THE VULNERABILITY OF TECHNICAL SECRETS TO REVERSE ENGINEERING: IMPLICATIONS FOR COMPANY POLICY

By
Cenkhan Kodak

M.S. in Electrical and Computer Systems Engineering (2001)
University of Massachusetts at Amherst

Submitted to the Systems 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

FEBRUARY 2008

Signature of the Author: ______________________________

Systems Design and Management Program
January 2008

Certified by: ____________________

Professor Eric von Hippel
Thesis Supervisor, MIT Sloan School of Management

Certified by: ____________________

Pat Hale
Director, Systems Design and Management Program

THE VULNERABILITY OF TECHNICAL SECRETS TO REVERSE ENGINEERING: IMPLICATIONS FOR COMPANY POLICY

By

Cenkhan Kodak

Submitted to the Systems Design and Engineering Program
On February 04 2008, in Partial Fulfillment of the
Requirements for the Degree of Master of Science in
Engineering and Management

Abstract

In this thesis I will explore the controversial topic of reverse engineering, illustrating with case examples drawn from the data storage industry. I will explore intellectual property rights issues including, users' fair-use provisions that permit reverse engineering. I will also explore the nature of the practice via several types of analyses including: the costs and benefits of reverse engineering; the practical limitations of reverse engineering; and a layered approach to reverse engineering as it applies to complex systems. I will conclude with a discussion of innovation and its relationship to reverse engineering

Thesis Supervisor: Eric von Hippel
Title: Professor, Management of Innovation and Entrepreneurship,
Sloan School of Management, MIT

Acknowledgment

I would like to acknowledge and thank MIT Sloan School of Management Professor Eric von Hippel for guiding me throughout this effort. His book Democratization of Innovation and his lecture notes on user innovation inspired me to start this work. His quick and straight to the point comments helped me tremendously with the direction of my writing. It has been a great pleasure working with him.

I also like to thank my mother Nermin Kodak, my father Yüksel Kodak and my brother Volkan Kodak with all my heart. They were very patient with me when I was writing this thesis. They cooked for me, pampered me, gave me advice on what & how to write and stood by me when I was going through stressful times. Everything happened with their encouragement; even application to MIT for business education was their idea. What more could I ask for? I am grateful to my family.

I also like to recognize MIT Sloan School of Management Professor Charles Fine and Harvard Business School Professor Robert Huckman who introduced me to Operations Strategy.

Also I would like to thank Bob, Amanda and Ann from the MIT Online Writing and Communication Center. They reviewed my prose and provided much needed clarification, rephrasing and elimination of punctuation errors.

Finally, I would like to express gratitude to my boss Bill Landers, Regie Wolff and Joel Schwartz who have given me unwavering support while I was juggling my work and academic commitments, as well as to Pat Hale, our Director at the MIT Systems Design and Management Program and William Foley, our Academic Coordinator.

Contents

List of Figures

List of Tables

CHAPTER 1

Overview

Corporations traditionally have sought to protect their innovations in part by revealing as little as possible about their internal design - keeping important technologies as trade secrets hidden somewhere within their products. In this thesis, I will explain that reverse engineering is a powerful tool that can be deployed to uncover such trade secrets.

Competing manufacturers of similar products often know at least something about or similar to their rivals' trade secrets. I will argue that reverse engineering can effectively fill the gap between what the competitors already know and the full information regarding rivals' trade secrets.

Engineers seldom need complete blue prints or exact source code to decipher the key technologies in a rival's products. Simple inspection, rudimentary testing and careful reading of product documentation is generally enough to uncover a wealth of knowledge about these innovations. Engineers use these means to construct an initial framework of the hidden technologies. This framework partitions the complete system into its component subsystems, and identifies system/subsystem interfaces. Determining the input and output relationship at these interfaces is then used to characterize the system and the subsystem. Reverse engineering can do the rest of "fill-in-the-blanks" job to fully reveal the trade secret.

I will explain a tiered approach via which reverse engineering can fill in the blanks... How one moves from an initial whole system analysis, to the analysis of the important innovations hidden within deconstructs the reverse engineering challenge.

A brief overview of the contents of the thesis follows.

Reverse Engineering and introduction to Data Storage Industry (Chapter 2)

Chapter 2 presents various reverse engineering definitions and reconciles for the differences in the literature, identifies business reasons to engage in reverse engineering, briefly introduces modern controller based data storage systems and provides examples of reverse engineering in the industry.

Capturing Innovations in Data Storage Industry (Chapter 3)

Chapter 3 emphasizes the importance of preparing a business case for reverse engineering, makes an argument for a tiered reverse engineering frame work, introduces impulse response characterization for black box analysis to support the proposed framework and summarizes what can and can't be captured with the framework and comment on the time frame limitation.

The Reverse Engineering Framework and Cost Analysis (Chapter 4)

Chapter 4 will make an introduction to fractals and self-similarity and propose a layered approach based cost-accounting and then will discuss cost elements in a qualitative and quantitative manner for each layer of the reverse engineering process.

Benefits of Reverse Engineering (Chapter 5)

Chapter 5 will answer the return on investment question in a reverse engineering exercise. I will establish relationship between creativity, reverse engineering and innovation. Discuss strategic or marketplace benefits i.e. action, inaction and resource allocation. Account the value of reverse engineering for the society and finally underline the emergence of increased honesty and transparency for the buyers.

CHAPTER 2

Reverse Engineering and Introduction to Data Storage Industry

How can one write about reverse engineering practices in data storage industry and then propose a layered approach to reverse engineering complex systems without providing the necessary background material?

Chapter 2 is here to avert such a gap. It means to lay a ground work for the future chapters by investigating the following bullet items in detail.

- Present various definitions of Reverse Engineering and use cases associated with the practice. Reconcile for the differences in the definitions.

- Position of reverse engineering in business and reasons to engage in reverse engineering activities

- A brief introduction to enterprise class data storage systems and a quick discussion of architectural challenges industry faces in day to day business operations.

- Examples and history of reverse engineering in the data structure industry.

2.1 Defining Reverse Engineering: Taxonomy

A number of definitions for reverse engineering exist in the contemporary literature. At first glance, this diversity creates a sense of commotion and uncertainty in a researches mind but after some consideration it is easy to observe distinct flavors and subtle trademarks revealing the different professional origins of the writers of reverse engineering definitions.

For instance, with in the context of judiciary, a commonly referenced article in the Yale Journal of Law, (Samuelson and Scotchmer) suggest a broad and general definition as it applies to all products that can be purchased by means of an exchange. That is, "Reverse Engineering is a process of extracting know-how or knowledge from a human made artifact; hence it is an engineering effort to uncover the previous engineering".

As it seeks reversal of September 2004 Federal District Court decision in favor of Blizzard Software, IEEE-USA (Institute of Electrical and Electronics Engineers) sponsors a (friend-of-the-court brief) and offers science and high-technology focused definition of reverse engineering: "The discovery by the engineering techniques of the underlying ideas and principles that govern how a machine, computer program or other technological device works through analysis of its structure, function and operation". The same friend-of-the-court brief also affords a standard legal definition as it was spelled out in a Supreme Court case ruling over Kewanee Oil Co. v. Bicron Corp. 416 U.S. 470,476 (1974), "starting with the known product and working backwards to divine the process which aided in its development and manufacture". As it applies to software and complex systems the definition given by (Chikovsky and Cross) is particularly appealing "Reverse Engineering is the process of analyzing the subject system to create representation of the system at a higher level of abstraction".

This is a good time to define what I mean by complex systems with in the context of this thesis: Complex systems refer to the design and implementation of electromechanical, software and solid state devices that aggregates data storage systems. These systems provide on-demand access, secure connectivity and reliable storage capacity for digitized information.

The common theme present in all above definitions that differentiates Reverse Engineering apart from Reengineering[1] and Value Engineering[2] is that the former of the three is a process of inspection but not of modification. While Reengineering and Value engineering aggressively seek to enhance or modify a product with improvement in mind, reverse engineering leaves the product unmodified. On the contrary, maximum effort is spent to leave the subject of study intact and operational for capturing the essence of the innovation and core-competency. Reverse Engineering process seeks to capture value through a holistic examination of the system leading to a higher level representation document. At the end of the process no improvement is sought and no new version of the subject product is released to the market.

As Samuelson and Scotchmer point out, while Reverse Engineering has a long history as an accepted practice, there is hesitation and considerable debate about it over the last few decades. I have found that it is not hard to document well known examples of reverse engineering from military history. Most weapons of choice have been reverse engineered fairly quickly upon first introduction to the battlefield, since lives are at stake and no one can to afford yield any tactical advantage to the enemy.

[1] Reengineering as defined by Chikovsky and Cross is "the examination and alteration of the system to reconstitute it in the new form"

[2] Value Engineering is process of examination to increase the ratio of function to cost according to Lawrence D. Miles value engineering reference center. It is an approach to improve value in a product by reducing problems and/or costs while improving performance/quality requirements.

For instance, the siege device Trebuchet[3] is a scaled up improvement over common sling shot that has been reverse engineered over and over by different civilizations. Its existence was first reported around 4th century in China and it was reverse engineered independently in the Muslim World, Byzantine Empire and finally in Europe and Scandinavia and did not become obsolete well until the 16th century.

More recent examples of the reverse engineering practice, especially during the cold war ear are documented in detail and results have drastically affected the course of engagement between the super power foes. For example, Soviet Union's introduction of Mig-25 Interceptor led United States to start new and costly F-15 All Weather Tactical Superiority Fighter Program. It was a decade later and only after a defection incident, the US got a chance to reverse engineer a Mig-25 and was able to understand the true capabilities of the fighter as a standalone interceptor. Not an agile multi-role fighter as initially thought. That initial misunderstanding led to the hastened decision to start the costly F-15 program.

Reverse Engineering is a strategic tool that can affect the end result of multilateral engagements where actions of one party may affect consecutive decisions of others affecting the outcome of the game. It seems the practice will be fashionable for times to come when agents engage in strategic games choose to maximize their return, given the strategies the other agents choose.

[3] Trebuchet is an example of early attempt in artillery warfare. It was used as a siege device that utilizes counter weights and a sling-shot mechanism to throw projectiles over fortifications in the middle ages.

2.2 Classic View on Reverse Engineering

Just like rival nations who vie for superiority of capabilities and positions in various theaters, commercial enterprises have strategic reasons i.e. maximizing return to engage in reverse engineering activities regardless of their market position.

Market leaders and new entrants alike have been documented to reap commercial value out of well placed efforts to seek trade secrets, leading to further competitive advantage or a successful new entry into a new market. In this strategic environment, it is not a surprise then that the reverse engineering is a legal practice. Why because it serves the benefits of all stake holders at least from time to time. Even the end-users of the value chain reap benefits as the practice is perceived to commoditize the need in that particular market. In fact, the law on Reverse Engineering states that "acquisition of the known product is by fair and honest means, such as purchase of the item on the open market"[4] is the only precondition for legal reverse engineering. Further more, restatement of the law of unfair competition imposes enforcement limits for the owner of a trade secret, "The owner of a trade secret does not have an exclusive right to possession or use of the secret information. Protection is available only against wrongful acquisition, use or disclosure of the trade secret,"[5].

[4] Official Comment on Sec.1 of Uniform Trade Secrets Act. Taken from The Law & Economics of Reverse Engineering (Samuelson and Scotchmer)

[5] American Law Institute, Restatement of the law of unfair competition, comment at Sec.43 at 493(1993). Taken from The Law & Economics of Reverse Engineering (Samuelson and Scotchmer)

[6] IEEE defines Interoperability as the ability of two or more systems or components to exchange information and use the information exchanges.

Samuelson and Scotchmer claim that software industry has different reasons to engage in reverse engineering activities compared to other industrial sectors and they position interoperability at the top of their reasons list. Samuelson and Scotchmer state that Interoperability[6] has been the "most economically significant reason to reverse engineer in the software industry and it is also the most contested" i.e. relatively more litigations turn around interoperability reverse engineering then any other motives.

When the motivation for reverse engineering is attaining interoperability, the process is performed on computer hardware or software systems to capture the IP (Intellectual Property) necessary to make different but compatible and coherent devices or programs. Reverse engineering process generally targets API (Application Programming Interface) component to achieve system compatibility because platform developers may choose not to provide this interface package.

Depending on their perceived strategic position, API providers may either choose to provide the API under a license agreement for free or for a fee, meaning that the provider wants to take advantage of "network effects" [7] or sometimes in a bid to keep trade secrets secret and raise barriers to entry for protecting their market dominance, API providers may not provide APIs at all or string a number of restrictions to keep total control on their product. And in general the second choice prompts other application developers to go into reverse engineering as mentioned above.

[7] Network effect is the phenomenon whereby a product/service becomes more valuable as more people use it, thus encouraging ever-increasing numbers of users in an industry

[8] Platform is an architectural concept that represents a software and hardware framework that enables higher by running other software to run. These new software components could be new features or applications and they can be created by other parties.

Applications require communication with the underlying platform for executing through the APIs. Due to economic advantages of specialization, in many cases platform[8] developers and application developers are from different ways of life. And in this common case APIs not only constitute a layer between the systems, they also separate different financial interests from each other. Depending on the nature of relationships between these two interests, industry structure and balance of power, reverse engineering is employed and in some cases its legitimacy has been challenged in the court.

Samuelson and Scotchmer express surprise when they discuss the status of reverse engineering as a standard industry practice. They argue that underlying high costs and difficulties of reverse engineering software are significant barriers to overcome before engaging in the practice. Never the less, they still list other reasons aside from interoperability as possible motivations as below:

1 - Sometimes buyers of a particular software application will reverse engineer to fix the bugs or add new features before the vendor company manages to release a new version of the software package or a consecutive service pack. This is generally done due to unpredictable development cycles of software vendors and immediate, quantifiable business needs of the buyers. Note that many of the enterprise class buyers are B2B entities with significant resources and sophisticated software teams. If they think that an additional feature will benefit their business, they may not wait till a particular vendor comes up with an updated version. Many examples of such vertical integration attempts by lead users[9] exist.

