FUNCTION AS AN OBJECT: A STUDY ON APPS

A Master's Thesis

by

Murat Pak

The Department of Communication and Design
İhsan Doğramacı Bilkent University
Ankara

SEPTEMBER 2012

Dedicated to Zettam,

a friend who dominated virtual worlds.

FUNCTION AS AN OBJECT:
A STUDY ON APPS

Graduate School of Economics and Social Sciences

of

İhsan Doğramacı Bilkent University

by

Murat Pak

In Partial Fulfillment of the Requirements for the Degree of
MASTER OF ARTS

in

THE DEPARTMENT OF
COMMUNICATION AND DESIGN
İHSAN DOĞRAMACI BİLKENT UNIVERSITY
ANKARA

SEPTEMBER 2012

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Arts in Media and Visual Studies.

....

Assist. Prof. Dr. Ahmet Gürata
Supervisor

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Arts in Media and Visual Studies.

....

Dr. Özlem Özkal
Examining Committee Member

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Arts in Media and Visual Studies.

....

Assist. Prof. Dr. Ersan Ocak
Examining Committee Member

Approval of the Graduate School of Economics and Social Sciences

....

Prof. Dr. Erdal Erel
Director

ABSTRACT

FUNCTION AS AN OBJECT: A STUDY ON APPS

Murat Pak

Master of Arts in Media and Visual Studies
Supervisor: Assist. Prof. Dr. Ahmet Gürata

September 2012

This study analyzes function of objects within the framework of digital technologies and relates the function with how we design and perceive technical artifacts. The main purpose of this examination is to change the way we approach to functions and reconsider them within the boundaries of use and design. The key element of this study is how applications are positioned in the social structure.

Keywords: *Function, Technology, Smart Device, Application, App.*

ÖZET

BİR OBJE OLARAK FONKSİYON: APP'LER ÜZERİNE BİR ÇALIŞMA

Murat Pak

Medya ve Görsel Çalışmalar Yüksek Lisans
Danışman: Assist. Prof. Dr. Ahmet Gürata

Eylül 2012

Bu çalışma objelerin fonksiyonlarını dijital teknolojiler bağlamında teknik araç-gereçlerin tasarımı ve algısı ile ilişkili olarak analiz etmektedir. Bu araştırmanın amacı fonksiyonlara olan yaklaşımımızı objelerin kullanımı ve tasarımı çerçevesinde yeniden ele almaktır. Bu çalışmanın anahtar öğesi uygulamaların sosyal yapı içerisinde konumlanma biçimidir.

Keywords: *Fonksiyon, Teknoloji, Akıllı Cihaz, Uygulama, App.*

ACKNOWLEDGEMENTS

First of all, I would like to thank everybody who has put their precious efforts into my work and made it possible.

I would like to express my deepest gratitude to Dr. Özlem Özkal for countless hours she had spent to guide me during long sessions of idea exchange. Without her guidance and support, this work would be impossible to finalize.

I would like to thank Assist. Prof. Dr. Ahmet Gürata for his great encouragement and suggestions as well as his help through the complicated problems I've encountered during the thesis process.

Finally, I would also like to thank to my family for their incredible support. My father İskender Pak for his inspiring enthusiasm to technology, my mother Nilüfer Pak for being the best ma in the universe, my sister Ece Pak for being crazy enough, my dog Kuki Pak for his friendship through the dark nights, my cat Miyu Pak for teaching me to save my documents frequently and finally my girlfriend Zeynep Özfidan for her patience during the tough times. Thank you all, you guys are awesome!

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Technology is diffusing into the society faster than any age humanity had encountered so far. The way we communicate with each other is extensively being affected by this technology as a result of this diffusion, and our perception is re-shaping itself depending on the apparatus of today. As a matter of fact, human activity, one of the key elements of communication, is under the influence of new concepts that are formed during the process of this technological evolution, such as meta-media. This study approaches the functions of today's devices with the help of these concepts of technology.

The second chapter defines software and hardware, and explains their relation with each other. It is also the passage between the actual and the virtual within the concept of virtualization; the process of turning an actual object into a virtual object. The relation between these concepts are essential to understand the relation of society with the technology of today since technical artifacts are always one of the key elements of our daily activity.

The third chapter is the connection between the software and the social effects of it. The notion of "app" has changed the way we approach to the devices of today as well as the functions that are ascribed to them. Technology has a huge role on this subject since the smart devices are evolving as a part of this growing structure of meta-media and connecting both of the sides together: The men and the machine. Can we relate the behavior of society to the devices we use?

The fourth chapter is directly related to those technical artifacts of today: Smart devices. The general approach of this chapter is finding the connection between the physicality of the device and the functions that are ascribed to this body. Various function theories are used for conceptualization of the relation between the device and the function;

therefore, design of the product is one of the main concerns of this chapter's research.

This study should be interpreted as a way of approaching the function as an object and therefore the whole research aims to identify function as a part of this approach.

CHAPTER II

A “VIRTUAL” MACHINE

2.1. Software Kingdom

With the emergence of technology in our daily lives, we adapted ourselves to work with various tools and new inventions that are not only mechanical devices but also functional through the use of more complex technological ecologies. The transformation of functionality from the mechanical to the digital was not an easy form of transformation, but still the adaption phase was smooth and necessary, just like we did in earlier eras when various technologies emerged in the society. Each new type of technology is surely another notch to change the way society thinks, but we can’t deny that digital age brought enormous speed to this process of infinite cycle of creation. We can see the wide variety of these new tools in every part of daily life and we use them all the time, sometimes with the proper knowledge and sometimes with only instincts, without even knowing that we are using a fully functional device to fulfill our need of functional ability for a certain state. This cycle of change is not only happening to the technology but also to us, starting from a conceptual level and (non)ending in a fully logical way of thinking, just like how television and other media inventions affected to whole society.

The technology behind the gear of today is like a universal language between objects which is starting from the root of mathematics and still imploding into the new concepts and levels of virtual as we continue to try to understand our needs for the future. This beast technology was so naïve in the beginning that it sometimes becomes impossible to fill in the gaps between the technological jumps if we look at this change from above.

Today's apparatus all have the hardware and software parts, just like the puzzle pieces that form up a bigger image. Sometimes the software form is embedded into the hardware, and sometimes it is just the opposite; but no matter how it creates a final device we see the result as a single device. In reality, it is all about the way we understand and modify the pure data in various forms. Thus all of the devices need data in some aspect to work.

It is important to see the thin line between software and hardware. From different perspectives they may seem to interfere with each other's domain but it is not hard to differentiate each other when the definition is made within a specific context.

Software is the program and other operating information used by a computer. It is the encoded instructions, usually modifiable, and they are used to direct the operation of a computer. In computer science, software is everything that you can load on your computer, from the simplest operating system to the game programs. It is the routines and symbolic languages that control how the hardware functions (Oxford Dictionaries) (Cambridge Dictionaries Online) (Dictionary.com) (Yahoo Education) (Vocabulary.com).

The software, as a term, is invented and used for the first time by the statistician John Tukey in a 1958 article in *American Mathematical Monthly*. He was the first person to define the programs which the electric-based (electronic) calculators ran. It is an important fact that this was about three decades before the founding of *Microsoft*. He also mentioned that software is "at least as important as" the hardware of the day tubes, transistors, wires, tapes and the like. It is also interesting to see that he made a definition for a binary digit twelve years before his definition of software. His definition of the term "bit" is still the base of all computer programs and most of the digital technology (Leonhardt, 2000).

As it can be understood from the definitions above, software is mostly related to controlling the operation of the device - the computer, which computes. But how does a computer work?

> Computers use integrated circuits (chips) that include many transistors that act as switches. These transistors are either on or off. Electricity is either flowing through the transistor or it isn't. Thus, a circuit is either open or closed. Something that can have only two states is called binary. The binary number system represents the two states using the symbols 0 and 1. Actually, there are no 0s and 1s inside the computer. Instead, the 0s and 1s represent the state of a transistor switch or a circuit (White, 1994, p. 26).

