15% was a result of the high growth 3rd Platform markets. By 2020, IDC expects about 50% of IT spending and over 100% of IT growth will be driven by 3rd Platform technologies (IDC - State of the Market: IT Spending Review & Forecast Q4 2014). More significantly for incumbent vendors the 3rd Platform is expected to cannibalize the 2nd Platform growth, which will soon start to shrink as more investment dollars shift to the 3rd Platform. The cannibalization effect isn't merely a replacement of 2nd Platform gear by the Software Defined Infrastructure but it will also give rise new solutions made possible because of the disruptive technologies of the 3rd Platform. This will result in value shifting up the technology stack from the infrastructure layer to the application layer and to players that can deliver an integrated platform.

We saw that from the analysis of the Double Helix framework that the industry movement now is towards offering well-integrated systems that can deliver a turnkey solution.

5.2 Customer Adoption

We are witnessing the transformation of Enterprise computing infrastructure by several high profile customers who are going down the path of building their own infrastructure based on the hyper scale computing approach.

Automobile manufacturer General Motors recently announced the building of a $130 million facility that can hold 20,000 square feet of commodity servers to run their private cloud using Hadoop on x86 general-purpose hardware. In order to do that they brought their outsourced IT environments in house and invested resources – both people and equipment – in order to build a Facebook-like infrastructure (27).

Similarly, Wal-Mart, via its Wal-Mart Labs technology division, adopted Openstack because of the flexibility and adaptability of the Software to fit its specifications without vendor lock-in. That infrastructure was the one that the big box retailer ran its Cyber Monday campaign in 2014 on (28).

The move to generate efficiencies like the web scale players has also spread to the financial services industry. For example - Goldman Sachs is also in the process of making moves to a "containerized" data center. These data centers called pods or micro DCs (Micro Data Centers) makes it easier to build infrastructure using building blocks that can be stacked incrementally in order to provide on demand resources (29).

Similarly, Bank of America has been implementing plans to move to cheap commodity hardware like Facebook and avoid lock in and compatibility issues of traditional IT equipment. The bank hopes to get on the innovation curve of the web giants in order to achieve its business objectives.

By 2018, Bank of America plans to have 80% of its workloads running on software-defined infrastructure.

These examples represent the growing influence and adoption of hyper scale computing in mainstream Enterprise environments. This puts significant pressure on EEVs as these customers represent some of their best high paying and loyal customers. The move by the financial services companies is extremely significant because these firms were the ones who wouldn't have migrated completely to a Public Cloud environment due to compliance, security and regulatory reasons. However, the approach they have taken is that instead of moving to the Cloud they would be replicating the characteristics of the Cloud in their internal IT environments. Their goal is to create a shared IT infrastructure that can be accessed on demand instead of statically allocating resources to each individual department. The 'Facebook like' infrastructure model will also provide the ability to repurpose hardware and save costs instead of buying new equipment. Bank of America, for example, hopes to save significant capital and operational expenses similar to Facebook's experience with their Prineville data center. This change is projected to save then $1.2 billion over the next three years by using cheaper hardware, as well as intelligent home-grown software that can cut down the human intervention for equipment management significantly. The Software Defined characteristics of the infrastructure enables Facebook to have one administrator manage more than 20000 servers whereas the average IT admin manages about 300 servers, which is an order of magnitude difference (30).

This transition is significant since IDC predicts (IDCs - WW Hyper Scale Data Center Census and Server Inventory 2014-2018 report) that in 2015 hyper scale data centers will be just 1.1 % of all data centers constructed with greater than 20,000 square feet but will account for 40% of the installed base of servers in the high-end data centers. The analyst forecasts the sales of servers by ODMs directly to customers will account for 16% of global x86 server shipments by 2018, amounting to $4.6bn in revenue. It is evident that ODM companies are rapidly changing their business model in order to directly target hyper-scale data center customers, says the report.

These examples clearly demonstrate the value that customers see in the Software Defined Infrastructure model. The key value proposition for customers includes

- Agility
- Flexibility, Adaptability
- Lower Costs (Capital and Operating) and
- Avoiding vendor lock-in.