[9] Lead user is a term developed by Eric Von Hippel , They are special set of users of a product that currently experience needs still unknown to the public and who also benefit greatly if they obtain a solution to these needs

2 – In a bid to extract maximum profit out of IP investments, industrial rivals tend to keep a close eye on new products introduced by competitors. Research costs are high and in order deter rivals from infringing on owned patents, reverse engineering is performed on the available products to detect any possible patent infringement violations. Any positive finding is a basis for initiating punitive lawsuits. These lawsuits could take up years but many companies have successfully used this lever to fend off competitors and discourage blatant "plagiarism".

3 – Documentation is a very important part of the software development process. Unfortunately, sometimes the documentation is lost or the developers do a poor job in documenting and commenting on algorithms or the implementers (individuals/functional group) leaves the company during or immediately after the project is done. In such cases, reverse engineering is initiated by a new team to gain full control of the product and make it viable again.

In addition to what is being listed above by (Samuelson and Scotchmer), there are other compelling reasons to reverse engineer products in the software industry. These are mostly strategic motives that are hard to quantify in terms of economic value but unavoidable with in the dynamics of competition.

4 – Industry Rivals may need to asses competitive positioning of available products in a particular market, and it is generally a good idea to resort to reverse engineering for accurately assessing the relative positioning of available products. First of all, every supplier needs to make strategic decisions regarding future product directions if they wish to stay in the business. Feedback from market and the state of competition is a paramount input for this decision. Redirecting scarce resource for a non-competitive product, ill conceived out of thin air would

be a grave business mistake no matter how technically innovative or brilliant the proposed idea behind the product is. In short one needs a solid business plan. To write this plan or fill a part of this gap in this plan, competitive intelligence is collected from the rivals. Purchasing competitive products and reverse engineering them is a solid method to determine the status of rival products and legitimate means to collect intelligence. It has to be noted that while there is ample information in the documentation about features and capabilities of a product, unless they are verified by engineering such claims and promises could only be rated as vaporware [10]. Marketing and sales machine of every company out there is known for sugar coating and to a considerable extent exaggerating capabilities of the products that they advocate. Distinguishing fact from fantasy calls for a commonsense, fact based approach: I believe reverse engineering is the only certain venue to understand what exactly is being offered by the rival.

No product manager could afford to make strategic product decisions based on sanitized information coming out of a competitor's website, there fore despite the costs, reverse engineering remains a well justified capital investment for unveiling the internal clock work of a competitive product.

5 - In B2B sales environment, companies are often involved in bake-offs leading to large amounts of sales. Bake-offs are special platforms or private functions where competing companies present their cases in front of the buyers. Here the company evangelists make special efforts to entice buyers towards the products offered by the business that they represent.

[10] This is a term emerged in 70s and 80s, refers to a software or a hardware product announcement that fells short of the feature set, delivery date and even the feasibility.

Sometimes, that is if the evangelist thinks that they can get away with it, glimpses of trade secrets are advertised in an exaggerated fashion for gaining tactical advantage over the rivals and tip the decision of the buyer in one's favor. One can prevent introduction of such uncertainty at such a critical time, but only through prep work done on competitors products i.e. reverse engineering. Well informed players in a bake-off environment will discredited any exaggeration on the spot. In fact, in many cases rivals make special efforts to make sure what they disclose is as close to the truth as possible, knowing that the competitor will be noting the claims and question the accuracy of the claim should there be any doubt.

6- Sometimes reverse engineering is done out of sheer curiosity, engineers are the type of people who like to know about how things operate. They have an intrinsic curiosity for finding out how things function. When a competitor builds a device, they will take it apart and try to figure out the conceptual framework behind it. This is only natural; Eric Von Hippel in his book Democratization of Innovation states that "Engineers seldom even want to see a solution exactly as their competition has designed: specific circumstances differ even among close competitors, and solutions must in any case be adapted to each adopter's precise circumstances.

What an engineer does want to extract from the work of others is the principles and general outline of possible improvement" and further states that "this information is likely to be available from many resources"[11]. We will argue here that one of these resources is the practice of reverse engineering and the motivation behind is that "others often know something close to your secret" [11].

[11] Democratization of Innovation, Von Hippel, Eric pp 80-82. Why users often freely reveal innovations.

Reverse engineers are intimately involved in current achievements relate to the industry. And they are either systems architects or people who are closely associated with them. They may be looking for inspiration for the next innovation. It is said that nature does an excellent job on inspiring innovators, yet it's not secret that competitors also provide distilled and quality material. They are mostly trying to achieve similar results in the market place but the internal context of their organization and motivations could be slightly different. Reverse engineers may want to take advantage of this contextual diversity. During the reverse engineering process they are able to look at the problem and the solution with a different perspective and they are ideally positioned to detect and fill the gaps –either left intentionally or unintentionally - in competitor's innovation. "Sometimes it is better to be the second to a market, as one will learn from the mistakes of the originator" as stated by Seymour Cray founder of Cray Research Inc.

2.3 Intro to Data Storage Systems

Customers of data storage industry are primarily large to midsized global enterprises with well defined and established system requirements. Before spending money and deciding where to store and how to scale up their existing storage capacity, they generally contemplate on a number of architectural attributes displayed in Figure 1. They are generally painfully aware that at the stake is the most prized informational assets of their company so they are conservative, diligent demanding and risk averse.

Reliability (or data redundancy) is the fault tolerance property of the data storage array. When a data storing component (disk, memory, data path) fails there is enough redundancy for recovery the lost data.

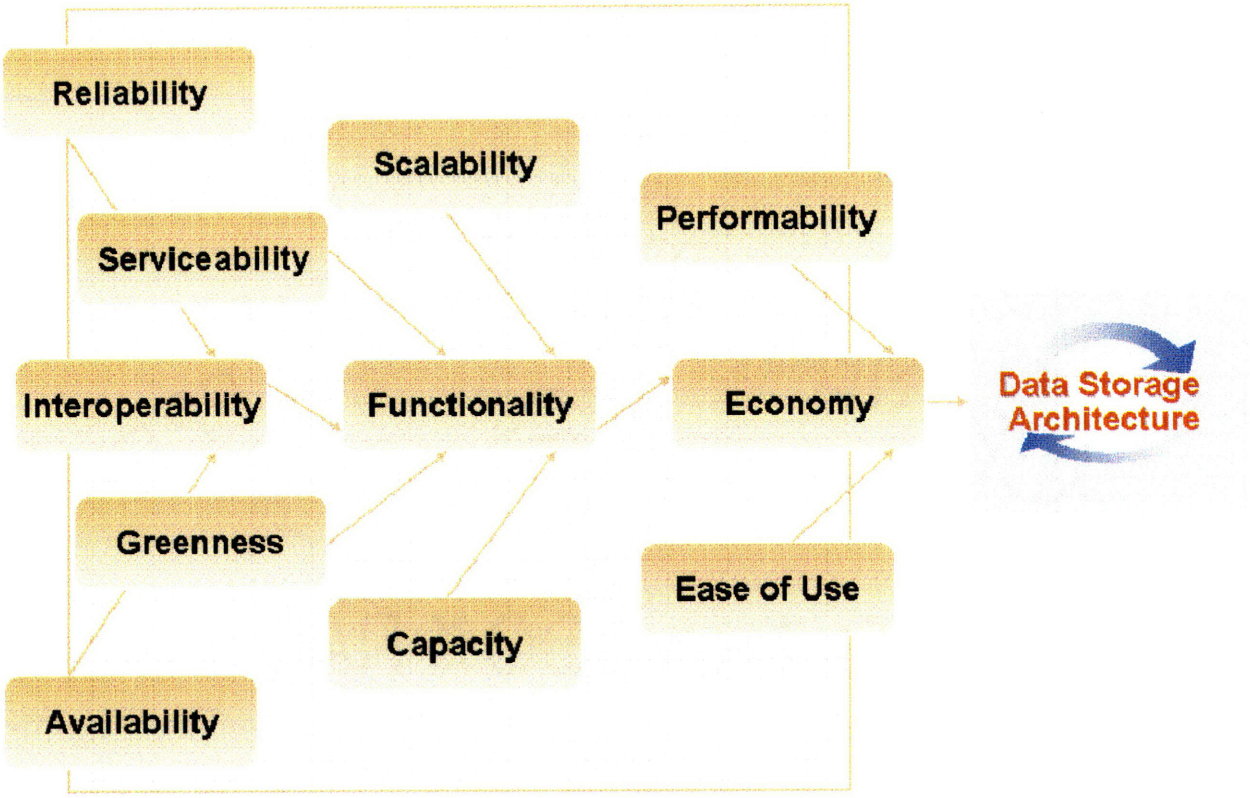


Figure 1: Architectural Principals behind Data Storage Systems

Serviceability is ability of service personnel to monitor components, identify faults, debug, perform triage and provide software and hardware maintenance in a bit to restore service.

Interoperability is ability of systems to operate and communicate in tandem and coherence.

Availability is the proportion of time the system is in function condition, including the degraded modes of operation.

Functionality is the ability to carry out multiple set of predefined tasks or functions in simultaneous or sequential manner.

Capacity: Amount of effective binary storage presented to host subsystem in raw file format.

Economy mainly refers to the expectation of the buyers to receive Moore's Law in terms of capacity and system response times with out paying extra.

Scalability is ability of the system to handle growing amounts of demand in a graceful manner.

Ease of Use is the ability of the system that a user can operate and control the system without having to overcome a steep learning curve

"Performability, with in the context of Data Storage is the ability to achieve high performance i.e. satisfactory response time sustainability, while guaranteeing disk subsystem[12] dependability."[13] Dependability is defined as "[...] the trustworthiness off a computer system which allows reliance to be justifiably placed on the service it delivers [...]"[14] in effect Performability is a composite attribute as described in Figure II.

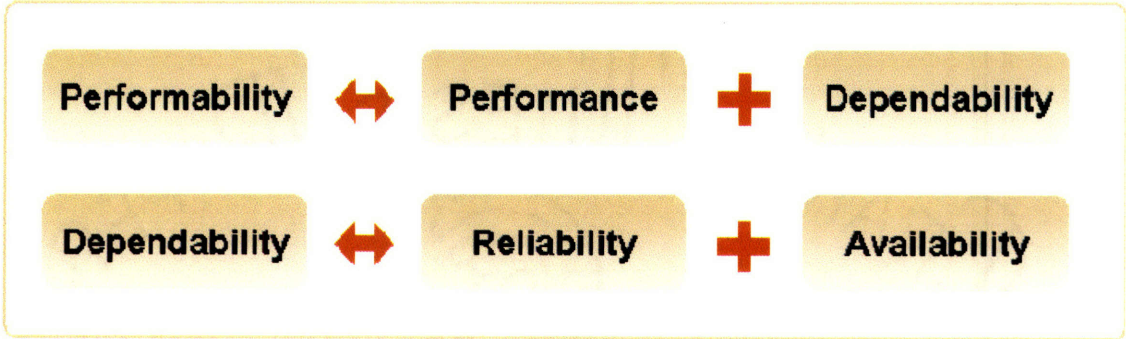


Figure 2: Visual Definition of Performability

The issue of power consumption (Greenness) is coming up as an agenda item in IT procurement bids. The demand for storage space is ever increasing and data centers are rapidly running out of cooling and power capacity. Industry pundits forecast that 50% of the data centers will suffer from insufficient power and cooling by 2008. To fill this gap, many vendors are engaged in a noteworthy effort to curb runaway power consumption. Some of the initiatives are:

- Efforts to increase data storage utilization by using novel techniques like Virtualization[15].
- Attempts to go for smaller form factors i.e. 2.5" drives in order to limit power consumption.
- Drive to avoid toxic components.
- Software features for utilizing systems resources in an adaptive-dynamic fashion.
- Bring in more recyclable components in to the manufacturing process.

[12] In the Raid Book which overviews the RAID Technology, writers of (RAB) Raid Advisory Board defines Disk Subsystem as a collection of disks and hardware required to connect them to one or more host computer. The hardware may include a controller, or the disks may be connected directly to a host computers I/O adapter. Chapter II Page 11 June 9, 1993. RAID (Redundant Array of Inexpensive Disks) is an umbrella term for computer data storage schemes that divide and replicate data among multiple hard disk drives.

[13] Product Data Sheet for Array Management Software OE (Operating Environment) June 8th 2007, Cenkhan Kodak, Senior Product Manager

[14] By IFIP 10.4 Working group on dependable computing and fault tolerance.

[15] Virtualization as defined by VMWARE Inc is the use of software (Operating System and Applications) such that the virtualization software transforming the hardware resources of the host computer including the CPU, RAM, Data Storage and the network – to create a fully functional virtual machine that can run its own operating system and applications like a real computer. http://www.vmware.com/virtualization

Vendors employ a complex combination of software/hardware systems components and optimizations for rising up to the challenges posed by Figure-I architectural principles. The point is to deliver competitive products bundled with customer focused features and functionality and then present value at a discount.

The list below articulates various systems implementation that constitute a modern storage array. Naturally, each of these pieces represents diverse areas of expertise. It is easy to see, from a system architecture point of view, that one of the challenges here is to seek ever efficient ways to promote interoperability and produce well balanced systems.

- Electromechanical components like disk drives
- Solid state devices like high speed memory and CPU.
- Interfaces devices that contain optical transducers that are capable of load balancing like Host Bus Adapters.
- Specially optimized algorithms that drive RAID technology.
- Software systems that constitute real-time Operating Environment.
- Special techniques and methodologies for delivering seamless, well balanced, dynamic systems.
- Third-party Operating systems like Windows and Linux.
- Layered application for data multiplication and disaster recovery
- Security enhancements, encryption algorithms.
- Management Software.