Basically, it is a very simple process. The first step of programming comes from controlling the bits, and creating a "binary code" which floats in a sequential order to control the physical parts of the computer. This floating series of bits is called a bitstream and it is the pure form of digital data that today's devices can understand. One of the earliest forms of usage of the bitstream is the Turing machine which works in a similar logic.

> Turing machines are not intended to model computers, but rather they are intended to model computation itself; historically, computers, which compute only on their (fixed) internal storage, were developed only later (Chen, 2008).

The fully-mechanical device, Turing machine, is only a physical starting point to the basic logic of how a computer actually works. The importance of Turing machine is that it illustrates the fact that the "computation" process is a fully physical process. Digital technology is not very different than the old Turing machine. It is the same physical logic behind the whole system, all related to bits and bitstream, to enable or disable certain switch.

I do think it is extremely important to notice this procedure as a fully physical procedure since it is what makes the whole "software" work as it does today. It is the story of how electrons travel from one point to another with a carefully made micromanagement. At the base level, technically speaking, lowest-level programming, it is not that much different than opening and closing the switches on the board with hand. This is one of the reasons what makes software something operational rather than something directly virtual.

Apart from all its technical parts, software is a logical operation. It is how we control the hardware made of metal silicon and plastic, it is how we get the power over the dead body of the physical machine. I am referring the hardware as a machine, since both of them, the machine and the hardware, are very alike in many senses, even from a mechanical point of view.

John Tukey was right about his argument about the importance of the software; maybe he was even more right than he thought. Today, we know that hardware production is at its maximum compared to what we have seen until today. However, when we check the top companies in the world, it is *Apple* and *Google*. *Apple* is making consumer electronics and *Google* appears to be the business of information. However, they are making something else, and apparently what they're doing is so vital that today they have the power to affect the whole global economy with a slight move. It is also important to count *Facebook* with these companies as well. Despite the logic behind what they actually do, it is also coming from the very same base of understanding; compared to *Apple* and *Google* (Gralla, 2010).

It is not hard to see that this "something else" is software. It is such a software that we already accepted all of them as a part of our daily lives, and we're happy with going along with them. Lev Manovich mentions this as a *cultural software* – cultural in a sense that it is being used by massive amounts of people, hundreds of millions, and it carries atoms of the whole culture, the whole media and information; as well as all sorts of human interactions with this data – which is only the top part of the iceberg that is the visible part of a much larger kingdom of software (Manovich, Software Takes Command, 2008, p. 4). Manovich also points that software is a universal method of operation, just like language, within the following lines:

> Software is the invisible glue that ties it all together. While various systems of modern society speak in different languages and have different goals, they all share the syntaxes of software: control statements "if/then" and "while/do", operators and data types including characters and floating point numbers, data structures such as lists, and interface conventions encompassing menus and dialog boxes (Manovich, Software Takes Command, 2008, p. 5).

Language, is one small magical word here. Software has a language and it is a high or low level language depending on how many steps it is far away from the roots of the digital machine: the bits and the bitstream where everything is almost manually operated by the low level software. I'm referring this as an almost-manual process, not a directly automated process since the automation lies within the way programming works but the actual knowledge behind all of this work is a manual operation of directing the army of bits within the bitstream. For sure, most of the high level programming languages of today are generally not in any sort of relation with the bitstream directly but it is still a matter of fact that this automation is not much different than a hierarchical manual work.

For this reason, the language is one of the most important keywords of the software. It is how we direct the software, it is how we command the bits and it is how all of the operations are executed. This is the power that takes the simplicity and adds up to itself until it becomes complex enough to control more with less.

As Alan Kay, the American computer scientist and one of the pioneers of object oriented programming, shares in one of his *Ted Talks* (Kay, 2008) "One of the things that goes from simple to complex is, when we do more" and he uses the metaphor that Murrah Gell-Mann used about Architecture, and how things can become complex starting from a simple unit, a brick. It is all about finding a good combination to build something new.

Just like the architecture that uses bricks and language that uses the complicated structure of words and meanings, the whole software logic is building on top of each other. As the software languages gets complex and the structure gets deeper it gives us the ability do more with the same repetition but in a different order and logical understanding.

For this reason, the software is dependent on the base it has been built on top of, and therefore it is hardware-dependent. In computer science, computer is the main hardware so all of the software is related to the original hardware, the magical apparatus, the computer itself. However, when we look out of the boundaries of the computer science and information technologies the software is the logic of the operational act.

Apart from the mechanical Turing machine, Theo Jansen's kinetic sculpture project is a modernized proof of how software can be based on something other than digital and electronic equipment (Jansen, Strandbeest: Leg System, 2010). Jansen created these amazing "life forms" that he calls kinetic sculptures based on plastic yellow tubes with a mathematical formula and he programmed them to survive on their own in public beaches.

The surviving operation is a simple logical operation but it needs the input of the nature and the decision making of the artificial beasts. Again, from plastic empty tubes, Jansen's creations include members (which he calls, the "feeler") to feel the water, earth or the wind around the beast.

His beasts also have their brains that are made of plastic bottles and air pressure that comes from mother nature. The brain can do the basic binary counting and with the help of the clever usage of this simplest operation it has the "imagination of the simple world around the beach animal" as Jenson speaks in his presentation (Jansen, Ted Talks: Theo Jansen: My Creations, a New Life Form, 2007).

Jansen's beach animals are complete machines with their software and hardware parts, which makes it easier to distinguish the software from the hardware, the logic from the physical object in this sense by looking at his designs. "The software is what operates the animal" in a sense he speaks, and it is the element that determines the function of the apparatus he created.

Figure 2.1 - Strandbeest of Theo Jansen.

At this point, it is still impossible to distinguish if software is a supporting element to the hardware or vice-versa, but it is still important to see the fact that hardware is meaningless without a software and a software is meaningless without the hardware. It is quite like a body without a brain, or a brain without a body.

In digital technology it is even easier to see the need of custom software for every different hardware (or just the opposite) since the relation between the software and the hardware becomes visible as a part of computing terminology. The "hardware architecture" as a term, is not very different than how we know architecture; it is -again- related to the systems' physical components and the relationships between components. So, each different architectural hardware model is a different technical identification (of the whole physical system) therefore they all need a custom way of operating the internal parts, therefore, a "custom software" is needed to make the hardware function properly (Malek, 2002, pp. 13-17).

This is where the terms "system software" and "operating system" jumps in. After this point, it is not hard to guess behind the curtains; what they do and how they work.

Modern general-purpose computers including mainframes and personal computers need (and have) operating system to run other programs, such as application software. The base (lowest) level of an operating system is its kernel. Kernel is the first layer of software that is being loaded into the memory when the whole system is booting or starting up. It is the system that provides access to various core services to all other software on the computer (like disk access, access to other hardware devices, memory management, task scheduling, etc.) (Mamčenko, 2008, pp. 5-6).

Software in this notion is the logic of operation about how the hardware should work. It is like a mechanical connection between this logic and the physical device that is connected to it.

After all, the whole logic of those digital devices and operating systems are quite like the idea of "devices inside devices" or "devices that control other devices" to create something complete. In reality, all of those devices, the sum of all hardware and software that is connected together, form a single unit that we can refer to as a machine. In this case, software is the extension of the hardware and vice versa. Both of them are completing each other since the hardware needs the operating system and additional software to operate whereas without the hardware it would be impossible for the software to even exist. The hardware and software in this notion should not be seen as things that are completely separate from each other but rather they should be considered as parts of the same machine: The hardware being the sum of all tangible members and the software being the sum of all non-tangible members.

2.2. The Essence of a Virtual Machine

The whole conceptual evolution starts with a very small step: "virtual". But what is virtual? Gilles Deleuze uses the term virtual to refer to an aspect of reality that is not actual, but nonetheless real. An example of this would be the meaning, or sense, of a proposition, which is not a material aspect of that proposition (whether it be written or spoken) but is nonetheless an attribute of that proposition. However, the real which is realized from possible (as a result of realization process) is in the image and likeness of the possible it realizes but the actual does not resemble the virtuality it is connected to. We arrive to a virtual as a result of a virtualization process beginning from an actual (Deleuze, Bergsonism, 1991, pp. 96-97). In this notion of virtual, a virtual machine is a machine which is completely real but not actualized.