As Enterprise equipment vendors strategize about their future direction they need to keep these key tenets in mind and make product and business decisions accordingly.

5.3 Competitive Analysis

The evolution of the Competitive Landscape and the value capture over the different computing paradigms is captured in Figure 43. The traditional IT environment was ruled by the EEVs and they captured a lion's share of the value that they generated. ODMs formed a small portion of the value capture and were mainly hidden from the end customers because of the EEVs who took the ODM equipment and added value via software and services and delivered a product for the Enterprise customer. Because of the hardware-defined nature of the infrastructure the ODMs were essentially developing purpose-built products for their customers (EEVs) and there was no way for them to move up the value chain without developing the software and system expertise the EEVs possess.

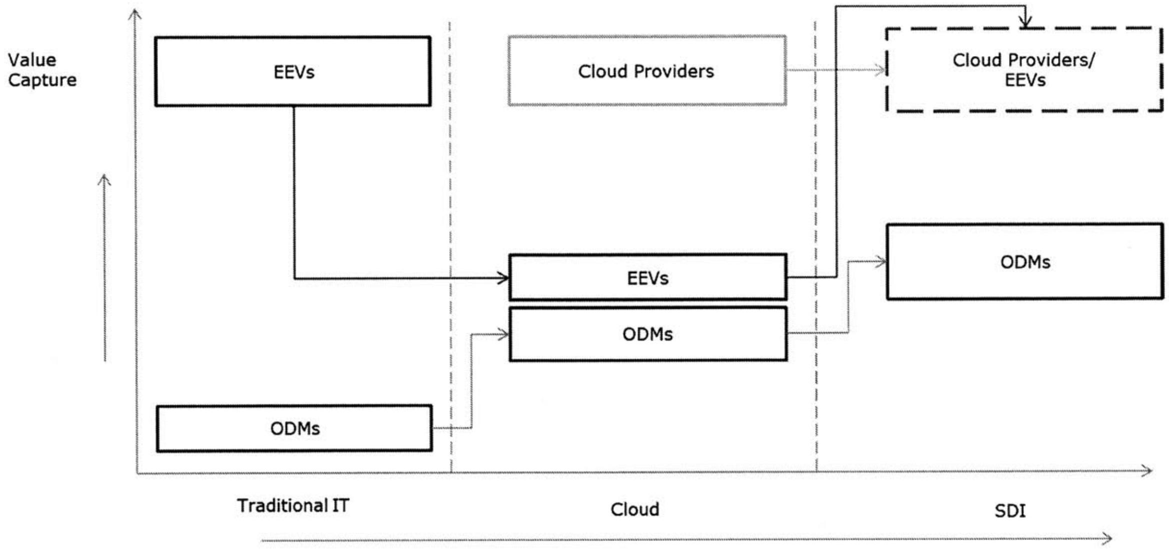


Figure 43: Competitive Landscape

With the advent of the Cloud, the value chain changed significantly, as now the Cloud providers captured much of the value derived from the infrastructure as well as the 'as-a-service' offerings that they offered to their customers. The move by Cloud providers to commodity software defined infrastructures meant that the role of the EEVs and their value proposition was significantly reduced. On the other hand, ODMs by directly selling to the hyper scale cloud providers, were able to increase their presence in such deployments. As we move to the era of Software Defined Infrastructure there are opportunities for Cloud and EEVs to generate and capture value in this new deployment architecture. Both of them understand where their customers are moving to and where the money is. The question is—would they be able to transform their product and business strategies to be successful in this space or would this give rise to new entrants or leaders to emerge.

From a Competitive perspective there are several threats that directly impact the EEVs and their business. On one hand is the threat from Cloud Providers like Amazon Web Services, Microsoft and Google who offer Enterprises the ability to rent out infrastructure as we discussed in chapter 1.