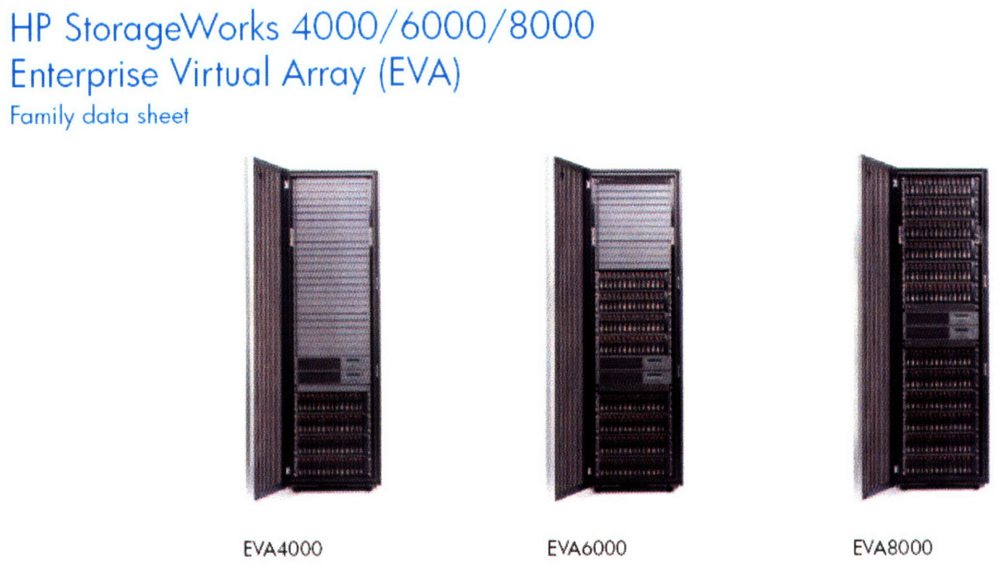
As the data storage industry is almost 100% Business to Business, I want to materialize the end-product for the general audience. Figure 3 and Figure 4 display a set of state-of-the-art data storage arrays. They designed by two leading of the leading mid-tiers vendors of the controller based data storage industry: HP Storage Works and EMC CLARiiON.

These Storage Arrays provide comprehensive storage solutions for the modern data center. They are highly scalable, capable of delivering short latency and sustained peace of mind for protecting and safe-guarding the information assets of an enterprise.

Associated Tables I and II display high level systems limits for helping customers with their different architectural requirements

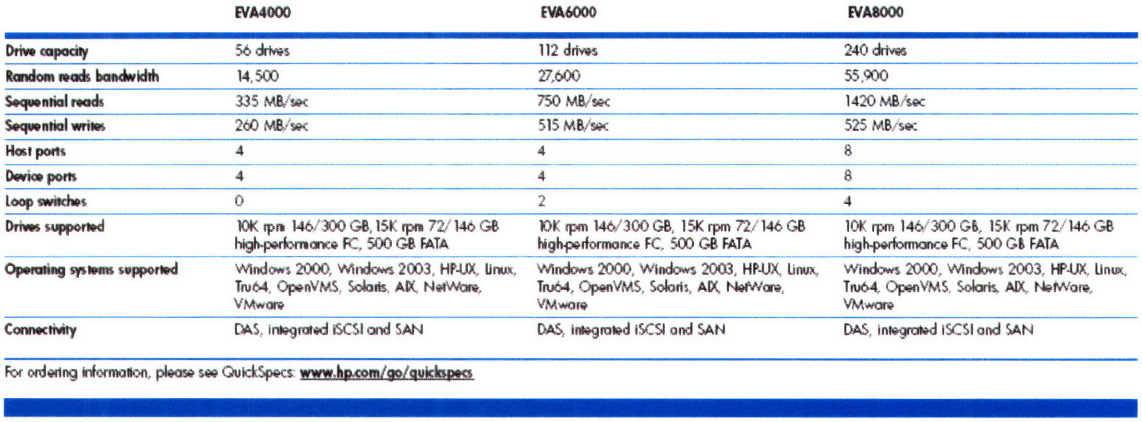
In addition, Figure V displays high level block-diagram of a typical data storage array and Figure VI shows the logical diagram of an older version EVA 5000 controller manufactured by HP Storage Works.

Technical details are beyond the scope of this work how ever from the block diagrams it is easy to see the amount of interdisciplinary work required to provide real estate for storing information.



HP StorageWorks 4000/6000/8000
Enterprise Virtual Array (EVA)
Family data sheet

Figure 3: Family of Data Storage Arrays offered by HP StorageWorks



	EVA4000	EVA6000	EVA8000
Drive capacity	56 drives	112 drives	240 drives
Random reads bandwidth	14,500	27,600	55,900
Sequential reads	335 MB/sec	750 MB/sec	1420 MB/sec
Sequential writes	260 MB/sec	515 MB/sec	525 MB/sec
Host ports	4	4	8
Device ports	4	4	8
Loop switches	0	2	4
Drives supported	10K rpm 146/300 GB, 15K rpm 72/146 GB high-performance FC, 500 GB FATA	10K rpm 146/300 GB, 15K rpm 72/146 GB high-performance FC, 500 GB FATA	10K rpm 146/300 GB, 15K rpm 72/146 GB high-performance FC, 500 GB FATA
Operating systems supported	Windows 2000, Windows 2003, HP-UX, Linux, Tru64, OpenVMS, Solaris, AIX, NetWare, VMware	Windows 2000, Windows 2003, HP-UX, Linux, Tru64, OpenVMS, Solaris, AIX, NetWare, VMware	Windows 2000, Windows 2003, HP-UX, Linux, Tru64, OpenVMS, Solaris, AIX, NetWare, VMware
Connectivity	DAS, integrated iSCSI and SAN	DAS, integrated iSCSI and SAN	DAS, integrated iSCSI and SAN

For ordering information, please see QuickSpecs: **www.hp.com/go/quickspecs**

Table 1: Data Storage Array Systems Compatibility and Limits

Systems limits and illustrative figures are taken out off http://www.hp.com

CLARiiON CX3 UltraScale series
Leading the midrange with advanced capabilities and a wide range of solutions.
CX3-10
CX3-20
CX3-40
CX3-80

Figure 4: Family of Storage Arrays offered by EMC Corp.

STORAGE ARRAY	MAXIMUM CAPACITY	MAXIMUM DRIVES	SYSTEM CACHE	HOSTS PER ARRAY
CX3-10	**30TB**	**60**	**2GB**	**60**
CX3-20	**59TB**	**120**	**4GB**	**120**
CX3-40	**119TB**	**240**	**8GB**	**240**
CX3-80	**237TB**	**480**	**16GB**	**480**

Table 2: CX3 UltraScale Disk array Characteristics

* Illustrative pictures and systems limits are from http://www.emc.com and http://www.emc.com/products/systems/cx_compare_357.jsp

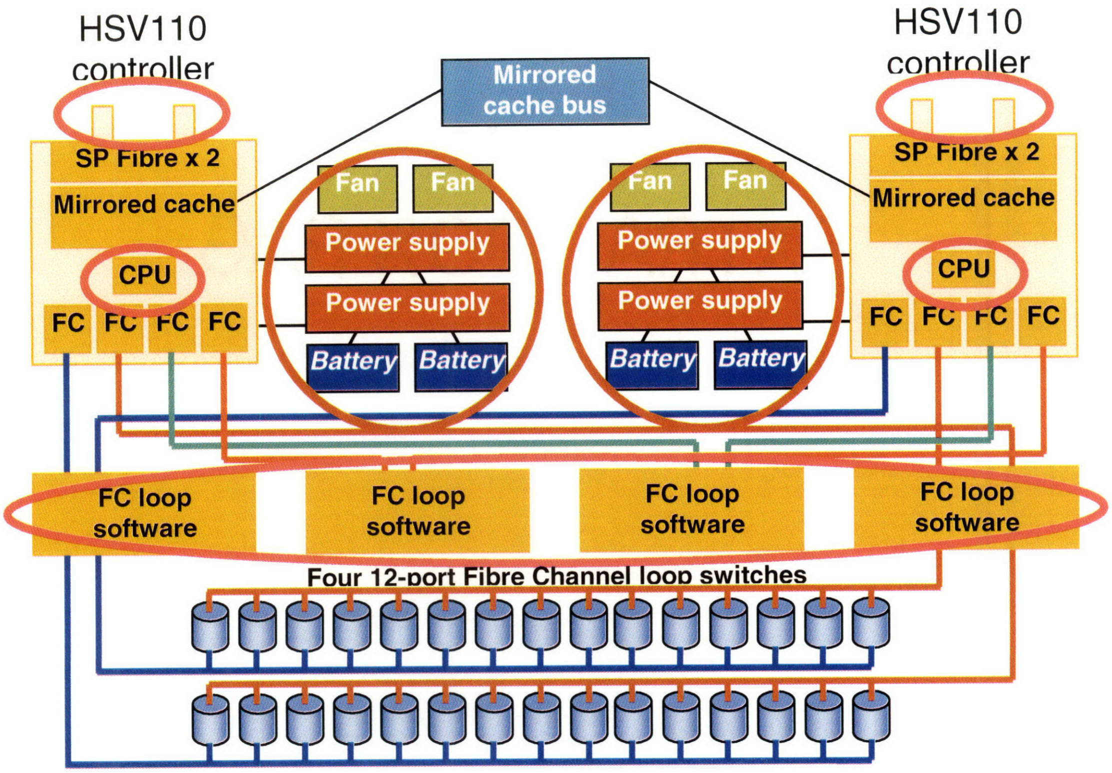


Figure 5: First Generation Full Fibre Channel EVA 5000 Systems Architecture

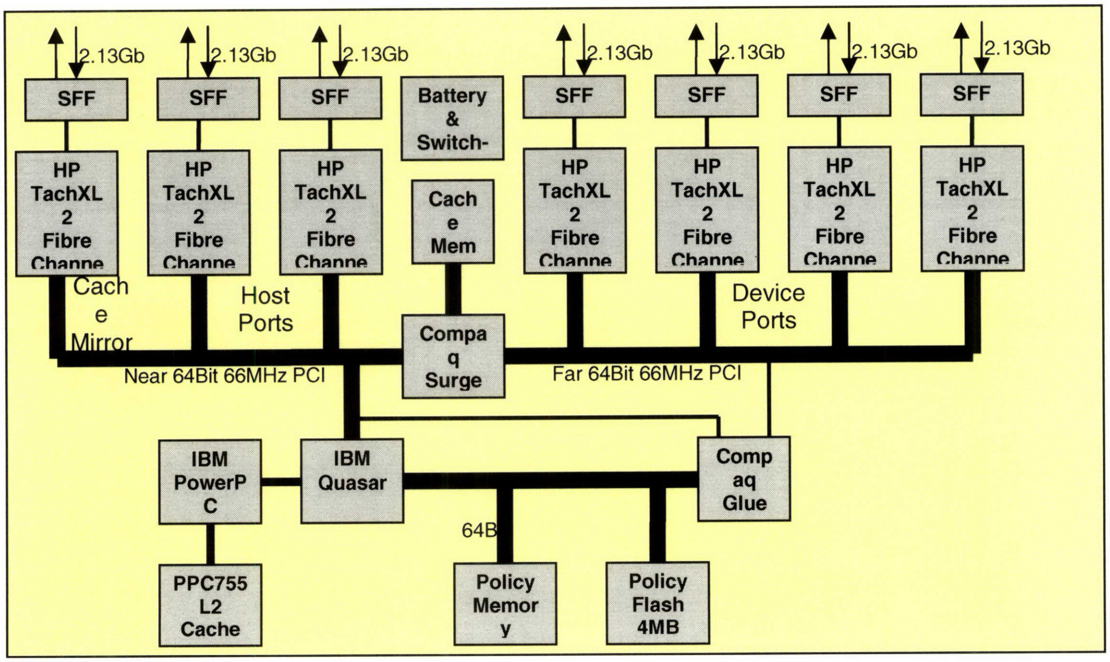


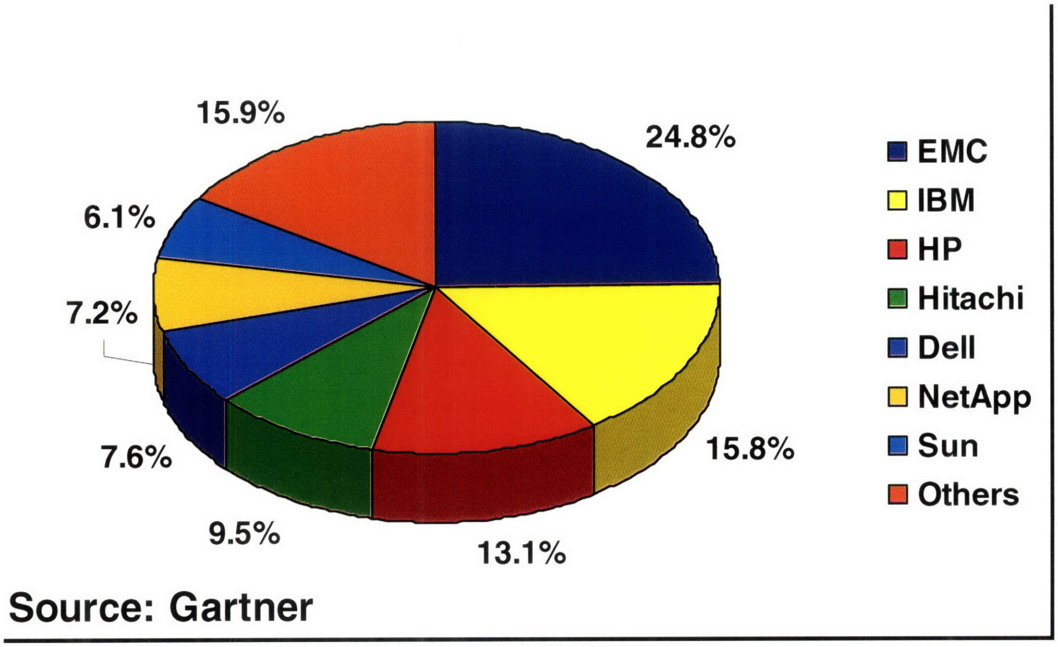
Figure 6: HSV110 Storage Controller Block Diagram for EVA 5000 Storage Array

2.4 Data Storage Industry Reverse Engineering Practices

Upon becoming a part of our lives, internet proliferated the way we create, store, exchange and access information. As a result, the last few decades witnessed a tremendous connectivity improvement afforded to individuals and organizations. And the subsequent content explosion we have faced is unprecedented in the history. It is said that 5ExaBytes (Billion Gigabytes) of recorded content has been produced in 2002 alone and further 30% year-by-year increase is projected for the foreseeable future. To put things into perspective, the 5ExaBytes produced in 2002 equals to all words ever spoken by human beings and assuming a world population of 6.3Billion, it also means 800MB of recorded data per person on the planet. A Berkeley study suggests that 92% of this new content produced resides on magnetic media; primarily on hard disks (HD). We believe that these informative statistics paint the market opportunity presented to companies in the Data Storage Business[16].