The general description of a virtual machine within a technical context is quite similar in relation to how it is virtualized.

> A virtual machine (VM) is a "completely isolated guest operating system installation within a normal host operating system". Modern virtual machines are implemented with either software emulation or hardware virtualization. In most cases, both are implemented together (Wikipedia).

Both of the definitions are related to the attribute(s) of the virtual since both of them have the relation to the actual which we arrive.

When we use the term “virtual” today, it somehow pulls the word “technology” from the web of meanings and behaves as a carrier that corresponds to almost anything related to the computer technology which covers almost every part of our daily life. We are way more closer to the functional meaning than we think!

Virtual machine is simply a process (or an application) that runs inside of the hosting system (in this case: the operating system) which is created when the process is started and destroyed when the process is stopped (Smith & Nair, 2005).

The word “machine” that is used in the computer terminology does not signify a physical object. A machine, is a representation of computer or computer software (which is already arguably virtual). So, the technical description of virtual machine is quite like a computer running in another computer.

Host and guest are also the two terms that explain the relation between levels in software. Lower level (base) systems are hosts to the higher level (content) systems. For this reason, guest systems are generally higher in level or virtualization.

The whole concept is directly related to the virtualization process in computing which is the creation of a virtual version of something actual, such as a hardware platform, a storage device, an operating system (OS) or network resources (VirtualBox, 1999).

Literally in all devices we know the whole process of virtualization in a digital environment is present. For example, when we connect a physical hard drive into our computer, it is a single actual drive. But with the usage of “partitioning” the drive, we can create multiple drives within a single physical unit (like C, D, E). With the help of this, without multiple physical devices we can make use of separate virtual drives as we need. This type of a virtualization is a simple and effective method of using the physical hardware. It other words, a partition is a logical division on the physical drive that is not actually divided into various parts. A logical partition (commonly called a LPAR) then becomes a subset of computers reachable hardware resources. As a result, a physical machine can be partitioned into multiple logical partitions and each can host a separate system (Singh, 2009, p. 73).

Another simple example is the whole “channel” system that we use in our daily lives. TV channels, radio channels, or the more complicated digital channeling into the “bandwidth” of the data are some other examples of virtualization. With the help of selectively limiting the signal frequency spectrum different channels exist. This is a very simple operation of virtualization compared to the utterly complex virtual systems that are being used in today’s Information Technologies (IT) (Lozano-Nieto, 2007, pp. 3-15) .

Today, virtualization as a computer term, is used to describe the process of creating multiple virtual environments under the same physical device to save power and physical space on complicated databases or any varieties of digital data. We can also simply say that most of the digital devices of today uses various levels of virtualizations. A virtualization process can end up with almost anything that is virtually (logically) existent.

Another similar concept is emulation. Emulation is the ability of a software or any sort of computer program in an electronic device to imitate (emulate) another device or a program. (Koninklijke Bibliotheek, 2009) In this sense, an emulator is a software or a hardware (or both at the same time) which can duplicate the operations and functions of the first system to the second one. Technically, the first system which is being duplicated is called the guest, and the second system is called the host.

It is a different concept than simulation: A simulation is a virtual form of performance and operation whereas an emulation is a complete imitation of the whole system. A computer simulation of an “earthquake” or a “melting ice-cream” is simply not an emulation, but re-creation and/or re-calculation of the original situation to imitate the behavior of the “actual”.

In other words, simulation is the imitation of an existing state, operation or behavior whereas emulation is the duplication of the device and how it works. In this sense, a simulation behaves similar to something else but it is implemented in an entirely different way but an emulation is a system that behaves exactly like something else and abides by all of the rules of the system being emulated. Both are abstract models of an existing system. (S.Robins, 2012)

> Emulation involves emulating the virtual machines hardware and architecture. Microsoft’s VirtualPC is an example of an emulation based virtual machine. It emulates the x86 architecture, and adds a layer of indirection and translation at the guest level, which means VirtualPC can run on different chipsets, like the PowerPC, in addition to the x86 architecture. However, that layer of indirection slows down the virtual machine significantly (Caprio, 2006).

Virtualization, on the other hand, is a way of using the same device or device resource (memory, hard drive, CPU, etc.) in a different way of logical abstraction. It is not simply imitating the function or the operation of something else but it is a way of performing a logical separation between the objects that are virtualized.

> Virtualization, involves simply isolating the virtual machine within memory. The host instance simply passes the execution of the guest virtual machine directly to the native hardware. Without the translation layer, the performance of a virtualization virtual machine is much faster and approaches native speeds. However, since the native hardware is used, the chipset of the virtual machine must match (Caprio, 2006).

An emulation is used to replicate existing complete physical or non-physical devices (systems, or subsystems) that are absent in our current system. This is an operation of tricking the software of the computer, just like putting a fake working imitation of what does not really exist. The imitation is fully functional just like the actual object and there will be no way for the software other than the emulation itself to know that what they are interacting with is a copy but not an original. The main usage of this tricky operation is to imitate (emulate) the non-existing hardware so that it becomes possible to run the software that are in need of the absent hardware in the system.

Another explanatory example is the "Atari Emulation". If you want to play your old Atari games on your current computer, you need a way to run the game. This is not as easy as it sounds, since everything is software and software has various types, structures and rules. It is like trying to run a vinyl recording on a CD player. They both have totally different structures, they both have totally different physical parts to support their job. Your Atari game will not run on your Windows or MacOS based machines since the machine does not really know anything about what the game actually tells to the hardware since even the hardware is not the same with the Atari device that the game was meant to run on. In this case, an "Atari Emulation" would solve the problem. It would create a virtual environment inside the operating system (Windows, MacOS, etc) and simply trick the game when it is run. The game would not even know that it is running inside the emulation. If there could be a way to ask the game about how it feels about the Atari device that it is being used on, it would probably tell you that the device is just as it had known before, perfect, actual and without any problems.

Please notice the term, “virtual environment”. That is a widely used technical term for the virtualization process for the created non-physical environments that run on the same device.

Virtualization as a process, is the process of creating a virtual representation that has a logical connection between the function or state of the actual object whereas an emulation is a virtual replication of an existing technology which gives the same output with the original object as a response to the same input.

Today, when the “virtualization” or “virtual machine” terms are used, they do not have constant meanings any more. As information technology is on steroids since the wide public use of Internet, these terms are constantly changing their meaning as part of technology and technological needs. Today’s virtual machines are complex applications for certain technical purposes and thus the virtualization process has become something else then where it had started.

As a result, today, virtualization is the act of making some effect or a condition virtual on a computer. It is a layer of logical abstraction between the actual (hardware) and the virtual (software). Emulation is the act to imitate another device or program. In both cases, some of the results may become virtual machines (if complete) and some may just stay incomplete and still be used as parts or subsystems rather than complete systems.

After some point, the terminology and the technical information becomes so important that the virtualization itself becomes a completely new and unrelated term to its origins. As mentioned above, a virtual machine as a technical term in the context of IT, is like running a computer inside another computer. It is very much like a "$virtual^2$" (virtual square) where the mentioned machine is already a virtual platform (like an operating system), which turns a virtual machine into something close to a "virtual operating system". However, the whole concept of virtual machines is very similar to the way the computer programs work in relation to real life where the actual objects and tools can be re-created within a virtual environment and still be used. Since computer-technology does not really care about the "real-real" things, the daily usage of a virtual machine is not exactly the same as the technology term of today. However it serves as a good bridge to understand the essence of the relation of computer programs or the software with the actuality. Please note that virtualization as a process (transforming the actual into virtual) and virtualization as a technical term (information technologies) are almost completely unrelated today. From now and on, for the purpose of this thesis, virtual machine will refer to “virtual” machines, as in the description of “virtual” being opposed to “actual”; but it is still important to know the origins of how it became a part of everyday computer technology and information technologies since it is required to keep the connection on a certain level.