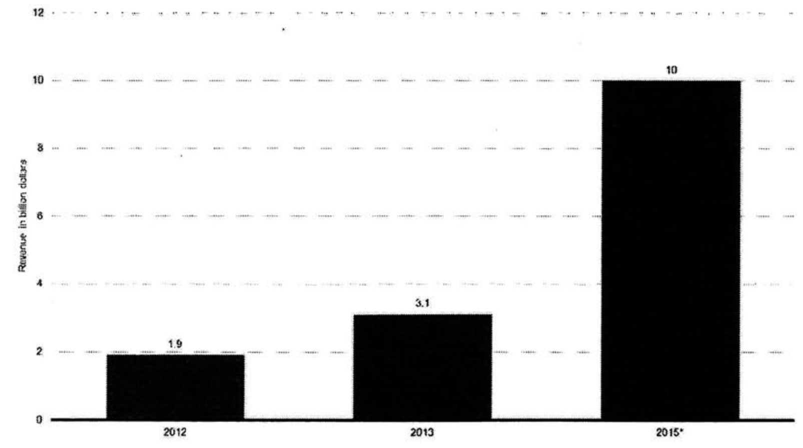


Figure 44: Forecasted AWS Revenues

The revenue growth of AWS (Figure 44 - Source: Business Insider; Various sources (Pacific Quest; ID 250520: Graph: Statistica) in a lot of ways is a direct erosion of the EEVs revenues as AWS and other Cloud Providers offer a substitute to the products and services offered by the traditional vendors. The Public Cloud providers are trying to attract the Enterprise customers by making it easier to connect to their infrastructure. Products and tools like Storage gateways, Direct Connectors and end-to-end VPNs offer means for the on premise customers of traditional IT equipment to migrate data and applications into the Cloud. Cloud Providers are quickly building solutions and products that can handle all the computing needs of Enterprise IT. The Cloud Providers build their own infrastructure using the Software Defined Principles and to further entrench themselves in the Enterprise they could start packaging their Hyperscale infrastructure as an integrated product to appeal to the on premise hyper scale deployments. Amazon is already well underway to do this as part of their Cloud for the US government where they would be essentially building an Amazon Cloud that's isolated from the rest of the world and in complete control of the government. This deployment could be expanded to other Enterprises to handle Hyperscale in the Enterprise and thereby directly impacting the EEVs.

As discussed in Chapter 4 – ODMs are a direct threat to EEV's in the hyper scale Computing world and for the building of Software Defined Infrastructures. ODM and ODM Direct – which is how IDC classifies the scale of ODM equipment to Hyperscale providers has seen a significant rise in demand over the last five years.