Figure 7 on the following page categorizes EMC Corp + Dell, IBM, HP and Hitachi as the top-tier technology companies competing to provide solutions for enterprises of all scales who seek to store, protect and manage information at the right service levels and the right price. The fact that 70% of the information real estate or the place where information actually lives is provided by these top four companies reflect the concentrated nature of the industry. The fifth contender Network Appliance is also significant. A later entry into the market, Network Appliance has posed challenges to the market incumbents in various segments. Many in the industry viewed them as a disruptive influence.

[16] How much information, School of Information Management and Systems UC Berkeley 2002. http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/



Source: Gartner

Figure 7: 2006 External Controller Based Market Share

I believe there are examples of reverse engineering in the industry with profound effects and game changing consequences. For instance, the only successful & significant non-US entry into the industry has been The Hitachi Data Systems of Japanese origin. Hitachi's midrange storage array offering has a remarkably similar architecture to EMC Corp's Midrange CLARiiON line. It is true that different brands of Storage Arrays share many similar architectural traits but there are some landmark characteristics that differentiate trend setting vendors like EMC Corp, IBM, HP, Network Appliance and Sun from each other. Many of these characteristics have to do with the history of the company and are complementary to the general computing environment that they have been proposing as value to users over time. Their past product offerings, and in a way their latest offerings, reflect in them all the product development that has been done, including the lessons learned and even perhaps the lessons unlearned. The continuity of these experiences and the accumulated know-how tends to give a distinct character to the Storage Array that is being conceived or proposed.

At the end, it is clear that these experiences reflect on the brand equity of the vendor.

Many engineers feel that Hitachi Midrange offering has a striking similarity to EMC CLARiiON product line. The common talk is that Hitachi Entry into Data Storage industry is just another enactment of past Japanese entry into global automotive industry: Start with imitating market leaders and capture market share by efficient manufacturing and a pricing scheme to sell.

Here is a suspect choice by Hitachi: The default number of drives recommended for a typical RAIDSET[17] for building LUNs[18]. CLARiiON Storage Arrays defaulted to five drives for a considerably long time. A historical choice due to a precedence of having 5 back-end SCSI I/O busses[19] to service the entire storage array. That meant the ever-presence of an initial-configuration enforced static load balancing for the entire array. Drives were automatically distributed across available busses preempting possible system bottlenecks. Over time SCSI bus interface was replaced with Fibre Channel technology. This act rendered the usefulness of this 5-Drive-Raidset design choice irrelevant, however CLARiiON rolled over the 5-Drive-Raidset configuration into the Fibre generation to keep a feeling of familiarity for the Data Storage Administrators.

[17] RAIDSet is a compilation of data stripes and parity stripes that contain a set number of physical drives. Depending on the RAID choice it can contain a dedicate parity drive, rotating parity, or a mirror set. LUNS are carved out of RAIDSETS.

[18] LUN or logical disk is a set of contiguous chunks on a physical disk. Logical disks are used in some array implementations as constituents of logical volumes or partitions. Logical disks are normally transparent to the host environment, except when the array containing them is being configured. The RAIDBook A Source Book for RAID Technology

[19] SCSI I/O bus is a backend data transmission bus that interconnects physical drives to the mid-plane and the controllers of the storage array. SCSI is the Small Controller System Interface which is a standard for physically connecting and transferring data between computers and peripheral devices, in this case the hard drive.

When Hitachi came up and unveiled a new Fibre midrange array with same 5-Drive-Raidset recommendations out of the blue it was obvious that the choice for drives was not coincidental. Hitachi had no logical reason to default to a five drive RAIDSET.

Reverse engineering acts go deeper than some architecture choices. Entire features that constitute parts of a business plan are duplicated when storage vendors consider the subject feature fits to the demands of the customers. Indeed, we have to note here that some customers have deep pockets and make considerable purchases and in return they can exert considerable power over the vendors, influencing the new product features. Enterprise-class customers fall in to this category, generally they want to focus on their core-competencies and perceive sustaining an efficient data storage infrastructure as a burden. High-end EMC Symmetrix line realized this phenomenon early on and built its brand around a top rated services organization. Offering a tiered approach to services, this total customer experience approach spurred some of most innovative additions to products of the company. For instance, inspired and modeled after Israeli Merkava Tanks – A complex system which used to fail frequently in battle filed and required expert attention. Experts were not always available in remote fields hence the tank was equipped with an automatic call home feature in case of trouble which enable remote intervention - each storage array is equipped with a similar call home feature connecting the storage array to EMC Global Control Center during failure states. This gives EMC a chance to fix problems remotely. In fact, in most cases, customers realize the existence of a problem when contacted by the EMC services organization. Failures are handled either remotely or a service technician is sent in for a remedy, either way the array remains operational.

Due to changing demands and competition HP adopted a similar approach. HP support now carries a native call home future. HP Storage works even went as far as adding this feature free of charge to their already installed customer base.

CHAPTER 3

Capturing Innovations in Data Storage Industry

Chapter 3 will first stress the importance of identifying business objectives for reverse engineering. There is a lot to be explored here and one has to make sure he knows what he is after before he takes a deep dive into the process.

Next is an argument for a tiered reverse engineering approach. I will suggest that peeling back the complexity of system is analogous to peeling an onion:

- Reverse engineering will uncover an upper layer and reveal the heuristics of the system around its interfaces.
- Each uncovered layer brings in a different quality and quantity of information.
- The distilled knowledge gathered in each layer will help to uncover the next layer of detail.

I will then introduce the "impulse response analysis technique of a black box system" from signals and systems theory in support of this proposal.

I will also propose a closed system model to define the layers of reverse engineering.

Lastly, I will mention, in a tiered format, what can and can't be captured in a reverse engineering exercise and highlight the time constraints on the project as far as the data storage industry is concerned.

3.1 The Reverse Engineering Business

Deciding what to reverse engineer in a system, and to what degree, is a business decision. You must decide what you really want to know and why because data storage systems are complex and reverse engineering a system you don't know much about can be very costly. You must identify and articulate the business problem before initiating the process; otherwise you might end up working hard but still waste time.

Previous findings will determine future directions in reverse engineering. Especially at the beginning, the number of unanswered questions far out numbers the ones with answers so it's a good idea to reevaluate the direction of the investigation every so often. Take advantage of what is available to determine what do next.

I would also suggest establishing a formal feedback mechanism. Such a forum would make it easier to make adjustments to the strategic direction and serve to synch-up all groups involved. There is a sizeable human side of this process and efficient coordination will increase the chances of capturing all business objectives.

Reverse Engineering in the data storage industry focuses on non-commoditized parts and functionalities. Hard Drives and individual components like CPUs, off the shelf parts like power supplies and HBA (Host Bus Adapters) [20] are generally rated as non-critical and their presence is only noted from a cost point of view.

[20] In computer hardware HBAs connect a host system to other network and storage devices. The term is primarily used to refer to devices for connecting SCSI, SAS, Fibre Channel devices.

The real prize and target of a reverse engineering effort is highly available controller systems, various RAID algorithms, caching algorithms, ease of management features. Depending on the business objectives, many other minor features and that are staged behind the curtains with in the operating environment can be targeted as well. Reverse engineers will work around the commoditized parts and note their interactions between core-features or functions but at the end they will focus on the features that impact the bottom line of the business.

Rival systems are designed to keep business secrets secret. This limits the availability information through conventional means. Web and documentation search helps but will not answer all questions. I am not saying that information out of web should be discounted but should be taken with a grain of salt. I believe the system and uncertainty it generates requires a black box approach with out prejudice and the reverse engineer must deconstruct the system in terms of its input and output characteristics to capture the hidden secrets.

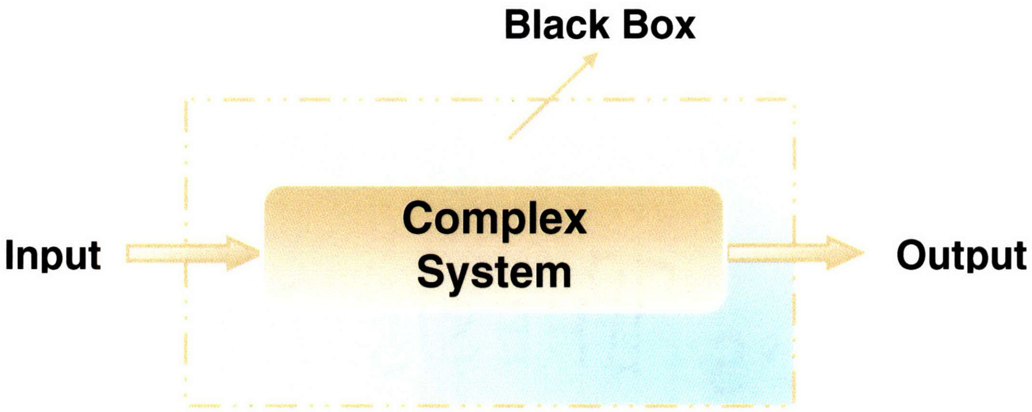


Figure 8: Classic Black Box System Model

Figure 8 displays the classic black box model of a system. Black box model does not pay attention to internal heuristics or operational mechanism of a system. It is only concerned with input and output characteristics.

I believe black box method is a great way to analyze a data storage system but I suggest a slight modification to do greater justice: The closed loop black box model. Storage arrays are attached to a host/server system during their normal operation. The presence of a host system literally closes the loop around the two systems both visually and operationally.

This Closed Model[21] approach brings in further analysis advantages over the classic model: The input source is no longer independent and external. This means that the same jobs keep circulating in the model helping the measurement of job response times and over all job accounting. Finally, Figure 9 examines the model in a visual form.

Now that the model is established, reverse engineer can start the experimental stage of the process. The goal is to characterize the system this can be done by controlling the input thru the server system and observing the output at the storage system. I will propose a methodology frequently used in signal and system theory in the next section

[21] Closed Model: No external input, jobs circulating in the model. Feedback loop scenario. p401. The Art of Computer Systems Performance Analysis, Raj Jain.

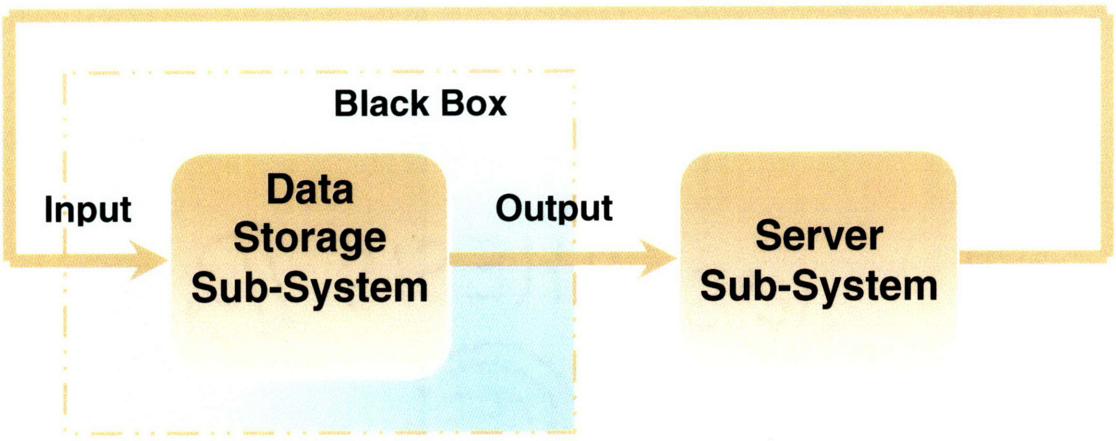
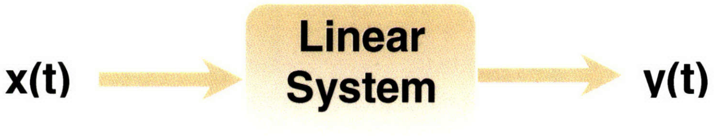


Figure 9: Closed Black Box Model for Data Storage Systems

3.2 Adapting Impulse Response Analysis to Storage Array Characterization

There is an application of the Fourier Transform to Linear Time-Invariant systems in analog circuit design which states that any Linear Time-Invariant system with input x(t) and output y(t) can be described as in Figure 10, where x(t) and y(t) represent signals as a function of time and they are linearly dependent on each other through a system function which is called a transfer function. Apart from the visual model, the system can also be described in the form of a differential equation with constant coefficients having initial conditions of zero.



$$A_n \frac{d^n y}{dt^n} + A_{n-1} \frac{d^{n-1} y}{dt^{n-1}} + \ldots + A_1 \frac{dy}{dt} + A_0 y$$

$$= B_m \frac{d^m x}{dt^m} + B_{m-1} \frac{d^{m-1} x}{dt^{m-1}} + \ldots + B_1 \frac{dx}{dt} + B_0 x$$

Figure 10: Linear Time Invariant System Model and its differential expression in general form

Fourier Transformation of the above general differential equation will transform the expression from the time domain into the simpler frequency domain.

$$\rtimes(jw) = ⌶(jw)\bowtie(jw) \Rightarrow ⌶(jw) = \frac{\rtimes(jw)}{\bowtie(jw)}$$

H (jw) represents the system function or the transfer function. The system input is already determined by the user and if the transfer function associated with a system is known, the characterization of the output becomes a deterministic task. If I know the transfer function of a system, then I know all there is to know about the system, as I already control the input of the system.

If the input is an exotic like δ(t) Dirac's delta function[22] (also known as an impulse function or a signal spike), the output from the black box is a truly special case. In fact, theoretical justification below shows that the response of a Liner Time-Invariant system to an impulse function δ(t) is the transfer function of that system. This observation is a great help in uncovering the secrets of a system.

Let input x(t) $= \delta(t) \Rightarrow$ **X** (jw) = 1

then **Y** (jw) = **H** (jw) **X** (jw) = **H** (jw)

then the output of the LTI system is :

y(t) = $\mathcal{F}^{-1}$[**Y** (jw)] = $\mathcal{F}^{-1}$[**H** (jw)] = h(t)

Impulse response of a system equals its system (transfer) function. Feeding an impulse to a Linear Time-Invariant system will result in outputting of the transfer function and once the transfer function is known, we know all there is to know.