2.3. The Pledge: Virtualization Process

Virtualization can be a simple or a complex process depending on the context from which it is approached. Technically speaking, it is one of the essential processes of today that is used to save power, physical space, and money.

The whole idea behind the technical manipulation is to create multiple logical devices within a single physical unit, each of them working separately just like virtual machines. The simple example could be the way hosting companies work, or the way your email is placed in virtual space.

When a user registers for an email address, he receives a "mailbox" which is completely personal, and which has a storage limitation. The mailbox is unrelated with the other mailboxes in the same service provider. It is very much like a separate personal space without any connection to the other content around it. However, in reality the "machines" of the server store multiple mailboxes within the same physical device but the users have no interaction with the rest of the content since it is a virtual space that is given to the user as a "mailbox".

The core idea behind this approach is to save space and processing power. The storage limitation that is given to the user is just a numerical value of a virtual disk space, and it is not directly an "empty storage space that is isolated from the other spaces" or saved in any way. This is more like an "access to the storage space" for the user. When compared to a physical mailbox system which occupy physical space even though it is empty or completely full, this virtualized approach is different in this notion. In other words, if the user is not using the whole mailbox's storage, than in reality the user is not occupying any space for the unused space that (s)he still has; but only occupying the space that is being used.

This is a very clever usage of the digital space since normally users would need a separate device to store their data and the empty space of those half-used storage devices would be occupied although there is no data reserved in them. Hosting companies also work in a similar manner when giving virtual space to the registrants. The whole approach is a way of saving space, and giving "more access" to the users about the existing services.

Virtualization, as a logical process is about how something becomes virtual from actual. As an example, an abacus is a very old device, a discovery to make fairly simple calculations. It is simply a device with a function. If we skip the fact that the concept of a number also includes a level of abstraction through the structure of language-meaning, an abacus is a physical system to do simple calculations.

If this is the case, a calculator is another device that has a similar function with an abacus. A calculator needs the abstract essence and the logic of an abacus as a starting point, but needs a different logical structure to work and to be executed. It is another physical device with one more level of logical virtualization (through the usage of zeros and ones and digital technology) which can still be taken as a separate device, or a separate "object" with a similar function.

In this case, again, an image or a photograph of a calculator can be counted as a signifier of the device itself. Everything seems fine till this point, however, if this image could do the very same mathematical operations, in other words, if it has the very same function, could we still call it a representation ? Or would it become something else ?

Today, thanks to technology, every one of us has a software called calculator on our computers or smart devices. It is an easy-to-understand concept of a virtual machine and how a virtual machine works if we keep everything connected to each other. The calculator software is not only a simple image or a visual representation of the device, but also a fully functional working device itself. It has all of the functions of the conventional calculator, which makes it a virtual and fully functional machine within the boundaries of the concept. Is it a virtual object without a body, or is it something totally new and different that does the very same thing as the original?

In Ted Nelson's "virtuality" definition, which is also the dictionary meaning of virtuality since the 18th century, it is argued that, everything has a reality and a virtuality. According to Nelson, we can also talk about the two aspects of the virtuality: the conceptual structure and the feel. "The conceptual structure of all cars are the same, but the conceptual structure of every movie is different (Nelson, Ted Nelson's Computer Paradigm, Expressed as One-Liners).

The conceptual structure of the computer software is the important part here. The feel of the software can be fine-tuned, just like the feel of a car, depending on how we use the user interfaces. However, the concept remains the same. If we think of only virtual machines (not the widely defined software), it looks like we can say that the concept of the virtual machine is defined by its function since the function is the only reason of that particular software to exist. After all, a virtual machine is “an efficient and isolated duplicate of a real machine” (Popek & Goldberg, 1974).

The whole virtualization process is an abstraction in reality, just like the radio channels, or the way disk partitioning works. It is not a simple abstraction of only the function, but everything related to the actual representation.

There are also various levels of abstraction for software in a certain hierarchy, lower levels being implemented into the hardware itself, and higher levels are in software only. In the hardware, all the components are physical with real properties, and physical definitions. In the software levels the components of abstraction are logical, and have no physical restrictions at all.

A software calculator is a good example of “abstracted and re-invented” version of a physical electronic calculator in this sense. An electronic calculator’s software lies in its electronic parts and how they are combined together on the physical chipset. This combination of electric engineering is what makes its software work. On the other hand, software calculator of today has no physical parts at all.

> Computer software is executed by a machine (a term that dates back to the beginning of computers; platform is the term more in vogue today). From the perspective of the operating system, a machine is largely composed of hardware, including one or more processors that run a specific instruction set, some real memory, and I/O devices. However, we do not restrict the use of the term machine to just the hardware components of a computer. From the perspective of application programs, for example, the machine is a combination of the operating system and those portions of the hardware accessible through user-level binary instructions (Smith & Nair, 2005).

According to the definition of Smith and Nair it may be possible to talk about various levels of virtualizations (or abstractions) in a computer system. Starting from the base point, the bitstream, and going up to the higher levels of software it is all -and should be- related to each other since they are using the same operational body. In this sense, the concept of those various virtualizations are the same (the operation) whereas the feel of them are just non-absent in certain areas until they are directly interactive to the user. In this case, the "Graphical User Interface" of the software is what carries the feel, just like the feel of the car.

At this point, we can possibly say that if a picture of a calculator is a representation of the object itself (and the meaning connected to it – in this case, the "function" of the calculator: to calculate) whereas a calculator as a virtual machine is a representation of a real and fully functional calculator device, with all its functions and abilities. At the same time, it can do what a calculator can do, so it is a calculator by itself. As a part of this context, the virtual calculator is not a simple representation but a working copy or a complete functional imitation of the original.

There is always a reason to create new versions of functional objects, just like creating something that takes less space or creating something that can do more with the same or similar function. A modified bicycle may be better in using the applied force, or a better designed spoon may be an easier object to mass-produce. However, the virtualization process in this sense is not only a copy of the original object but also the abstraction of the original object in long-term. This “discovery” becomes more and more useful as the new object becomes a part of our life.

In this notion of copying process, the model is the original object whereas the copy is the virtualized version. Depending on how the actual process works, it is possible to say that the imitation is no longer imitating the model. Those reasons force the imitation to become something that can do more than the original version in most cases. The imitation becomes another original, or something else. Therefore, in long term it destroys the need to the original object in the structural existence of a “tool” – which is related to the reason of the tool to exist: The function.

It may also be possible to create a connection from Deleuze’s consideration of cell animation as a part of cinema with the current topic, the abstraction.

> If it belongs fully to the cinema, this is because the drawing no longer constitutes a pose or a completed figure, but the description of a figure which is always in the process of being formed or dissolving through the movement of lines and points taken at any-instant-whatevers of their course (Deleuze, Cinema 1: The Movement Image, 1986, p. 6).

Although the subject of cinema and the subject of virtualization is completely different, it it may still be possible to talk about the slight connection here. It is the abstraction itself. The abstracted can – and does – become something “else”, or may represent something else, even the new version of itself could be the represented object.

In order to deal with virtual and actual, Deleuze proposes to change (replace) the real-unreal or true-false opposition with the actual-virtual. According to Deleuze, both virtual and actual are real, but not all of the things that are virtually contained in this world are actual. In other words, the “virtual” (the memories, the dreams, the imagination, etc.) can affect us, therefore, it becomes real. (Deleuze, Bergsonism, 1991, pp. 96-97)

An actual desktop is a physical working space. Each object on the desktop has attributes, functions, a certain look (or physical body in this case) and a reason to be there. We also have a virtual space called desktop on almost every smart device and all types of personal computers. It can be named differently such as the working space, the pane, the desktop, or anything however the purpose is the same: Using it as the working space, the command center. We also have our virtual tools for use on our desktops (in this case, technically only shortcuts to the virtual machines), which are there for defined reason(s). Simply, if we are writing a lot, we can have our favorite text editor application on our desktop for easy reach – just like a pen and paper or a typewriter on an actual desktop. If we need a calculator too often, we can have the virtual calculator somewhere close to us so that we can click it as quickly as possible. This sort of a representation of a desktop is not only the representation of the actual desktop which physically does exist, but also an easier-to-use tool for the virtual environment. You could never delete a pile of documents with a simple finger movement in reality!