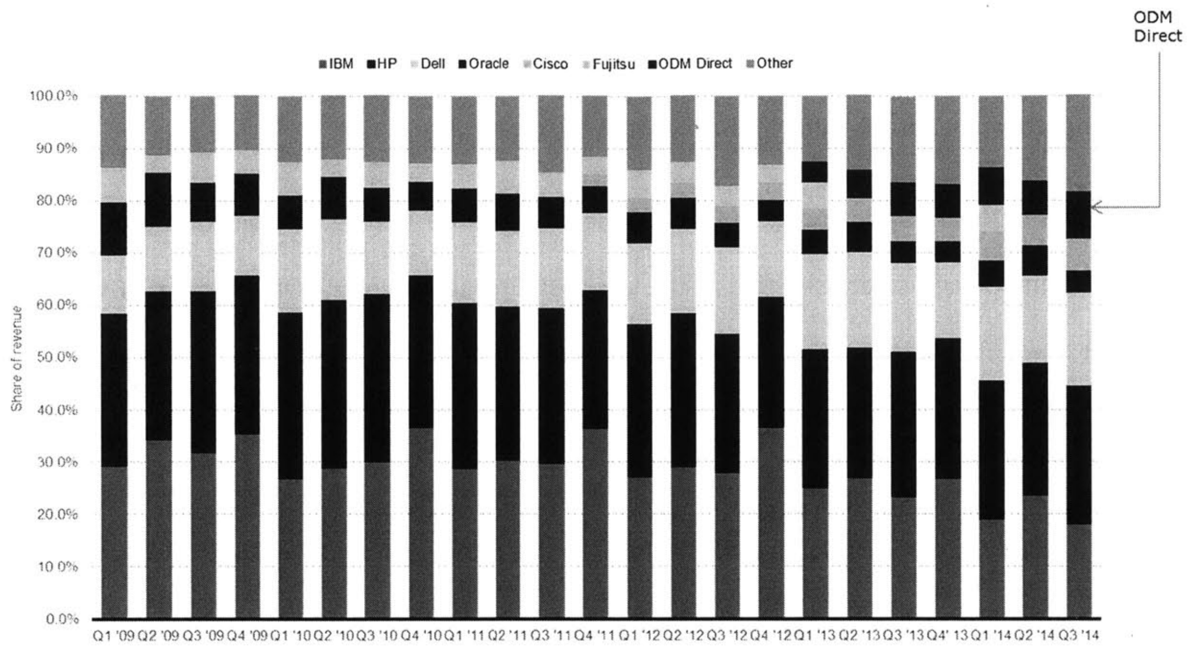


Figure 45: Server Vendor Global market share 2009-2014

As shown in the chart in Figure **44** (Source: IDC; ID 269396 – Graph: Statistica) ODM Direct sales have risen from almost zero to 10% of the total market in a span of seven quarters. That is a stunning growth rate and one that is expected to continue as more Enterprises start looking at deploying Software Defined Infrastructures.

5.4 Product and Business Strategy

Based on the technology trends, business drivers and market opportunity, we recommend that EEVs look into evolving their product and business strategy through the following ways -

5.4.1 Product Design and Vertical Integration

To compete in the hyper-scale space with ODMs, the EEVs need to optimize product design and their go-to market strategies. As discussed in Chapter 4 in the context of the Double Helix Framework – the IT industry is moving to a phase of vertical integration. The firms that are able to provide a vertically integrated but at the same item an open product are the ones that would be able to generate and capture value in this new era of computing.

EEVs have a significant opportunity in front of them to go after the weaknesses of the ODMs while playing to their strengths. As attractive as the low cost alternatives of the ODMs seem, EEVs need to remember that not all customers have the scale requirements of the hyper scale providers – Google, Facebook, Amazon and Microsoft - and neither can all companies afford investment in research and development to build software to provide the software defined experience that they desire. An ODM solution has hidden costs associated with it including building software defined management and orchestration layer in addition to providing all the components of a software defined data center – software defined compute, network and storage. This is a daunting challenge for a majority of the Enterprises. The cost of integration of the white box hardware, the support burden and managing the software and their technology trajectory as well as debugging and troubleshooting are both beyond the scope of most customers but also beyond the core competencies of the ODMs. This is where EEVs can truly differentiate and create value by offering solutions that can be turn-key while providing the kind of service and support experience most Enterprises are used to. In order to be successful in era of hyper-scale computing EEVs must adopt Software Defined strategies in order to leverage commodity components. The key to enabling SDI is the system infrastructure or the management software that glues all the pieces together. Open source and community led initiative like Cloud-stack and Openstack are gaining momentum and have become the de facto standard for the management and orchestration layer in the SDI.

SDI can be viewed as the Operating System of this new architecture that manages, schedules, executes and regulates the behavior of the system. As discussed previously, SDI is still in the early stage of adoption. The disaggregated Hyper-scale architecture discussed in chapter 4 is starting to emerge as the dominant design for the next generation of data centers with some key players coalescing around it. There are products and technologies that are starting to emerge as the core components of the SDI Platforms. Several EEVs are part of the Openstack movement and actively contribute code to the community or have their own Openstack distributions. Openstack is similar to Linux in its evolution, it is open source but each EEV will have the opportunity to customize it and monetize it in different ways. Multi Cloud and Inter Cloud working is the key to the success of an SDI platform. Companies like EMC, Cisco Systems and HP have all recently bought companies that have their own Cloud management Platforms or build their own Openstack distributions in order to control and influence this core component of SDI.

To succeed in their efforts EEVs will need to continue building and expanding the core set of features. They cannot afford to make their software distributions entirely proprietary. It will need to be extensible so that partners and complimentors can build on their platform. EEVs like Cisco and EMC have traditionally been the drivers of innovation in their industries. In the era of SDI, where the industry is moving back to a vertically integrated model no single firm can deliver all the capabilities required by the market. They need to work collaboratively with others to create initial applications and with complimentors to create applications that can leverage the interfaces of the platform. Similarly for the infrastructure software vendors like VMware and Microsoft, it is critical to work with EEVs like Cisco and EMC in order to build their software components or the Cloud Management Platforms that can interwork with various different infrastructures.