Data Storage Systems are not by any means Linear Time-Invariant. Any theoretician that attempts to describe a data storage system will find that the initial conditions of the differential equation will be non-zero. However, reverse engineers are not seeking pure characterization of a given storage system. If we apply carefully chosen workloads comparable to Dirac's delta function, then short of a transfer function, we can still learn quite a bit about the system. For instance, by pushing a system to its limits we can deduce system bottlenecks; by varying time-exposure to workloads we can quantify system stability; and by using variable workloads we can get a good idea of how the system performs and can discover many hidden features.

Needless to say, by varying our input or workload we can infer a lot about what the system is doing and how. The mathematical fundamentals behind this methodology are shown above[23]. Experience has shown that stress testing, which is testing the system at its limits is an adequate empirical method to characterize the system. Finding system bottlenecks generally gives the analyst a great deal of information about features as well. The discipline that specializes in creating stress workloads is performance engineering. There is a special synergy between performance engineering and reverse engineering.

[22] Dirac Delta $\delta(x)$ is a mathematical construct introduced by the British theoretical physicist Paul Dirac (1902-1984). Informally it is a function representing an infinitely sharp peak taking up one unit area and is a function $\delta(x)$ that has value zero everywhere except at $x=0$, where its value is infinitely large in such a way that its total integral is 1.

[23] System and Signal Analysis by Chi-Tsong Chen Chapter 3, Section 3.4 Liner Time-Invariant Continuous-Time Systems—Convolutions, Impulse, question of initial time, impulse response and unit step response pp: 117-140

This synergy is curiously comparable to synergy between innovation and performance engineering. Let me borrow an example from a completely different walk of life for making this point clearer: Automotive and Motorcycle companies leverage racing as the ultimate confirmation of their design abilities. Racing in the circuit with specially built cars gives them an opportunity to expose various features that they intend to incorporate into their future designs. The racing circuit is truly the place where the rubber meets the road. Everyone observes how each contender handles the challenges of the road and gets a chance to compare contenders with industry accepted benchmark. The act of observing the problems with the vehicles under duress and addressing these issues for the next race brings in innovation. The inspiration behind the inventive genius is in fact the insatiable desire to win the race trophy. Pushing the system (automobile or motorcycle) to its limits generates a treasure chest of content or problem sets or some say opportunity sets to be addressed. This phenomenon was also picked up by an HBS case study:

> Honda's successful 4 stroke engine eased the pressures on Fujisawa by increasing sales and providing easier access to financing. For Sochiro Honda, the higher horse power engine opened the possibility of pursuing one of his central ambitions in life: to build and race a high-performance, special-purpose motorcycle—and win. Winning provided the ultimate conformation of his design abilities, racing success in Japan came quickly. As a result, in 1959 he raised his sights to the international arena and committed the firm to winning at Great Britain's Isle of Man—the "Olympics" of motorcycle racing. Again Honda's inventive genius was called into play. Shifting most of the firm's resources into this racing effort, he embarked on studies of combustion that resulted in a new configuration of the combustion chamber, which doubled horsepower and halved weight. Honda leapfrogged past European and American competitors—winning in one class, then another, winning the Isle of

> Man manufacturer's prize in 1959, and sweeping the first five positions by 1961.[24]

Just like the inventive genius who determines the next innovation by testing the system to its limits, a prolific reverse engineering will do the same to uncover the innovation.

* * *

The basic reality is that the law of diminishing returns will put a cap on how much information a reverse engineer can extract out of the second layer of system complexity. If there is a particular innovation of interests or a strategic reason to go any deeper, then the reverse engineer will rely on the knowledge obtained at the previous levels and then break the second layer storage system model into subsystems as in Figure 11.

The subsystem model takes advantage of the black box model as well. This means third level is simply a reverse engineering effort at a different scale, the methodology remains similar to layer 2: Take advantage of Reverse engineering and performance engineering methodology. Work around the interfaces and push the systems to its limits for characterizing the subsystems of the storage array. The only notable difference is that the reverse engineer will now have to define new metrics for measurements and utilize different set of tools to dig deeper into the bowels of the system.

[24] Honda (B) HBS Case 9-384-050 Dr. Richard T. Pascale of Stanford Graduate School of Business

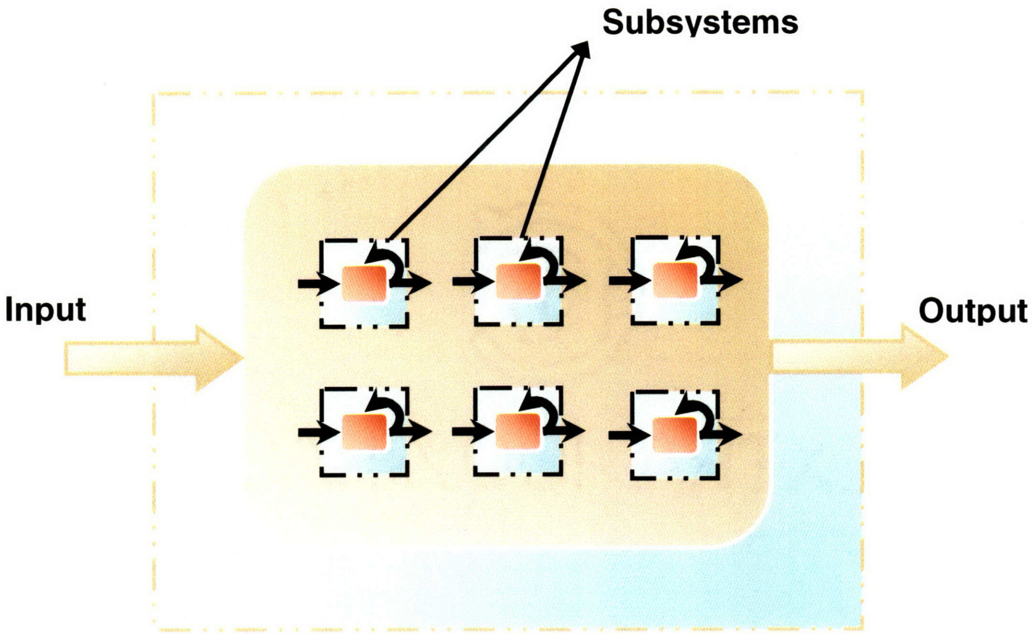


Figure 11: Subsystem Level Reverse Engineering Model

3.3 What can be captured?

Layered approach will yield enough information about the competitor to see what he is doing and how he is doing it. Let's see what can be captured in each reverse engineering layer.

The suggestion for an initial layer is along the lines of a literature search plus some hands on learning activity. I think it is possible to gather a lot of knowledge from documentation, industry trade shows, attending special classes, customer bulletins and simply owning the system in question. A careful reverse engineer also knows about the existence of "some dark matter". Dark matter refers to the information that has not said or left aside in many of the aforementioned sources. Drawing on experience, one can make intelligent guesses, ponder over the blanks and make expedient attempts to shed light over this dark matter. This attempt is not a waste of time. A bit of deductive preparation work at the beginning is always helpful before embarking on a voyage.

Next layer is black box analysis—that is quantifying the quality of output of the system. Take the system for a ride around the racing circuit—but before that don't forget to define your metrics and decide what the measurement criterion will be like. We are playing a strategic game with multiple systems so results are always relative. Be careful about interpreting absolute measurements. So, how fast does the system go? How does it corner? How does it respond in wet surface? Make sure to answer all these questions in comparative manner.

It is time to kick the tires and pitch contenders among each other in this race. Once the ride is over we will already know a lot: We will know maximum performance under a variety of workloads; we will uncover several different types of system bottlenecks; we will have a good mental map of the internal system and data flow through systems subsystems.

Is the system performance sustainable? In one reverse engineering case I found out that it was not. The next logical question is why not? Thanks to our initial work around documentation and in house knowledge of how processors interact with data mapping, I was able to deduce that the culprit was the low powered processor. A set of race conditions formed and inadequate computational power coupled with dynamic built up of data structures over time diminished the system performance as a function of time.

There may be gaps but what the reverse engineer does is close to what the system-innovator is doing so it may not be necessary to have the entire picture. Everyone faces similar challenges so it's easier for the reverse engineer to see the dilemma of the innovator. Experience comes in handy here and the reverse engineer will fill in the gaps that he can't account for.

In addition, the contextual difference between the reverse engineer and the innovator can bring in some value. Working for different companies is not the only difference between reverse engineer and the innovator. One has critical the other has a creative role and sometimes this is exactly what is needed to bring everything together and make a good innovation the best. Most innovations are incremental and "arriving" to the innovation from a different context may bring in a second incremental, only this time reverse engineer will incorporate that change into his own product.

Next layer of detail is characterizing subsystem of interest. Working around the interfaces but this time at a different scale it's possible to characterize yet again the interactions between the sub-systems. The difference between subsystem and system level effort is generally the metric of measurements and equipments to perform such measurements. At this level, complexity, specialization and naturally time to capture the essence increases. For example, in data storage arrays interactions between the disk drives and the controllers may be measured to uncover the control algorithms or file system behavior. This level of detail will require different set of analyzers, tools and expertise. The instrumentation may be purchased or built in house so the challenge is not mission impossible.

This kind of scale-invariant approach to analyzing complex systems, transforms reverse engineering into an act of interface engineering. Once again, engineers hardly need the verbatim; a characterization is enough to provide intellectual capital needed for their next set of products to lift off. As was mentioned in the first chapter, when Mig-25 was reverse engineered, the whole effort was not for cloning an adversarial aircraft--it was to devise and engineer the F-15 which is a superior and more advanced fighter. Mig-25 was the inspiration and the competitor. The strategic intent was to draw from a good design and make the best.

3.4 What can't be captured?

Is there such a thing? I think not… If there is a will then there is a way. I believe that any competent organization may easily turn "all valuable stones upside down" given enough financial resources, time, and experienced intellectual capacity.

Does this mean revere engineering will be walk in the park and everything can be captured? No, everything can't be captured. There are number practical business considerations to watch for:

The Clockspeed[25] of the industry is a major concern. Data storage industry is nimble and the five-forces-of-the-industry[26] is relatively weak but the Clock-speed is relatively fast and getting even faster thanks to increased rivalry among established competitors and a few new entrants. The life cycle of a major software and hardware platform release is about three years. That spells the expiration date for an industry player's innovation driven advantage.

[25] Prof. Charles H. Fine provided well-documented case studies of companies that have either thrived or failed in anticipating and adapting to rapid change in his book. CLOCKSPEED: Winning Industry control in the Age of Temporary Advantage. The term Clockspeed is coined; it is to business genetics what lifecycles are to human genetics. Drawing upon the motif "learning from the fruit flies" Analyzing cycles of product, process and organizational innovations the fastest moving industries, Prof Fine introduces powerful management tools for anticipating and mastering the forces of change in a world where all business advantage is temporary. The faster an industry evolves—that is, the faster its Clockspeed—the more temporary a company's temporary advantage. The key is to choose the right advantage—again and again.

[26] Porters Five Forces shape the market in which businesses compete. These are: the threat of new entrants, the bargaining power of suppliers, the bargaining power of buyers, the threat of substitute products or services and the rivalry among existing competitors

[27] Moore's law describes an important trend in the history of computer hardware: that the number of transistors that can be inexpensively placed on an IC is increasing exponentially, doubling approximately every two years. This observation was first made by Intel co-founder Gordon E Moore in a 1965 paper. The trend has continued for more then a half century and is not expected to stop for a decade at least and perhaps much longer.

Furthermore, data storage industry operates in synch with the mainstream computer industry and the complementary nature of the relationship forces both industries to keep up with each other's hardware or software improvements. To that end, the famous Moore's Law[27] is still determining the evolution frequency or "The Clock-speed" of both the computer and the data storage Industry.

These factors bring a time limit to how many layers can be uncovered in a reverse engineering attempt before the product is out of fashion. Product reverse engineering and product development are parallel efforts that seamlessly feed innovations and new features to the next product release. The challenge for both processes is not just what to do something but also when and how fast to do that thing.

Aside from industry evolution rates, some of the players have chosen to remain vertically integrated. These companies either bring in additional expertise from their downstream, supplier side of the business or create special synergies with their complimentary or upstream buyer side of the businesses.

For instance, those who have downstream microchip or IC (Integrated Circuit) businesses have been able to leverage their relationship with fabrication foundries and successfully integrated proprietary ASIC (Application Specific Integrated Circuits) designs into their storage architecture. This capability offloads software tasks to specialized hardware devices and diverts traffic away from the main CPU enhancing overall system performance. This situation is a capability based competitive advantage and constitutes a barrier for players who only have data storage focus.

Some of the industry-incumbents have full blown computer/server businesses on the upstream/complementary side. They are able to pull customers towards their products either by a pricing strategy or taking advantage of network effects[28]. They bundle computer and data storage systems together and shift costs to win strategic accounts.

The take away here is that the fruits of reverse engineering may be inapplicable for the next product set unless the act of capturing innovations is aligned with the business context of the industry.

Decompilation[29], is also out of limits because copyright laws in United States and Europe has curbed commercial use of this software except for interoperability purposes. At any rate, simply copying and pasting decompiled source code is perceived to bring little value because quality assurance and interoperability cycles in software development are just too complex to integrate something out of the context. A lot of engineers do not believe in Decompilation at all.

[28] Network effects create an increased value for to a potential customer as a function of the number of other customers who own the good or take advantage of the service.

[29] Decompilation is the opposite of compilation— it is the construction of source code from the machine language (compiled) form to a form recognizable by human cognition. Decompilation requires special Decompiler programs or suites.

CHAPTER 4

The Reverse Engineering Framework and Cost Analysis

Following Chapter 3's introduction to reverse engineering methodology in the form of a case study on Data Storage Industry, chapter 4 will present the theory behind this method and contemplate its financial burden. I will show in this chapter that the "reverse-engineering-around-the-interfaces" method introduced in previous chapters is in reality a self-similar (Fractal) approach to complex systems analysis and this self-similar (Fractal) analysis framework is basically dividing the system around its interfaces into subsystems and reiterating across scales. This is a deconstruction process.

In grasping the logic of the reverse engineering process, I will need to go into certain details of Chaos Theory, Fractals, and Object Self-Similarity. In addition, I will also introduce a new set of taxonomy for constructing a sample reverse engineering algorithm.