At this point, we can say that the virtual desktop has functional and conceptual structure of a real desktop but it is impossible to say that the virtual desktop is a copy of the actual desktop or it is there to become a copy. The important point here is the ease of use and improvement. The virtual “desktop” is a new “tool” (maybe even a new toolset) with a conceptual connection between the real one. Is this some sort of a reconceptualization of what already exist?

Another example would be the physical typewriter which needs paper and ink to work. Apart from all of those physical needs, it is impossible to delete the typed content without leaving trace on the final output. On the other hand, a typewriter application would simply do all of those plus deleting what was typed. Editing the certain area within the text is just a few clicks away from the user. Just like *Microsoft Word* which became one of the universal digital tools for text creation and editing, with a lot of additional functions that assist us in many different tasks. Looking from this point of view it is possible to say that the application which started as a digital copy of the tangible typewriter became something else, which has a lot of functions as well as the typewriting ability.

By going back to the calculator topic once again, it is possible to show the steps of transposition of the functional ability.

1. Counting is an operation.
2. Abacus is a basic operational tool.
3. Abacus is an object that can do calculation (add).
4. Calculation is a function.
5. Calculator is a functional object.
6. Picture of a calculator is an image-based signifier.
7. A software calculator is a signifier of the original.
8. A software calculator is a functional virtual object.

The signifier becomes the thing that it signifies (maybe even more), and it starts to change the meaning of the signified that it is supposed to signify. Maybe twenty years after today, no one will remember the good old physical calculator with all its buttons and plastic body, but they will remember the virtual machine, the piece of small software.

This will literally turn the virtual machine into the object that it is supposed to signify with the help of carried functions and attributes in its index.

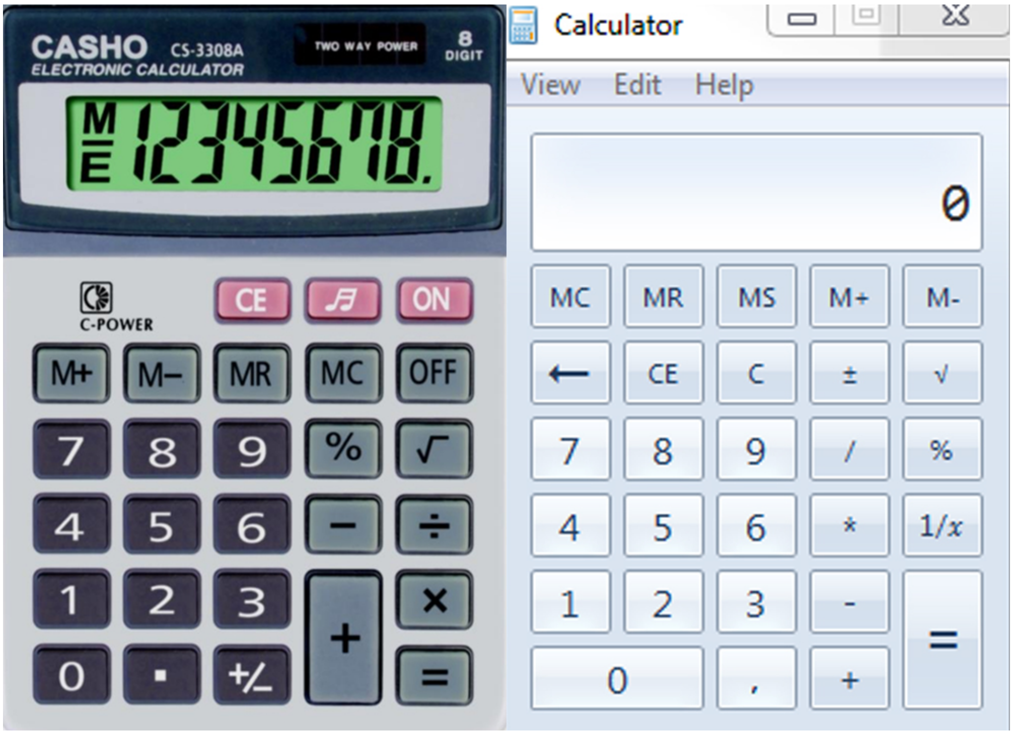

Figure 2.2 - Physical calculator vs. virtual calculator.

This sort of a transformation is very similar to the transformation from the old rotary dial telephones to newer digital telephones which almost does the same thing in a different technical route. The "function" is the thing amplified and carried over species in each new step. Both of those devices are telephones but the whole logical approach of how they work is different. By noting down these lines and going back to the virtual object, can we talk about a type of re-invention of the represented here?

If we consider the starting point, the imitation of the original, as how the representation works in this transformation process; it becomes easier to consider the result, virtual object, as a simulacrum since in reality it has no connection with the original object any more, which makes the virtual representation a copy without an original.

It is possible to ask Deleuze's question, the "question of making a difference, of distinguishing the thing itself from its images, the original from the copy, the model from the simulacrum" in a sense that the virtualization process creates copies which are different enough to know the difference yet same enough to replace the original (Deleuze, Logic of Sense, 1990, p. 253). The functional capabilities of the virtualized original (the copy), which are the reasons of the original to exist in the first place, remain at least as much as the original. It is not a degraded version of the original object since it has the same functional properties (sometimes more) and the essence of the first object.

> The simulacrum is not a degraded copy. It harbors a positive power which denies the original and the copy, the model and the reproduction. At least two divergent series are internalized in the simulacrum— neither can be assigned as the original, neither as the copy.... There is no longer any privileged point of view except that of the object common to all points of view. There is no possible hierarchy, no second, no third.... The same and the similar no longer have an essence except as simulated, that is as expressing the functioning of the simulacrum (p. 262).

It is also possible to argue about the new system that those virtual objects bring into our everyday lives not only the technical system but the system of meanings and their relations, more importantly how we -users- relate to this new system of functionality and technical approach. This new structure which takes its origins from the existing objects is an actual transformation of function and operation from a physical notion of object to a non-physical notion of object.

In this context, if we think the hardware (the processor, hard disc, and everything required to "run" our virtual machine) is just a hosting body for everything inside it, and understand the fact that none of the software have real physical bodies, a computer program is a real object that does not have a physical body. It might even be possible to say that it is not virtual, but actual, since virtuality is the way of its existence in the actual world. It simply is not a copy, but a false copy in the Deleuzian sense. The definition and how it evolves over time make the virtual object seem like a copy however it is something completely new from how it works (physically) to what it does (functional). The hosting body of the object, the hardware, is what remains as actual extension of the virtual object. In other words, if we look closely, the whole concept is not simply the re-contextual version of what already existed, but more like an extension, maybe even a shift of actual into virtual and vice-versa.

CHAPTER III

APP SOCIETY

3.1. What is an App ?

When the word “software” is used, applications are what most people think of. They are the kind of software which run on top of the platform software – the operating system, device drivers, firmware, etc – and they are the kind of software we, users, love to learn and play around. An application (or an app, as a shorter version) is a type of computer software that is created to help the user perform special tasks, and these programs are designed for the end users. They do not interact with the computer hardware directly (at the basic level) and they reside above all sorts of system software and middleware.

It is easier to start by defining what an application is not. For this reason, it is also important to know the difference between the application software and the middleware, which is like the "glue" between system software and the applications. The middleware is somewhere between the operating system and the applications, and it acts like a bridge connecting both sides to each other. Middleware is a virtualized communication tool between the operating system and the software developer whereas applications are created directly for the users.