EEVs also need to be aware of the customer desire to avoid vendor lock in. The promise of SDI is to enable customers to be agnostic to the hardware underneath. As EEVs build their SDI Platforms, they must ensure that there is no lock-in to the components that they bring to market; otherwise, they can damage the customer perceptions regarding the openness and neutrality of their software platform. EEVs will have their own secret sauce so that their platform is well integrated and works the best with their products but they need to make sure they still interoperate and enable equivalent capabilities for competitive products too.

5.4.2 Customer Centricity

The EEVs must try to be laser focused on their customers and their needs for these next generations of computing environments. Understanding the jobs that their customers are hiring their products for will enable EEVs to evolve their products according to the needs of their customers. Customers buy from EEVs to get high quality products that are serviced and supported by the EEVs. One clear differentiator for EEVs is the service component, as mentioned before. EEVs must try to be the trusted partner for their customers as they make their transition from the hardware to software defined infrastructure. Focusing on customers also lets the EEVs leverage one of their most prized asset and that is the installed base of their gear. By leveraging their installed base of customers and offering those seamless data and application bursting capabilities to take advantage of the Public Cloud, EEVs can offer a compelling solution to fight the attack from the ODMs at the low end. This will also enable them to take an offensive posture with regards the transition of their workloads to the Public Cloud providers. By focusing on the helping their customers for both their traditional and emerging deployments, EEVs can serve as a bridge across the two architectures and be able to provide generate value for their customer. The on premise presence of EEVs is a huge advantage for EEVs as they can effectively server as a broker or the service layer to bridge the two architectures. Doing so would effectively enable EEVs to make the promise of Hybrid Cloud infrastructures a reality. The vision of the Hybrid Cloud is something both the Cloud providers and EEVs are gravitating towards. They realize that customers especially the large Enterprise customers who are interested in hyper scale and Software Defined Infrastructures have already made significant investments in the traditional computing architecture both from an infrastructure and applications perspective and aren't going to start over by deploying SDI. It is going to be an evolution where they gradually transition from the old to the new. That journey is the one which EEVs are uniquely positioned to exploit and create value. Enterprise customers are going to need trusted partners who can navigate the complexities of SDI while at the same time working with them to transition application and workloads to the new architecture. This is extremely important for EEVs as it plays to their strength of on premise computing and their huge customer base. By being a trusted partner in this journey, EEVs can enable their customers to transition to these new architectures while thwarting Cloud providers from making inroads into the on premise computing market. Customers who adopt SDI for an internal private cloud want to be able to easily move workloads between the public cloud and the private cloud. This interoperability is not an easy task. This is where EEVs can work with the various players in the Ecosystem to build applications and tools that make these movements seamless. Success in multi-cloud environments and the ability to make data portable across different Clouds is going to be the key to the success of a SDI Platform. None of the EEVs have a core competency in this area. This is where they can partner and or buy an emerging player—especially the ones who have been born in the Cloud, who have the necessary expertise to enable these workload movements to and from the Cloud.

5.4.3 Business Model Reinvention

The business model as discussed in chapter 4 is undergoing a transformation along two fronts. EEVs will need to redefine their business model in many cases and to do so in conjunction with a major industry transformation will be challenging. With SDI the EEVs have an opportunity to leverage an open source CMP to provide the interfaces that their partners and complimentors can use to develop applications and product using the SDI platforms. Software based business models as well as a Product and Service hybrid model will be the best way to capture value. There will be opportunity for EEVs to continue their existing hardware-centric business model because there will be a number of customers who would prefer a turnkey vertically integrated system with a 'single throat to choke' support so that they do not have to deal with the several vendors for each of the layers of their SDI. There is evidence of this today with the success of various converged infrastructures that EEVs offer, as well as the success of startups and established players in the emerging Hyper-converged deployments.