The quest for every reverse engineering venture is to unveil an innovation surrounded by a mist of uncertainty and unknown. Experimental sciences seek the help of Chaos Theory, Probability Theory and the study of Entropy to explain uncertainty, unknown and randomness.

The way to demystify and de-cloak the hidden value behind an innovation calls for conducting a full and factual investigation. In my opinion, the turnaround comes from enlisting the help of a Fractal Analysis Framework. This proposition is hardly unwarranted. Why? Because chaos theory attempts to; make sense out of chaos and to describe the unknown. To do so, it takes full advantage of the fractal analysis framework with great success.

Reverse engineering process is also challenged by uncertainty and it attempts to uncover the unknown in chaotic circumstances. Therefore, it is easy to see the synergy; any tool that helps chaos theory should also help revere engineering as a matter of one to one correspondence.

My history research suggests that modeling chaos is a well defined business with a rich past of discovery and application; hence I see no reason to face the kind of deterministic uncertainty in a reverse engineering endeavor without taking full advantage of the readily available tools of the chaos business.

Thanks to scientists like Benoit Mandelbrot and Von Koch who have broadened our horizons on the phenomenon of chaos over the last century. The line between chaos and order is fading and the phrase "out of chaos comes order and out of order comes chaos" is acquiring a new perspective and meaning. We can steadfastly claim that reverse engineering is not a black art anymore. On the contrary, it is streamlined undertake with the light at the end visible even before the tunnel itself is visible.

4.1 Introduction to Fractals and Self-Similarity

Various descriptions of self-similarity or fractal behavior exist in the nomenclature and more often then not these descriptions are generalized in terms of Non-Euclidian geometry. We will opt to pick a syntax that is more general or holistic in nature so that the adaptation to reverse engineering is clearer. This particular definition treats fractals as general objects. Reverse engineering process is not a traditional area of interest for chaos theoreticians;

therefore I think it is necessary to keep the definition as a superset of statistical or geometrical self-similarity.

Self-Similar or Fractals Objects *are built using a set of rules and recursion where some aspect of the limiting object is infinite and the other is finite, and where at any iteration, some piece of the object is a scaled down version of the previous iteration.*

Statement above delves into the meaning of self-similarity, which is *appearing the same at all magnifications*. Scalability is a concern for all growth-based complex systems. The main question for any scalable system is what happens as the system grows due to higher demand? Will it be able to keep up with the original service levels and response times? Once the scalability issue is resolved, a healthy growth of the system can be expected. Scalability is a dominating issue in the Data Storage Industry. All stakeholders; customers, providers and suppliers expect storage arrays to counter-balance the data explosion mentioned in Chapter One and keep up with providing services that are satisfactory at the very least.

In the absence or deterioration of system scalability, the fear of diminishing returns in economies of scale dominates the agenda of decision makers and once the system resources are wasted an investigation for an alternative, more successful and robust architectures is initiated. All the major players in the Data Storage Industry address the scalability issue by applying the principles of modular architectures across their system designs. This modular or layered approach brings in an opportunity for reverse engineers and that is the use fractal analysis methodology which is being defined here.

[5] The Koch Snowflake or Koch star is a mathematical curve and one of the earliest fractal constructs to have been described. It first appeared in 1904 on a paper written by Mathematician Helge Von Koch: "A continuous curve without tangents constructible from elementary geometry".

[6] Mandelbrot set is a set of points in the complex plane that forms fractals. Mathematically it can be defined as the set of complex c-values under iteration

7 Julia set is a holomorphic function consisting of points under repeated iteration

The previous definition of self-similar or fractal object stated a finite dimension. It's time to introduce new taxonomy. From now on, we will call this finite dimension the Microcosm and the metamorphosis of the finite dimension at an infinite iteration as the Macrocosm. The definitions are below:

Microcosm *is the smallest fraction (finite fraction) that displays the properties of the greater whole. Properties of the microcosm have to be clearly defined in order to achieve successful scalability and avoid interoperability problems.*

Macrocosm, *which is the whole itself, is really a scaled up version of the Microcosm. It is achieved at infinity or at a numerically high iteration.*

Upcoming examples showcase the application of fractal growth or scalability: A depiction of a mathematical constructs like The Mandelbrot Set[31], The Julia Set[32] and The Von Koch Snowflake[30], computer graphics rendering of moon and mountains. These renderings are achieved at much higher but finite iterations.

Microcosm: The primary building block is an equilateral triangle with a hallowed base replaced with two lines with the same length as each of the sides.

Scalability: Replacing each side of an equilateral triangle with our predefined microcosm and then continuously repeating the replacement of each line segment with the microcosm gives us the snowflake.

Macrocosm: Shapes up to be a snowflake. Note that with in the same frame it is possible to extend the microcosm to infinity through iteration and obtain a very fine granularity snowflake.

Sometimes in the literature this process is called creating an infinite coastline in other words extending the perimeter of the snowflake to infinity.

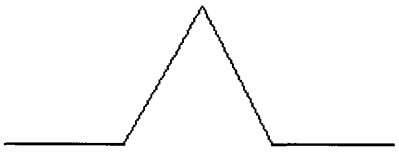


Figure 12: Microcosm of Von Koch Snow Flake

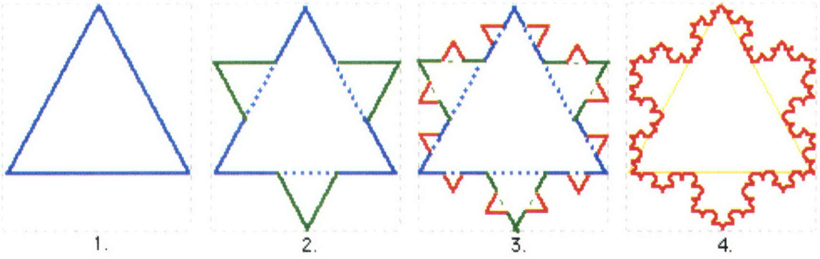


Figure 13: Growth of Microcosm to Macrocosm; Von Koch Snowflake at iteration fou

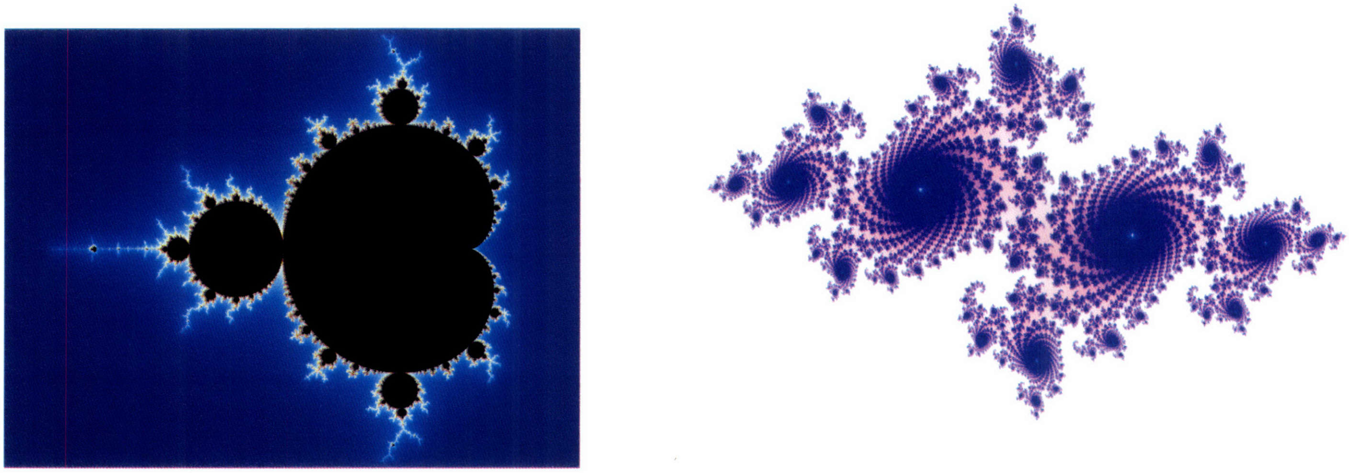


Figure 14: Mandelbrot Set (Right) Julia Set (Left)

Figure 15: Computer Rendering Fractal Mountain and Moon using Self-Similar Algorithms

Fractals are also often used in generating or rendering naturally appearing shapes or textures such as land or cloudscapes in computer graphics. Now that the taxonomy and examples are clear, it is possible to introduce a simple algorithm for fractal growth as follows:

- *Define a Microcosm;*
- *Define as set of rules that will constitute iterations; and*
- *Scale up to Macrocosm at a predetermined iteration or iterate until infinity.*

In reverse engineering our concern is in dissecting the macrocosm; the end product or, in our case, the Data Storage System. A reversal of the fractal growth algorithm is necessary to achieve what we define as fractal analysis framework, shown below. The proposed algorithm is broad, but it serves to clarify what we have already conveyed. Interfaces correspond to the iterations:

- *Identify the Macrocosm;*
- *Identify a set of rules that will layer or distinguish Iterations; and*
- *Layer down to a Macrocosm at a predetermined iteration.*

Customized for a data storage array, version below displays an algorithm identifying the layered analysis components.

- *Data Storage System (Macrocosm);*
- *System or Subsystem Interfaces; and*
- *Target Feature set or Target Subsystem (Microcosm).*

A reverse engineer is seeking the building blocks, the fraction that represents the whole or the self-similar unit. And we will use the same uncovering methodology for the system and subsystem in each layer, this is akin to systems characterization methodology mentioned in Chapter 3; Adaptation of impulse response systems analysis techniques.

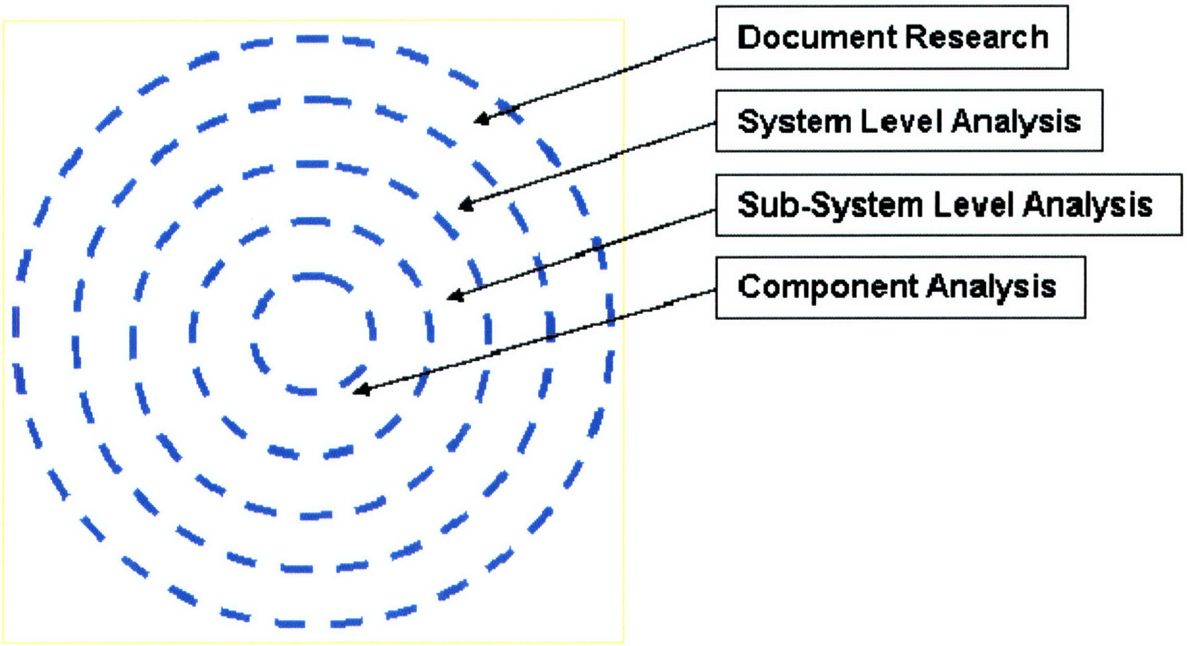


Figure 16: Tiered Reverse Engineering Approach to Data Storage Systems

Figure 16 identifies four layers of reverse engineering for data storage arrays. Each layer of abstraction provides feedback to the next layer below in terms of strategic direction and system know-how. Each level is progressively more complex, requiring more specialization, tooling and expertise. Each level also represents a new band of costs, which I will discuss in section 4.1.

4.2 Costs of Reverse Engineering

I am going to identify operational costs associated with reverse engineering data storage systems as Fixed Costs and Variable Costs. And once each cost item is properly categorized as a fixed or a variable cost, I will then calculate a pro-forma total operational cost. As a final a step, I will structure the costs around the "onion layers" i.e. identify costs associated with each layer of reverse engineering.

While fixed costs are the expenses that do not change in proportion to the activity of the business with in the relevant period of operation, variable costs depend on how we execute the business activity with in the relevant period of operation. Before I can identify each and every reverse engineering cost item and then categorize them into fixed costs and variable costs, I think it is important to say a few words about the period of operation (Duration of Reverse Engineering).

The product life cycle curve of a data storage array is at most three to three and a half years. This means the whole sales cycle: Market Introduction, Sales Growth, Product Maturity and Sales Decline takes place with in this short three to three and a half years. Why is the sales cycle so short? The reason is simple: Data storage industry is not vertically integrated. Pulling on one end of the rope are the downstream buyers who demand that the data storage industry products robustly attach to their refreshed and brand-new servers and simultaneously remain highly-responsive to the application workloads that run on these brand new and more powerful servers. Pulling on the other end of the rope, are the downstream suppliers who introduce newer technologies and interconnect devices at shorter then three to three and a half year sales cycles and then stop selling "out of fashion" parts. The data storage industry; sandwiched between these upstream and downstream powers and challenged by new entrants lacks adequate bargaining power to mandate its own sales cycle in the market place. As a consequence the dynamics and the clockspeed of the data storage industry is such that every three to three and a half years a new generation of hardware platform is released. Obviously it is wise to reverse engineer a particular storage array right after its release and unwise to continue reverse engineering beyond its sales life cycle. You can't stop in front of a moving train.

In reality, the reverse engineer has even less time. About one and half or at most two years to finish his/her work. Why? Because the enterprise will require an additional year or so to absorb and productize the knowledge gained. Two years plus one year equals three years and we are up to the magic number of years before a particular storage array is "out of fashion" in the industry.