> The term application refers to both the application software and its implementation. A simple, if imperfect, analogy in the world of hardware would be the relationship of an electric light - an application - to an electric power generation plant - the system. (Science Daily)

An application is also not a utility software, which is a special type of software that is designed for configuring, analyzing or optimizing the main computer. A Utility software is similar to someone who helps you to get everything tidy as you are working with the applications. It is the kind of software that will focus on how everything (hardware, operating system, applications, data storage, etc.) operates. Due to what they do, utility software is generally targeted at people with more technical knowledge about the computer. All sorts of anti-virus software, backup software, file managers, disk compression or disk clean-up tools are part of this kind of a software. A single utility software is often called a utility, or a tool.

After all, an application software (or an application) is any program which is not a utility, a middleware, or an operating system; which leaves us a wide varity of software with all sorts of purposes, expanding everyday with the social evolution of the computer and the whole software culture. A simple word processor, web browser, email client or an image editor are various examples for what an application is, as well as a specific tool for complex database managements or software development tools. It is an everyday apparatus for most of the people, and every one of us uses applications for one thing or another depending on what we need and how we are trying to achieve it (K.R.Rao, Bojkoci, & Milovanovic, 2006, pp. 22-27).

Maybe it is better to talk about various types of applications separately by categorizing them depending on their origins of existence instead of talking about all types of applications in a single continuous topic. Please note that the borders of these category descriptions are translucent since technology is evolving and it is filling the gaps faster with new devices and definitions. However it is still essential to make the initial distinction between those fluid categories to hive the notion of an "app" better and to explain why an app is different than an application.

By looking at the "desktop applications" it is possible to see that these are the applications that we use on our computers, on our laptops or any sort of device that can be categorized as a computer, and running a native operating system like *Windows*, *MacOS* or any distribution of *Linux*. Those applications are widely used from very general purposes to mission-specific actions, and they were the first generation of "application software" that have met with the end users – us (PCMag Encyclopedia).

Desktop applications are shortly what we use on our so-called desktop's of our computers. They are also called "standalone" applications, despite the fact that most of the applications of today are standalones. Like most of the applications desktop applications are platform-specific and they only work on the platforms they are built for.

On the other hand, Web applications are quite different than desktop applications.

> A web application is an application utilizing web and (web) browser technologies to accomplish one or more tasks over a network, typically through a (web) browser. (Remick, 2011)

When the definition is broken down into smaller pieces, it becomes more clear to understand what is going on in that terminological perplexity.

Browser is one of the key elements here. A browser, in other words, an Internet browsing tool (Like *Firefox, Chrome, Safari, Opera, Internet Explorer*, etc) is a software application that is used to reach information resources on the World Wide Web. An information resource is identified with the usage of URI[1] and may be any type of content, just like a Web page or an image. In other words, a Web browser is an application software that is designed to enable users to access and view documents and other resources on the Internet (Ian Jacobs, 2004).

A Web application is a type of application which uses a Web technology through the usage of the Web browser. A Web technology is a computer technology that is intended to be used across networks (just like HTML, CSS, Flash, JavaScript, Silverlight, etc.) (Remick, 2011). In this context the Web browser becomes the client of the Web application, where a client is used in a client-server environment to refer to the program the person uses to run the application (Nations, 2010).

From a technical point of view, the World Wide Web is a highly programmable environment that allows infinite customization of every kind of element; and today's websites are far from the static text

---

[1] Uniform Resource Identifier

and/or image only contents of early and mid-nineties. Modern Web pages use technologies to pull the content dynamically as the “output” according to the given input. Just like doing a Web search; the website is generated dynamically for the user depending on the existing data and content of the database connected to it. Apart from dynamic content, modern Web allows users to send/retrieve all types of data (personal details, registration forms, credit card information, any sort of uploading/downloading, login pages, shopping carts, etc.) through various Web technologies (Acunetix).

Web applications are, therefore, computer programs that uses Web technologies (generally through the usage of the browser) that allow the users to send/retrieve data from a remote database/host. The presented data is generally dynamic and it is rendered using the client’s (user’s) browser using a Web technology.

Most of the widgets (in W3C[2] sense) are good examples of Web applications, since they are packaged / downloadable / installable, as such, they are closer to traditional applications then a website content. However, today most of the websites are partly web applications by themselves and the boundaries are not strictly defined at all (Hazaël-Massieux, 2010). Those Web applications can be as simple as a weather widget or as complete and complex as a whole package like "Web based office applications" of Google (Spreadsheet, Typewriter, etc.).

Many of the applications that use social networks (like *Facebook* games) are also Web applications since they pull the data from social networks (like your *Facebook* friends list). Sometimes they are called "social apps" but the definitions are completely flexible at this point.

Web applications are very important in a sense that they are not platform specific since they do not directly work on the client machine but they are only visually rendered at the client machine by using the browser. For this reason, they do not need to be installed or executed,

---

[2] The World Wide Web Consortium (W3C) is an international community that develops open standards to ensure the long-term growth of the Web.

they are only displayed as a part of the browsing process and this makes most of the Web applications a part of today's Web content.

Apart from Web and desktop applications, one huge step is the technological evolution of applications into mobile applications. Current smart devices allow the users to do more compared to the earlier stages of the mobile industry, and this is where the whole revolutionary changes happen in a way that applications are perceived and used.

First of all, it is necessary to remember that a mobile phone and a smartphone (or smart device) are completely different things. The technical advancement may be considered as a continuation of the route (since they use hardware coming from the same origins) however the social impact is what makes a smart device valuable.

Combining a telephone and a computer was conceptualized and patented in 1972 (Paraskevakos & G., Apparatus For Generating And Transmitting Digital Information , 1972) (Paraskevakos & G., Decoding and Display Apparatus for Groups of Pulse Trains, 1971)

(Paraskevakos & G., Sensor Monitring Device, 1972) but the term smartphone appeared in 1997 with the *Ericsson GS88 "Penelope"* with a similar concept to that of today's smartphones (Stockholm Smartphone). One of the first smartphones which included PDA[3] features was the *IBM* prototype with the code name *"Angler"* and it was showcased as a concept product in *1992 COMDEX* (Smith R. , 1993).

The evolution of those devices which combine a digital system for operation (an operating system) and the features of a telephone was only the beginning of a huge step which lead most of the companies towards the smartphone scene of today. The important aspect of the smartphone (just like the earlier smart devices, PDAs) is the operating system since it is what helps the user to interact with the digital interface of the device and it is what keeps those devices alive in the scene.

[3] Personal Digital Assistant

The problem in these earlier devices was the content. Without the content, it is impossible for any sort of media device to survive, since those devices are there to be used. It is similar to the earlier battles between *Windows* and *Macintosh* where *Windows* almost took over the whole computer industry even though the *Macintosh* was far more advanced on many sides. It was the content, not the technical advancement of the device. *Macintosh* was expensive, just like the earlier smart phones, and it became meaningless for programmers to build something for this device since it has a very small market in the industry. However the *DOS*[4] (before *Windows* OS) based machines were widely being used as a part of business and industry so it was a lot meaningful to produce something that will run on a *DOS* based machine. That is the basis of how *Windows* became the popular and preferable operating system from mid-nineties until now. *Apple* had surely done a lot of changes within the last years that reduced the gap between the sales of *Apple* and *Windows* based machines (Pachal, 2012).

---

[4] Short for "Disk Operating System", is an acronym for several closely related operating systems that dominated the IBM PC compatible market between 1981 and 1995, or until about 2000 if one includes the partially DOS-based Microsoft Windows versions 95, 98, and Millennium Edition.

Because of the importance of the content, some of the early smartphones survived and some did not. First widely used smartphones in this sense was the smartphones which used *Symbian* operating system since it needed cheaper hardware to run (compared to the operating system of the day) and it was easier to develop software for that operating system. It had a file structure just like the desktop applications and it was far more advanced than the naïve operating systems that were included in the mobile phones of the day (I-Symbian, 1999).

What was being installed on *Symbian* based devices were applications that are platform-specific (only works on *Symbian*) but those application programs were very close to what we know as "apps" today.

The "app" as a definition is still quite different than all of the above. It is not necessarily a technological step but rather a part of the social evolution and how smart machines became a part of our social existence.