In addition to the shift from a hardware-centric business model to a software-centric one; there are also other changes that are implied by the Cloud and the Software model. The Cloud has accustomed customers to a consumption-based model where they pay only for what they consume. Similarly the SaaS (Software-as-a-service) business model also brings out subscription based licensing which is different than term licensing that EEVs are using. This will have a significant impact on the financial reporting model for the companies, as they can no longer recognize revenues right away after a sale, but will rather accrue over time via a subscription-based service.

Another side effect of this model is the go to market strategy for their products and services. The traditional go to market strategy and sales compensation is based on commissions derived from the total price of the equipment sold. With the move towards a consumption-based model the sales compensation process must also evolve to compensate and incentivize the sales force accordingly. These business challenges are as significant if not more as the product challenges facing the EEVs.

Based on our analysis these are some of the prescriptive recommendations for competing in the era of Hyper Scale Computing:

- <u>Build</u> a Software Defined Platform that can be adopted by both Enterprises and Public Hyper scale providers.

- Differentiate with the core aspects of the product like Software Defined Storage and Networking components in order to create a competitive advantage in the market place.
- Evaluate a services strategy by differentiating open source components based on services and seamless integration with the core products.
- Collaborate and form alliances with a coalition of partners and complementors for an ecosystem to rally undecided players or those not big enough to launch a Platform.
- Disaggregate the Software from the hardware for their existing appliances in order to leverage their core IP in the new Software world.

5.5 Conclusions

Through the technology and business analysis presented in this thesis, it is quite evident that the dominant design for the next generation of data center architecture is starting to coalesce around Software Defined Infrastructures. The ability to instantiate functions whether Compute, Storage or Network in software will revolutionize the processing capability paving the way for the Internet of Things. Hyper scale computing environments are a central part of this landscape.

For firms hoping to be successful in this space need a strategy on two fronts –

- A Software Defined Infrastructure that can enable infrastructure agility, flexibility and elasticity and one that enables seamless interworking across both private and public Cloud environments.
- Reinventing of the traditional equipment business model to monetize software and services based on commodity hardware. The business challenge for Cloud Providers is to be able to adapt their business model to sell infrastructure versus renting it out. Those capabilities are something they will need to develop in order to succeed in this space. On the other hand, the incumbent equipment vendors (EEVs) have the opposite problem. They know how to sell infrastructure; however, solving the technology challenge and then doing it in a reasonable time frame, such that the revenues for the software-defined infrastructure make up for the erosion of their core business will be the key.

This brings out an interesting dynamic, and levels the playing field for the different competitors in this space. For Cloud Providers who have the Public Cloud market figured out, it means packaging their infrastructure and offering it to customers in a private setting, thereby providing that single Cloud that can be available both in a the Private or a Public Cloud deployment. The key for EEVs is to make their SDI solution the Operating System or the Platform for the hyper-scale data center. By adopting such a Strategy, EEV's will not only be able to create value for the Enterprise customers, but will also capture a significant portion of the value created. They will also be able to enter new markets (hyper–scale data centers) by offering a compelling value proposition to the Cloud Providers to adopt their Platform in order to offer a seamless and unified experience for their customers.

Bibliography

1. *"Shadow IT - Should CIOs take umbrage?" . CXO Unplugged.*

2. **Andreessen, Marc.** Why Software Is Eating The World. *[http://online.wsj.com/articles/SB10001424053111903480904576512250915629460].* [Online] WSJ, August 2011.

3. Nest . *https://nest.com/.* [Online]

4. **Wikipedia, Hyperscale -.** *http://en.wikipedia.org/wiki/Hyperscale.* [Online]

5. **2014, AWS RE:Invent.** ZDNet: AWS: With more than 1 million active customers, we're your stack. *http://www.zdnet.com/article/aws-with-more-than-1-million-active-customers-were-your-stack/.* [Online]

6. GigaOm - Google's Biggest Quarter ever for Data Center Spending. *https://gigaom.com/2015/02/04/google-had-its-biggest-quarter-ever-for-data-center-spending-again/.* [Online]

7. *The Google File System.* **Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung.** s.l. : Google.