Following, two tables will itemize the costs of reverse engineering for a storage array. I like to note a few disclaimers before bring them to the reader's attention:

- These are only rough dollar costs. There are many purchasing and configuration scenarios to choose from so the final costs will depend on the scale of the array and the channel from which it was acquired. Tables below assume that the array in question is a mid-end array. A low-end array would be significantly cheaper and a high-end would be an order of magnitude more expensive.

- I am not contemplating reverse engineering any remote data mirroring technology used in data storage disaster recovery. If I did, total fixed cost plus total variable cost would instantly multiply and there would be additional overhead costs.

- Many teams and individuals cooperate and allocate valuable company time to produce tangible reverse engineering results. This involvement also depends on the nature of innovation as well so my estimate for the cost of human resources below is classified a variable and it is conservative.

- Specialized software tools are only authored once; consequently their full cost is not passed only to one reverse engineering project. It is the same idea with the servers and the hardware analyzer equipment; all these cost will depend upon how many times the reusable items were utilized in different projects—sometimes many times.

- Finally, the following cost structure should be considered informal and regarded only as a general guideline. Variable costs are hard to predict, so the spread in my estimates are high.

APPROXIMATE FIXED COSTS ASSOCIATED WITH REVERSE ENGINEERING DATA STORAGE ARRAYS	**US DOLLARS**
Cost of purchasing storage array and hard disks (Mid-end)	**$1,000,000**
Cost of setup, install, tuning, 3 year support	**$25,000**
Cost of software licensing	**$50,000**
Cost of Attach and Switches for all three interfaces (iSCSI, Fibre, SAS)	**$50,000**
Cost of Lab Space	**$5,000**
* Cost of power and cooling for 2 years	**$40,000**
Total Fixed Costs	**$1,170,000**

Table 3: Fixed Costs. *Power costs will be structured in the next two pages

APPROXIMATE VARIABLE COSTS ASSOCIATED WITH REVERSE ENGINEERING DATA STORAGE ARRAYS	**US DOLLARS**
Cost of human resources for 2+ years	**$300,000**
Cost of developing software tools necessary for testing	**$30,000**
Cost of hardware analyzer equipment for testing	**$30,000**
Cost of servers	**$100,000**
Cost of lectures, education, traveling	**$5,000**
Total Variable Costs	**$465,000**

Table 4: Variable Costs

Grand total of reverse engineering comes to about $1.7 million, but it would be safe to assume that it will inflate to $2million with additional unforeseen costs.

* * *

We talked about a tiered approach to reverse engineering and made an onion layer analogy previously. Each tier or layer represents a different level of reverse engineering engagement and consequently additional costs. In the next few pages, I will explore these costs in a qualitative manner, except for the power consumption costs, which I will treat quantitatively in relative detail.

Preliminary Research involves web research, reading technical guides, documentation, interviewing with end-users, following literature or bodies responsible for endorsing standards and attending seminars or trade fairs to collect any additional up-to-date information. The most significant investment required is time and intellectual capacity; hence preliminary research is the least financially taxing among "all the layers of the onion".

The process can be completed in a few months or could be done on an ongoing basis as a background activity. Either way, it requires meetings with different functional groups with in the organization and a lot of brainstorming. It sounds easy and compared to the other layers, the effort is not as exhaustive but, I would like to emphasize here that the preliminary research is actually rather important and should be conducted with care. At the end, an irreversible purchasing decision appears for the next layer. If the organization acquires a rival array that does not offer reverse engineering value, then the expense would be a waste of significant resources. It is also important to order the array with the right configuration and appropriate features installed and in order to make these right judgment calls one needs to do the necessary "homework". If the "homework" is done properly, the overall ride is smoother from start to finish.

System Level Analysis requires significant investment. Purchase of a full blown Data Storage System is an expensive business transaction. The cost of this layer can easily run up to a few million US dollars, depending on the choice of array. In the data storage market, arrays are generally classified as high-end, mid-end and low-end. Low-end arrays can go up to fifty thousand US dollars, mid-end arrays can go up to a million or so and high-end systems can start at around a million and can go up to tens of millions of dollars.

A typical mid-end array will have about 1,000 drives, sold bundled with special replication software and a number of mission specific add-on software components. All these cost items are itemized and priced separately in the invoice. In addition, the array will likely require an extensive support agreement and to top all these costs, the installation, the setup and the performance tuning will be at an extra charge. The bargaining power of a storage array buyer is limited. Sometimes big buyers fill up large data storage centers and they may require future upgrades i.e. promise of more sales. Under these circumstances there will be a reprieve, but this will certainly not be the case for the Reverse Engineering party so one should expect to pay the full list price.

The data storage system will require a powerful server/host system for creating the application workload necessary to mimic real-life, actual-use. CPU cycles required to drive a storage system could run up to significant MIPS. Thanks to strides in fabrication technology; both the server and the storage array performance has been complying with Moore's law in tandem. Today's IT center is a finely balanced and tuned room of servers and storage arrays; therefore an investment in storage technology requires a comparable investment in server technology.

We have accounted for the fixed costs related to owning an array. There are operating costs as well, such as power and cooling.

Arrays consume quite a bit of electrical power and constant air conditioning is needed to diminish the chance of hard disk and other component failures; hence keeping the operational environment cool is important for preserving investment. Power is needed to keep the disks spinning and the servers running. Cooling or airflow is needed to prevent disks and solid state components from overheating.

POWER CONSUMPTION OF A MID-END STORAGE ARRAY	
1,000 Hard disks at 19 Watts per Drive	**19,000W**
60 Enclosures at 150 Watts per enclosure	**9,000W**
2X Controllers at 1,000W per controller	**2,000W**
100 Fans at 7W per fan	**700W**
3 Power supplies at 200W per supply	**600W**
120 Interface circuits at 5W per circuit	**600W**
%20 for additional electronics like interface switches etc…	**6,400W**
Approximate Total Power Consumption	**39,000W**

Table 5: Itemized Power Consumption of a Storage Array

Table 5 itemizes power consumption of a mid-end storage array. Additionally, servers that drive the storage array will draw about 2,500W of electricity each. A typical mid-end storage array needs about four servers to clock maximum performance and that makes an additional 10,000W of power for the server component.

Cooling will draw about the same amount of power consumed by running servers and the storage, plus perhaps about 10% more due to thermo-dynamic inefficiency. That means total cooling watts will be at about 54,000W.

Operational power consumption due to running and cooling storage/server system comes to about 49,000W+54,000W=103,000W which equals to 10.3kW. In 2007, Massachusetts, USA retail electricity price for industrial users was 13.4 cents per kWh on average, distribution charge was 1.35 cents per kWh, and generation charge was about 7.6 cents per kWh => Total Electricity Charge = 22.35 cents[33]. We have to assume continuous operation for 2 years as storage arrays are designed to run 24 hours a day and it might be risky to switch them on and off. Therefore, the total cost of operating the array will be 10.3kW x $0.2235 x 24 x 365 x 2 = $40,331.

Wandering into a "Data Storage Room", whatever the motivation, is anything but cheap. And for reverse engineering purposes, micro-targeting a set of features that is deemed valuable is not an option either i.e. one can't simply purchase special feature or an option then reverse engineer only that subsystem of interest.

From an operational perspective, subsystems of a storage array are bundled in such a way that it's all or nothing so the reverse engineer needs the whole system. Rival systems are incompatible as well, so owning a system by a particular brand is having interoperability with in one brand, no operational synergy with other brands exists, obviously this is on purpose.

[4] Energy information administration: Official Energy Statistics from the US Government. http://www.eia.doe.gov/cneaf/electricity/epm/table5_6_b.html Transmission and generation charges from Chicopee Electric and Light Company http://www.celd.com/cits/index.php?PID=cel_2005_rate_card.htm

Major brands like to demarcate their products in order to consolidate their position. Data Storage Vendors consider a foothold in a particular customer account as a positional advantage and perceive interoperability as a weakening of that position.

Data Storage Industry is not commoditized yet so there is already a barrier to entry with the high cost of owning and operating systems.

Small Cap. Players who may want to enter into reverse engineering domain will be discouraged by the financial burden that they may have to bear. Reverse engineers will also need special software tools to test and drive the system to its limits. Although, certain forms of benchmarking tools freely exist on the web, an exhaustive investigation requires simulation tools developed in-house. And to further the monitoring of the input and output signals, special analyzers built by third parties are required at a cost of thousands of dollars.

Sub-System Level Analysis: There is a limit to the amount of information that can be extracted out of system level analysis, yet for all that is worth; "the mere tip of the iceberg" provides adequate empirical evidence to pursue other gems of interest. The good news is many subsystems of value can be exercised through system level access portals, unfortunately not all features can be examined to the core with enough granularities. To unveil more of what is simmering in the bowels of the subsystems, engineers would need highly specialized analyzer equipment or software. Even capturing simple input and output characteristics at this level necessitates special tools. This equipment is generally very expensive or may not even exist and have to be designed and built in house. It is also not unusual to take the system apart in this level of detail and work on what if scenarios to understand the mind of the original innovator. Human resources involved in this layer are generally the key people in the organization. Costs may not be as high as the initial system level costs, because the fixed costs of purchasing the storage array has already been absorbed. Yet at the end it may be close to the system level with all the expertise and the time extra time involved in this layer of effort. Yes, this level of detail requires significantly more time and patience.

Component Level Analysis: In general the level of detail found at this layer is unnecessary to understand features and functionality of data storage systems. Low level components are mostly off the shelf parts. Some ASIC designs in the industry have been customized, but reverse engineering ASIC designs have not been relevant due to existence of alternative architectures. Decompiling exact source code is useless as well, because interoperability among competitive systems is almost no existent. Besides characterization of source code may be achieved by other means such as solving the state-system of the code. Finally, atomicity of system design starts breaking down in component level analysis and value added features and functions of Data Storage Industry are no longer recognizable. Professionally, I think going into this level of detail is unnecessary.

CHAPTER 5

Benefits of Reverse Engineering

After examining all the costs and complexity associated with reverse engineering data storage systems, it is time to take a look at the benefits or the return on investment. There are two simple questions:

- As an organization, do we really want to spend millions to buy equipment from a rival in order to subject that equipment to reverse engineering? After all, the purchase will bring profit to that rival- is out there to "steal our lunch".

- Are there any benefits in reverse engineering? If so what are they?

As far as the Data Storage Industry is concerned, I think the answer is a big yes for both questions. Spend the money and acquire the array. In fact, spend the money as soon as you hear the general availability announcement of your competitor and be the first one to get the array, fresh out of the manufacturing line. I believe the intensity of rivalry in the industry and the difficulty of capturing market share more then vindicates this expense. Any threat against the market share of a data storage vendor should be evaluated and should be evaluated fast.

* * *

What are the benefits of reverse engineering? From 30,000 feet above, the main beneficiaries of reverse engineering are the initiator-organization and the society. To a lesser extent, there are benefits for the target-organization whose product is subject to revere engineering as well, so I will spent some time on this topic too.

The discussion in this chapter will revolve around the five main points as summarized below:

- Knowledge gained by Reverse Engineering stimulates creativity and may result in a new innovation. Stimulation of creativity presents a new opportunity for the organization: An ability to add on to the absorbed knowledge and innovate in the market place.

- Knowledge gained by Reverse engineering can be used in surviving the challenges of the market place i.e. fend off competitors and conserve resources.

- Standard economic theory suggests monopolistic industries will sell lower quantity of goods at higher prices. Reverse engineering provides a means to break a monopoly over technical advances to the benefit of society.

- The organization is able to authenticate the size of the threat posed by the competition and able to act – or to decide not to act – based upon more complete information.

- Firms that reverse engineer rival systems will be able to provide facts that can deflate inaccurate advertising by rivals. The beneficiaries of this increased honesty and transparency are the buyers.

5.1 Creativity

Edward Crawley defines creativity as "ability or power to cause to exist, to bring into being, to originate, or to combine in a new way".

Figure 17: Creativity…

He goes further and attempts to tier creativity:

- Raw or pure creativity that is thinking of something that no one has ever thought of (rare indeed).
- Transfer of experience or metaphor from one field to another. (Very common)
- Organizing of knowledge, finding patterns, interpolating and extrapolating (Even more common)

Crawley[34] also suggests a number of approaches to stimulate creativity. The list is extensive, but he prioritizes reverse engineering, benchmarking and patent search on the top for creativity exercises.

Quantifying the exact value of raw or pure innovation is difficult but it is easy to see that a new, unique or differentiated product will command a higher value in the market place then the old-fashioned, ordinary and undifferentiated. Reverse engineering boldly brings into the organization the "unattainable" differentiated innovation that was beyond anyone's reach for one reason or another. There could be many reasons for not innovating, from perceived cost of innovations, lack of resources (human or material), lack of initiative, misguided strategy, organizational inertia to plain lack of innovative capacity.

The reverse engineering process brings the essence of an innovation under an organizations finger tips. In effect, this is a new chance to recapture a position in a lost market and this quality makes the reverse engineering effort almost as valuable as the innovation itself. Now, the possibility of capturing at least part of the business success of the original innovator exists and perhaps more. The "more" part is of interest for the business savvy reverse engineering planner. Surely, innovation alone does not guarantee business success. Trying to walk with somebody else's shoes might seem redundant and yes, the position of the first innovator is filled with rightly earned glitz. Nonetheless, being second to a market is not necessarily a losing proposition, especially in the high technology business.

In fact, market experience shows that it is the third or fourth entrant that encounters difficulties i.e. stumbles walking with the new shoes.

Second entry is actually well positioned for success. Why? Because, the second entrant has a chance to evaluate the efficiency and business success of the original innovator and this kind of feedback enables the second entrant to fill

[5] Edward F. Crawley is a Professor at Aeronautics and Astronautics Engineering Department at MIT. Annotations are from ESD.37 Systems Architecture IAP class in 2005. Lecture 3 notes on concepts and creativity.

potential gaps, clear the warts and bugs then launch even a better product. There is clear market value in an expedited reverse engineering venture and a consequent swift product launch.

There is more to commercial success then an earth shattering innovation. The story of the browser wars that took place between the brilliant innovator Netscape and the entrenched incumbent Microsoft is a testament to the assertion: Second is better!