Before examining what an app really is, it is a necessary step to understand how Steve Jobs changed the whole mobile industry, and how he affected the whole technology in a way a "revolution" happens. This revolutionary change which has started with the first *iPhone* that was released in 2007 is still effecting the whole computer industry by redefining application.

> Every once in a while, a revolutionary product comes along that changes everything… In 1984, we introduced the Macintosh. It didn't just change *Apple*, it changed the whole computer industry. In 2001, we introduced the first iPod. And it didn't just change the way we all listen to music, it changed the entire music industry. Well, today, we're introducing three revolutionary products of this class. The first one is a widescreen iPod with touch controls. The second is a revolutionary mobile phone. And the third is a breakthrough Internet communications device. So, three things. A widescreen iPod with touch controls, a revolutionary mobile phone, and a breakthrough Internet communications device. An iPod, a phone and an Internet communicator. An iPod, a phone... Are you getting it? These are not three separate devices. This is one device. And we are calling it *iPhone*. Today, *Apple* is going to reinvent the phone (Jobs, 2007).

As Steve Jobs mentions in his 2007 Keynote, the *iPhone* is a device which is capable of doing multiple things at once not only with the help of installed software but the way it is designed. He also mentions that the device is the simplest device to use compared to all other smart phones "which are not smart enough to compete with *iPhone*" (Jobs, 2007) and the way *iPhone* is designed to work and the way we interact with the *iPhone* is the key element to its social existence. The *iPhone* did not only created a clever way to listen to music, browse the Internet and use the features of a mobile phone all at once, but it also introduced a lot of new technologies into the mobile scene. One of these technologies was the multi-touch screen which changed the whole way for the users to interact with the device from a complicated physical button-based interface to a fluid and graphically customizable clean and simple interface which only includes the interaction elements (such as buttons and sliders) only when they are needed within the software. This step is far more important than how *iPhone* is positioned in the social structure since this is what made it simple and easy to use in the first place.

When *Apple* introduced the newer *iPhone* at 2007, they've also made a big update to the connection software, *iTunes*, with a "great new feature: The "*AppStore*" as they named it (Apple, 2008). The release of the *AppStore* is what actually changed the whole understanding of the applications, or the "apps as *Apple* named them. This is the breaking point from the applications into the universe of apps that are cheaper, lighter and easier to use.

One of the differences between an application and an app is how it is positioned within the social structure; how they are named (and distinguished) transparently from the existing "applications" of the whole market. The apps are light applications that can be installed through an online market easily (in this case, the *AppStore* of *Apple*). They are generally priced with a 0.99 USD tag which is a lot cheaper compared to the usual desktop applications which scale up to hundreds of thousands of USDs in some situations. Also, the way they are categorized within a visual catalog is quite similar to going to the store and browsing between racks while asking the question "do I need something today?" whereas for application programs this is a meaningless case since they are necessary to perform certain tasks or operations and they are really functional and meaningful through how they are used.

This situation brings the functional need to an application back in question. All of those applications and apps have various functions depending on the context they are used. However apps changed the way users and developers approach to functions. Developers started to invent new functions along with the applications they have created. Users started to browse the racks of the digital store to learn and to discover new functions that they did not know that existed before. Those new functions are embodied into the virtual tools that multiplied everyday with the addition of new abilities to the application stores and as a result to the society. Some instances of such new functions are reverse image search, which was not possible before someone imagined and created the app; or the *Find Your Friends* app , that caused a lot of arguments about how it effects the social structure at numerous conventions, which enables users to see the exact physical location of their friends, wives, husbands or children on the map. Those tools that were absent before are results of this creative creation process of the apps.

Imagine a market which is growing every single day. *AppStore* has over 650000 apps total and developers are bringing over 10000 new apps every single week (Spriensma, 2012). It is all related to the mass-distribution system that *AppStore* -as the first application store in this notion- started: A distribution that is based on the popularity of the applications. Right now, no one exactly knows what brings the apps to the "top charts" of the store, since *Apple* is hiding the equation behind it, but one of the known variables of the equation is the application popularity which is related to the number of downloads and number of uses. It is also known that a lot more variables such as first use time and use frequency affect an application to be placed in the top charts. This approach to perceive an application as a popularized object changes the whole perception of how they exist in the software scene, since the applications that we knew before were there for specific tasks that we were aware of. For this reason, an application is commoditized by the mass use whereas an app is popularized by the mass use.

The app wave started with *Apple's AppStore* is now a necessity in smart phone market. Other operating systems like *Android* and newer versions of *Symbian* use their own software markets (*Android Market* for *Android* and *OviStore* for *Nokia* based *Symbian* versions) since it is necessary to create an active flow of app marketing simultaneously as they are developed. This flow also causes the developers to focus on these mobile platforms since more sales is always better for a software company in the mobile scene and reaching the devices means reaching the users in this situation.

The whole app scene and app based mobile industry took shape depending on how the applications are created and distributed in the following years. In 2010, the word "app" had so much impact that it was voted as the word of the year by the *American Dialect Society* (American Dialect Society, 2010).

It is also possible to talk about the idea of device customization that is created with this new playground of software. For a user, picking and customizing the attributes and functional properties of a smart device was never easier than now since it is a cheap and lightweight way of creating a distinction between two identical equipment.

Those mobile apps are also the part of our daily life and mobile Internet use, which opens a whole new level of interacting with the world by means of getting information and social communication over the extensive use of social networks. Steve Jobs calls this “your life in your pocket” in his *iPhone* release keynote (2007).

For the apps, it is also possible to get more input from various new input devices that are carried within the smart devices’ internal equipment - the hardware. For example, most of those smart phones have an internal GPS for location or an internal accelerometer and gyroscope for the software elements to reach and use when needed. Such interaction with the hardware would be quite meaningless for a desktop device since it is meaningless to change the orientation of the device or move it between long distances. These new hardware elements and device parts are reachable through the software by the developers, which enables them to get more data about the user and use that data as needed.

As an example, how we use a "map" on a digital device has changed since those new mobile hardware is added to our smart devices. It is possible for users to see their physical location in the world, directly by looking at the digital interfaces of their preferred "map app" and locate themselves exactly as they exist in physical space. This simple idea of geo-location, just like all other hardware based systems in those smart devices, is completely modular and it can be applied to any sorts of applications such as image sharing applications like *Instagram*[5] that shows the users' geo-locations on the public map when they share a photo.

Those mobile devices are also the part of today's new media or "metamedia" as Manovich calls it (Software Takes Command, 2008, pp. 85-90). Because, today the content is not only created for the desktop devices but also for the mobile devices. The statistics are informative at this point. Depending on the mobile Internet usage statistics, since that is where the metamedia is, it is possible to say that

[5] Instagram is a free photo-sharing program and social network that was launched in October 2010. The service allows users to take a photo, apply a digital filter to it, and then share it with other Instagram users they are connected to on the social network as well as on a variety of social networking services.

smart devices will be the main tool for metamedia. By looking at the statistics (O'Dell, 2010) (BBC News, 2011) (Alberts, 2012) (Rogers, 2012) it looks like somewhere between 2014 and 2015 the mobile Internet usage will exceed the desktop Internet use, which means the content will be made mainly for the smart devices. Just by depending on this simple information, it is possible to say that the way we socialize will be different than today, in the very close future.

At this point, it is possible to mention cloud based applications where the storage and the processing power are stationed at another physical location than the device itself. Cloud applications are the future of apps, as how it seems from today’s perspective. They are reachable from any location, they are as powerful and capable of desktop apps and they do not need any hardware to work on.

Shortly, the main difference between an application and an app is not in its technical aspect but in its social aspect. Technically they are similar to each other but the difference lies in how we perceive them. Apps are lightweight compared to applications and they are mostly used on mobile devices. But this may change in the near future as well since mobile technology is evolving every day and becoming more and more powerful just like the way personal desktop computers did from mid-nineties up until now. Apps are cheaper than applications as the whole business logic behind them is to make them affordable and accessible. With the help of this, they gain popularity and are widely used by many smart device users.