8. **Engineering, Facebook.** https://www.facebook.com/notes/facebook-engineering/under-the-hood-scheduling-mapreduce-jobs-more-efficiently-with-corona/10151142560538920. [Online]

9. **Newsroom, Facebook.** http://newsroom.fb.com/company-info/. [Online]

10. *Finding a needle in Haystack: Facebook's photo storage.* **Doug Beaver, Sanjeev Kumar, Harry C. Li, Jason Sobel, Peter Vajgel,.**

11. **12K, System Architecture - Cisco.** Network World. [Online] http://www.networkworld.com/article/2283224/lan-wan/chapter-1--internet-protocol-operations-fundamentals.html.

12. [http://www.vmware.com/software-defined-datacenter/compute]. . *VMware* . [Online]

13. **Research, Gartner.** *An Overview of OpenStack.* 2014.

14. The Register. *http://www.theregister.co.uk/2013/03/18/servers_pets_or_cattle_cern/.* [Online]

15. Ensure Cloud Application Resilience NetFlix Way. *http://www.cio.co.nz/article/466719/ensure_cloud_application_resilience_netflix_way/.* [Online]

16. *Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms.* **Clark, Rebecca Henderson and Kim.** s.l. : Administrative Science Quarterly, Volume 35, No. (March 1990).

17. **Christensen, Clayton.** [Online] http://www.claytonchristensen.com/key-concepts/#sthash.jHx9CyM8.dpuf.

18. **Christensen, Clayton M. (1997),.** *The Innovator's Dilemma: when new technologies cause great firms to fail.* Boston, Massachusetts, USA:, : Harvard Business School Press. ISBN 978-0-87584-585-2.

19. *Disruptive Technologies: Catching the Wave.* **Bower, J. L. and C. M. Christensen (1995).** s.l. : Harvard Business Review.

20. **Rogers, Everett.** *Diffusion of Innovations.* s.l. : Simon and Schuster, August 2003. ISBN 978-0-7432-5823-4..

21. **Moore, James F.** *The Death of Competition: Leadership & Strategy in the Age of Business Ecosystems.* New York : HarperBusiness. ISBN 0-88730-850-3.

22. *A. Osterwalder, Yves Pigneur, Alan Smith.* **Generation, Business Model.** 2010.

23. *Reinventing Your Business Mode.* **l Mark W. Johnson, Clayton M. Christensen, Henning Kagermann.**

24. *'Opportunity Discovery, Problem Solving and the Entrepreneurial Theory of the Firm .* **Hsieh, C, J. A. Nickerson, and T. R. Zenger.** s.l. : Journal of Management Studies , 2007.

25. **Porter, Michael E.** *Competitive Advantage: Creating and Sustaining Superior Performance.* New York : Simon and Schuster, 1985.

26. *CLOCKSPEED-BASED STRATEGIES FOR SUPPLY CHAIN DESIGN.* **Fine, Charles.** s.l. : Production and Operations Management, Fall 2000.

27. GM Data Center. *http://siliconangle.com/blog/2013/05/15/gms-130m-data-center-takes-after-facebook-runs-hadoop/.* [Online]

28. WalMart chooses OpenStack for Walmart Global Ecommerce. *http://www.walmartlabs.com/2015/02/17/why-we-chose-openstack-for-walmart-global-ecommerce.* [Online]

29. http://www.wired.com/2012/10/goldman-sachs-as-google/. *What do Google and Goldman Sachs have in common? More than you think.* [Online]

30. Bank of America's Data initiative Follows internet Companies into the Cloud. *WSJ Website - http://www.wsj.com/articles/bank-of-america-follows-internet-companies-into-the-cloud-1425676214.* [Online]

31. *WikiBon Study - Converged Infrastructure .*

32. **Christensen, Clayton M. (2003).** *The Innovator's Solution : Creating and Sustaining successful growth.* Boston, Massachusetts : Harvard Business Press. ISBN 978-1-57851-852-4..

33. *Technological regimes and Schumpeterian patterns of innovation.* **Breschi, S., Malerba, F. & Orsenigo, L.** s.l. : The Economic Journal.

34. *Technological discontinuities and dominant designs: A cyclical model of technological change.* **Anderson, P. and Tushman, M.** s.l. : Administrative Science Quarterly , 1990.