Microsoft shows us that incumbents may leverage their consolidated positional advantage effectively or use other means to neutralize the value proposition of the innovator for achieving greater business success. Netscape, unfortunately, can't show us anything anymore. The company is long gone, even the Netscape browser itself is a part of history despite its unprecedented brilliance.

5.2 Economic Benefits of Reverse Engineering

Sometimes it's hard to quantify and comprehend the economic value that a reverse engineering project could bring to the table. The general argument presented by Samuelson and Scotchmer underlines the virtues of the practice for various industries: "Lawyers and Economists have endorsed reverse engineering as an appropriate way to obtain such information, even if the intention is to make a product that will draw customers from the maker of the reverse-engineered product." [35]. However, in the case of the software industry, Samuelson and Scotchmer state that high costs and inherent difficulties in reverse engineering object code erect formidable barriers and they state that the process is not an efficient way to develop competing but non-identical products.

Quoting a technologists remark, Samuelson and Scotchmer suggest software reverse engineering does not "lay bare a program's inner secrets. Indeed, it cannot. The inner secrets of a program, the real crown jewels are embodied in the higher levels of abstraction material such as the source code commentary and the specification. This material never survives the process of being converted to object code" [36]

[6] James Pooley, supra note 1 at 5-16; David Friedman, William M. Landes & Richard A. Prosner, Some Economics of Trade Secret Laws, 5 J Econ. Persp. 61, 71 (1991) See also sources cited infra notes 26-36 and 162.

[7] See Frederick P Brooks, Jr. N Brooks, Jr., F.P. 1986: "No Silver Bullet—Essence and Accidents of Software Engineering," *Information Processing 86*. H.J. Kugler, ed. Amsterdam: Elsevier Science Publishers B.V. (North Holland): 1069-1076. (Invited paper, International Federation of Information Processing (IFIP) Congress '86, Dublin, Ireland, September 1986.) Reprinted in *Computer*, **20**, 4 (April 1987) 10-19. Also reprinted in *The Mythical Man-Month, Anniversary Edition*, Frederick P. Brooks, Jr., Addison-Wesley, 1995

At first glance, costs and difficulties associated with reverse engineering software systems may appear prohibitive and it may be surprising for some to find the process coming up over and over again as a common occurrence in the industry. Nonetheless, there are solid and fundamental reasons why reverse engineering remains a beneficial strategic tool.

It is true that for everyone concerned a reverse engineering is difficult to execute except. However, crafting a state-of-the-art, original, killer-software-application out of thin air is even harder and more expensive and a riskier proposal then getting ultimately inspired by an already proven and marketable product. I thought looking at the issue from the lens of engineers and project managers out of the development discipline might cast some light to the benefits question.

One of the pioneers of Software Engineering profession is Fred Brooks. In his seminal book "The Mythical Man-Month: Essays on Software Engineering" [37] and his complementary article "No Silver Bullet – essence and accidents of software engineering"[38], he articulates the extent of complexity-wall facing the software discipline with appealing analogies.

His collection of articles in the book "The Mythical Man-Month" offers a good discussion of all the resulting problems in productivity[39], complexity and reliability due to non-linear nature of the software development process.

[8] Johnson-Laird Supra Note 181, at 896.

[9] See Frederick P Brooks, Jr. *The Mythical Man Month*, Anniversary Edition Addison Wellesley (1975, 1995)

[10] Software development productivity is no where near as hardware development productivity. Over the last decade the industry has not been able to achieve one order of magnitude improvement in productivity, reliability and in simplicity.

In his trademark silver bullet analogy, Brooks compares software development projects to a werewolf-monster of missed schedules, blown budgets and flawed products adding to exorbitant cost overruns and frustrations.

He mentions about the existence of natural and desperate cries from the non-technical management side of the profession for a "silver bullet" to slay this "monster-werewolf" of complexity and make software development costs drop as rapidly as hardware development costs have done over the decades.

But at end, he reasons that there is no silver bullet in sight. He foresees a road but not of a royal kind; breakthroughs, innovations and new promises approaching but all in discrete and incremental fashion. Brooks says, revolutionary "silver bullet" killer solutions are as mythical as the werewolves themselves. Software development is not a realm of miracles.

Brooks classifies software complexity into accidental[40] and essential[41] of kind. He claims that most of the accidental complexity has already been taken care of by the advent of popular high level programming languages like ADA, C++ and Java. Nonetheless, the essential complexity inherent in the nature of problems that the software tries to solve still remains embedded in the business problem that the customers intend to solve at the first place. And no programming language could take care of such complexities.

[11] Accidental Complexity arises in software or in development process which is non-essential to the problem to be solved. It could be caused by approach chosen to solve the problem, ineffective planning or low priority given to a critical path project. For instance, Widows O/S crashing while doing long complex calculations is an accidental complexity, because problems solver picked windows as the calculation platform.

[12] Essential Complexity refers to a situation where all reasonable solutions to a problem must be complicated because the simple solution would not adequately solve the problem.

Furthermore, Brooks describes the most difficult part of development as deciding what to do at the very beginning: "It is establishing technical requirements, including interfaces to people, machines and other software. No other part of the work so cripples the resulting system if done wrong and no other part is more difficult to rectify later...... In many cases the client usually does not know what questions must be answered and he has almost never thought of the problem up the detail necessary for a specification leading to the code... a long iteration between the designer and client ensues..." [42]

Software Engineering remains a difficult domain to live and innovate. Progress is difficult: Accidental complexity is already pretty much out of picture and we are mostly left with incremental innovation opportunities to address remaining essential complexities. In this harsh environment, when the perceived payoff is adequately large, despite all the costs, it's easy to see why reverse engineering could be valuable and beneficial. That is of course, if there is already someone in the market place who has already overcome the barriers to development and has produced a competitive and marketable product.

What is the economic value a third-party (competitor) innovation that you can't achieve or worse yet, can't implement? You have thrown in enough resources, suffered from delays and cost overruns and all this time you have been painfully aware of the famous Brook's law which states:

[13] No silver bullet. Essence and Accidents of Software Engineering, Computer Magazine, April 1987 by Frederick P. Brooks Jr. http://www.virtualschool.edu/mon/SoftwareEngineering/BrooksNoSilverBullet.html

[14] Brooks Law: Coined in the book The Mythical Man-Month, states that due to complexity of software projects new workers added to the endeavor must first become educated in the work that precedes them. This education diverts resources away from the projects it self and temporarily diminishes the productivity. Another reason is the increase in communication overhead as the number of participants increase.

“Adding man power to a late software project makes it later”. [43]

What realistic choices do the decision-makers have? Is it time to throw in the towel? I think not, sometimes cost of reverse engineering could be far less of a punishment then beating the path of paths already beaten, and get beaten at the end. There is no reason to reinvent the wheel. Preventing duplication of research investment is the social rationale behind reverse engineering; and it can be a strategic tool for firms to survive in the market place. Reverse engineering could be priceless if you are left behind and losing hope.

5.3 Benefits for other stake holders

Economists have long argued for the benefits and economic value of reverse engineering. The value for the society is simple enough: Reverse engineering prevents duplicate innovation efforts.

Other sections of chapter five concentrate on the benefits of Reverse Engineering from a microeconomic or a price-theory perspective. Price-theory suggests that resource allocation decisions are made in an environment where goods and services are bought and sold. The operators in this environment (firms) keep their financial interests on the fore front of all business decisions that they make. This means all strategic decisions, including product innovation or where to invest research dollars are made with the state of competition and prevailing market conditions in mind. Furthermore, the dynamics of the marketplace encourages competing firms to shut each other from each other’s innovations and markets. Each company keeps corporate-secrets secret and everyone fends for himself. If this means to rediscover what has already been discovered then so be it. A company’s objective is to get ahead and all is fair with in the boundaries of law. In this philosophy, macroeconomic health or the performance structure of the regional economy or global economy is not in the forefront or a priority.

When we change hats and re-calibrate our focus from microeconomic to macroeconomic heuristics, it becomes evident that the society does not really reap benefits from duplicate efforts to innovate. The contention for limited resources brings forward the opportunity cost of duplicating innovative efforts as a priority rather then the individual firm's profit motivation to compete for the similar innovation. Naturally, macroeconomic perspective, perceives the value proposition of reverse engineering more attractive as the practice provides a venue out of duplicate innovative efforts.

There is another reason why reverse engineering is favorable for the greater good. If the industry in question is concentrated or a particular firm is meeting most of the supply curve, then that firm is said to enjoy a greater ability of determining pricing flexibility. This means, that the potential is there for the firm to capture most of the value in the value chain by effectively weakening the bargaining power of the end users and suppliers. Now, this is the realm of a monopolist.

There is a lot of debate about monopolies; lots of pros and cons but at the end I believe an industrial monopoly is only good for the monopolist. And in the "laissez-faire" economy where we live in, unfortunately innovation is a legitimate means of consolidating or creating a monopolistic hegemony.

Innovation allows the precipitation of a new supply and demand curve in the industry. And in fact, if the improvement is truly ground breaking and inimitable, an altogether new industry may form. Either way, innovator will reap financial benefits by meeting the increased demand and will act to consolidate his grip on the industry. This shift in balance of power has financial consequences for the buyers, suppliers and the society as whole. No one can argue against a temporary advantage brought about by a great innovation, but it is important to keep this positional advantage temporary. An everlasting extraneous pressure for the other stake holders in the value chain is counter productive for the health of the economy as a whole.

As far as the society is concerned, reverse engineering is a lever that brings in and encourages competitive forces into the marketplace and that is all good for the stake holders in the value chain. In fact, any strategic instrument that can dilute industry concentration and make sure the positional advantage of the innovator stays temporary will receive a warm welcome by the establishment. As a rule of thumb, society prefers perfect competition where no one has the capacity to significantly influence prices of goods and services. If diluting market concentration is the ways to this end then reverse engineering is one of the means.

Reverse engineering also helps the innovator! This could be a surprise for some. And one may wonder why? Why would anyone want to see the heat turned-on on himself and deal with the extra tussle and bustle of the market place, as if the market was a rose-garden to walk around at the first place? For the ultimate strategist who seeks to establish a real institution with a lasting legacy there is a good explanation: The threat of entry and consciousness about the transient nature of the positional advantage brought in by the innovation keeps the innovator and his organization running on its toes at all times. Successful innovation brings growth. And, with the temporary advantage clause attached to the innovation, the growth of the organization is engineered in a more thoughtful and robust manner. Nimble start is kept at a nimble and one time innovator transforms into a serial innovator. As reverse engineering opens the backdoor for the competition, the shadow it casts keeps corporate execution, operation, strategy and marketing at an optimum. At the end, the value of the enterprise will increase for all stake holders. We must remember; diamond is a piece of coal made under pressure.

5.4 Strategic Directions

Reverse engineering is also very important for setting the strategic direction of a company. The knowledge of where the competitor stands with its product line filtered out of the marketing confetti is very important. As it stands, in software and computer architecture the design work is not ad hoc. There are trends and general directions in implementation and many current products carry with them a number of hooks and nooks that are strategically placed with in the innovations and those hooks and nooks are the messengers of future product directions. Once the entire knowledge is extracted by reverse engineering and analyzed, the people who carry out this task acquire an idea about the general outlook of the competitor's next innovation or the product set. This information critical and constitutes a direct feedback signal into strategic plans of the company.

5.5 Checks and Balances

Profit seeking enterprises tend to overstate their messages inn an effort to sway buyers' opinions and pull buyers towards their products and services. Vendors also attempt to make a concerted effort to conceive perceptions in the minds of the buyers for promote future sales. To create this effect, various instruments and strategies are utilized but in general, the idea is to increase the brand equity of the product or the firm itself in the eyes of customers.

I am not going to rate the promotion efforts of the firms as disinformation campaign but in most markets there is excess or a pollution of information that makes product comparisons difficult. Abundance of claims and counter claims and information bombardment just makes the life of buyers difficult.

For instance, why is there an expiration date on a can of soda? Is it necessary to convey this information for health reasons? Is it true that a mix of caffeinated sugary-water can expire in a pressurized can and pose a health hazard? The answer is probably no but we are not sure because we haven't reverse engineered the product. These vendors are trying to bring in a perception of freshness into the market place and they are targeting buyers who care about to compete fresh beverages and compete with the other beverage makers that actually do have a genuine freshness attribute associated with their products.

Truth is told by reverse engineering, the practice brings out the good, the bad and the ugly into the open. As the mystery unfolds, there is a greater chance of distinguishing the bark of the subject from its bite. And if rivals engage in reverse engineering, they will cross examine each others claims and bring forward the truth to keep the playing field level and once the truth is be told customers can make realistic comparisons.

I have seen that even under the existence of the checks and balances system mentioned above, there is a tendency to exaggerate messages. Nonetheless, I believe if the messages were not countered in the marketplace by the knowledge gained from reverse engineering, it would have been even more difficult to hash out the reality and keep everyone honest. Customers can not possibly examine all there is to verify about the products and services that they are acquiring by themselves so this job is ideally cut for the competitor.

Bibliography

Pamela Samuel, Suzanne Scotchmer. *The Law and Economics of Reverse Engineering.* The Yale Law Journal, Vol. 111, No. 7 (May 2002), pp 1575-1663 Doi: 10.2307/797533

Case No 04-3654, Brief of IEEE-USA as Amicus Curiae in support of appellants and reversal. http://www.ieeeusa.org/policy/POLICY/2005/DavidsonvBlizzard.pdf

Chikofsky, E.J.; J.H. Cross II (January 1990). *Reverse Engineering and Design Recovery: Taxonomy in IEEE Software.* IEEE Computer Society: 13–17

von Hippel, Eric (2005). *Democratization of Innovation* Published 2005 MIT Press ISBN 0262002744

Brooks, Jr., F.P. (1995). *The Mythical Man-Month: Essays on Software Engineering, 20th Anniversary Edition.* Reading, MA: Addison-Wesley.

Fine, Charles H. (1998). *Clockspeed: Winning Industry Control in the Age of Temporary Advantage.* Basic Books, New York ISBN: 0-7382-0153-7

Porter, Michael E. (1980). *Competitive Strategy: Techniques for Analyzing Industries and Competitors.* Free Press First Edition **ISBN-13:** 978-0684841489