35. Software-Defined Networking: The New Norm for Networks. *White paper. Open Networking Foundation.* 2012, April .

36. *Michael Porter - Competitive Strategy.*

37. *Mapping the winds of creative destruction.* **Abernathy, W. and K. B. Clark.** 1985.

38. **Innovation, Mastering the Dynamics of.** *James Utterback* . s.l. : Harvard Business School Pres, 1994 . ISBN 0-87584-342-5.

39. **Utterback, James M. and Linsu Kim.** *Invasion of Stable Business by Radical Innovations.* 1985.

40. *IDC's Worldwide Software-Defined Storage Taxonomy, 2014.*

41. *IDC Predictions 2014: Battles for Dominance — and Survival — on the 3rd Platform.* December, 2013.

42. *Google Data Centers.* **Furrier, John.**

43. *Fusion-io Launched New Hyperscale Flash Product to Scale Data Centers for Enterprises, Not Just Facebook.* **Rouse, Margaret.** s.l. : Silicon Angle.

44. *Finding the Right Job for Your Product.* **Clayton Christensen, Scott Anthony, Gerald Berstell and Denise Nitterhouse.** s.l. : Sloan Management Review, Volume 48, Number 3, pages 38-47 (2007.

45. *Exploring the Limits of the Technology S-Curve Part II: Architectural Technologies.* **Christensen, Clayton.** s.l. : Product and Operations Management, Volume I, Number 4, pages 358-366 (Fall 1992).

46. **Christensen, Clayton.** *Exploring the Limits of the Technology S-Curve Part II: Architectural Technologies.* s.l. : Product and Operations Management.

47. **Utterback, J. M. and F. F. Suarez.** *Dominant designs and the survival of firms.* 1995.

48. **Kirtley, Fernando F. Suarez and Jacqueline.** *Dethroning an Established Platform.*

49. Definition: Hyperscale Computing. *TechTarget.* [Online]

50. **CloudBook.** Cloud Workload Types. *www.cloudbook.net.* [Online]

51. Cloud services and the new platform wars. *http://gigaom.com/2012/03/21/cloud-services-and-the-new-platform-wars/.* [Online]

52. *Capabilities, cognition and inertia: evidence from digital imaging.* **Gavetti, Mary Tripsas and Giovanni.** s.l. : Strategic Management Journal, Volume 21, Issue 10-11, pages 1147-1161 (2000).

53. **Suarez, F. F.** *Battles for technological dominance: an integrative framework.* 2004.

54. *A Hyper-Scale Cloud Data Center, Seen From the Clouds.* **Miller, Rich.** s.l. : Data Center Knowledge.

55. **James M. Utterback and William J. Abernathy.** *A Dynamic Model of Product and Process Innovation.* 1975.

56. *"Eager Sellers and Stony Buyers: understanding the Psychology of New Product Adoption",* . **Gourville, John.** s.l. : Harvard Business Review, Volume 84, Issue 6, pages 98-106 (June 2006).

57. *"Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms".* **Clark, Rebecca Henderson and Kim.** s.l. : Administrative Science Quarterly, Volume 35, 1990.

58. *[Three Elements of a Successful Platform Strategy.* **Choudary, Mark Bonchek and Sangeet Paul.** s.l. : HBR, 2013.

59. *"The NIST Definition of Cloud Computing" . National Institute of Standards and Technology. Retrieved 24 July 2011.* **National Institute of Standards and Technology.**

60. **Kincaid Jason, Anderson Chris.** *"Startup School: Wired Editor Freemium Business Models".* . s.l. : TechCrunch, Wried, 2012.

61. *Exploring the Limits of the Technology S-Curve Part I: Component Technologies.* **Christensen, Clayton.** s.l. : Product and Operations Management.

62. **HAMDAQA, Mohammad.** *Cloud Computing Uncovered: A Research Landscape.* s.l. : Elsevier Press.

63. **Diagram, Cisco 12000 System Architecture.** Network World. *http://www.networkworld.com/article/2283224/lan-wan/chapter-1--internet-protocol-operations-fundamentals.html.* [Online]