3.2. The Turn: Network Society

Starting from a very narrow definition and expanding itself into a social fact, the whole app concept is now one of the things that define the ways we communicate with others. As a result, one of the core subject matters of sociology – the social change, is being affected by these new ways of communication objects as they emerge into our daily lives with all their functions and possibilities. As a result, social life is changing dramatically and sociology is focusing upon the dynamics of macro-changes of what had been called “the postmodern condition” (Lyotard, 1984).

One of the most compelling names for these epochal changes was given by Manuel Castells: "The Information Age". The role of information and communication technologies in contemporary change is emphasized by Castells because of its strategic importance within the boundaries of social formation, as it is important to understand how members of a society communicate with each other to understand the society at large. He also mentions the ways of online communications as the part of this expanding infrastructure as well as declining prices of communication. As a result, online communications are developing "not as a virtual world but as a real virtuality integrated with other forms of interaction in an increasingly hybridized everyday life" (Castells, Rise of The Network Society, 2000, p. 29). According to Castells, computer-mediated communications and social relations in the information age are characterized by this complex network principle and social relations change as an effect of technological changes. It is a "technological transformation" of communication into an interactive network.

Both Manovich and Castells also mention hypertext as a step for understanding the evolution of technology within the social structure It is one of the main points of how we perceive the text, as it is the base of how we understand the language through methods of visual communication.

Castells starts with defining the alphabet as the new invention and innovation of 700BC which then became the mental infrastructure for cumulative, knowledge-based communication, and connects the whole idea with the formation of hypertext as a formation of meta language which is very similar to how the alphabet is formed as a way of perceiving information. This new form of text as a part of so-called "information superhighway" is vital to understand the new human-computer interaction since it changes the characteristics of communication (Castells, Rise of The Network Society, 2000, p. 356).

Manovich, on the other hand, refers hypertext as a particular case of hypermedia which is a form of media that is based on interactivity and connected media elements (Manovich, The Language of New Media, 2001, p. 57). His definition as a part of new media (and hypermedia in this notion) is later merged into his definition of metamedia – as a "fundamentally new semiotic and technological system which includes most previous media techniques and aesthetics as its elements" (Manovich, Software Takes Command, 2008, p. 60).

> Hypertext is a text which does not form a single sequence and which may be read in various orders; specially text and graphics... which are interconnected in such a way that a reader of the material (as displayed at a computer terminal, etc.) can discontinue reading one document at certain points in order to consult other related matter (Blustein, 2003).

Hypertext is a structure or a network connected to each other non-sequentially with the help of links (hyperlinks in this context). This forms a special sort of document which creates a huge feedback loop for the user. Currently it is everywhere in the computer medium, from Web to the simplest program. Conceptually we can say that this is a far more developed version of a regular "text" which can be typed out with the help of the typewriter.

Although hypertext has the same base with the regular text, it is a working network, a functional structure of many texts which forms a completely new way of reading and writing, which needs different tools and more importantly new ways of thinking/approach to create a hypertext, which makes it new and different from what a digital text used to imitate or signify. This is one of the simplest examples of birth of a new function. By re-organizing the data that already exist, it is possible to get more with a simple change of approach; which turns the end result to something else.

One of the early approaches of hypertext was Ted Nelson's *ZigZag*. Here is a short excerpt from the creator of hypertext's famous "trollout" text, on the new possibilities created by "indirect documents":

> For years, hierarchy simulation and paper simulation have been imposed throughout the computer world and the world of electronic documents. Falsely portrayed as necessitated by "technology," these are really just the world-view of those who build software. I believe that for representing human documents and thought, which are parallel and interpenetrating– some like to say "intertwingled"– hierarchy and paper simulation are all wrong. (Nelson, Indirect Documents At Last ! Now for a Humanist Computer Agenda, 2005)

It is also possible to talk about the *ZigZag* (software), as a “tool of visualizing the text” which is based on Ted Nelson’s data model that he designed for computer interaction between the users and programs. The overall design is focused on the structure of information which is called *“zzstructure”*. It is a similar form of “linked list” with a difference: It has connections in multiple axes (in all three dimensions). The information is visualized as “nodes” in all three dimensions, in empty space. They may or may not be connected to anything in that dimension. Overall, *ZigZag* is a way of displaying a special model of hypertext in a different visual arrangement so that it gives a totally different way of interaction and user experience. (Nelson, The ZigZag® Database)

It is possible to see the *ZigZag* and the *zzstructure* as a “tree based system” in a computer environment. Notice the term, “tree”, which is used for nested listing types. It is still connected to the original abstraction of a real tree and how it works/looks in the nature.

The whole hypertext approach is only a simple part of how we perceive, or forced to perceive the information formed as a part of “the alphabetic order" of Castells. It is the base of our interactivity with the World Wide Web, which takes the *HTML*[6] as it is starting point. It is possible to say that hypertext is how we communicate with our devices; in other words, it is how we communicate with everything related to our devices and today's IT; since all of them are evolving around the WWW and user interaction.

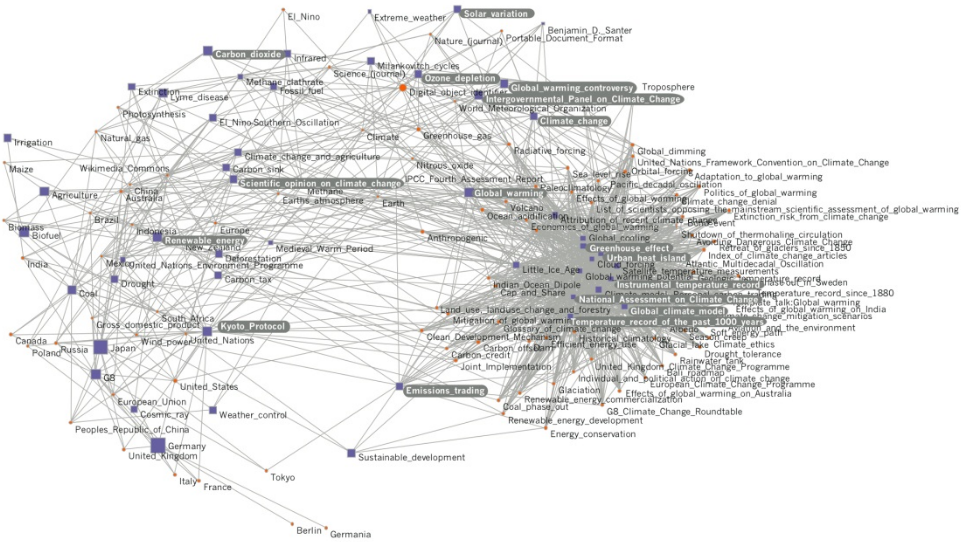

Figure 3.1 - Topology for "global warming" on Wikipedia.

The connections between hypertext elements on Web form an incredible structure of interactive data. Even a few levels of depth of

[6] Hyper Text Markup Language

those hyperlink topologies form huge webs of looping information and they can also be tracked to understand more about the behavior of the visitor. As a matter of fact, hypertext and text are incomparably different than each other.

Hypertext as a notion and way of user interaction is only a small step of our technological evolution within the social structure however it is a working proof of the concept that processing the same data in a completely different way is able to change the perception of the users, as well as the whole information culture.

> The emergence of a new electronic communication system characterized by its global reach, its integration of all communication media, and its potential interactivity is changing and will change forever our culture. (Castells, Rise of The Network Society, 2000, p. 357)

This hybrid form of communication, includes many forms of data in its complicated structure; forming a sea of "new media" as we know it today. It is hard to make the distinction between the steps, and where they change, but with the help of definitions it is at least possible to understand the boundaries of what new media actually is.

Depending on general definitions, it is not really easy to draw the outline of new media, however, it is easy to see an overall concept of what it really is. According to the definition, new media is a "broad term in media studies" which refers to "on-demand" access to a content at anytime, anywhere or with the help of any digital tool or interface. Moreover, it has interactive user feedback, forms of participations (communities and other virtual types of formations) around the content itself. It is directly related to all interactive forms of communication. However the definition of new media changes daily, and will continue to do so (Wikipedia) (Socha & Eber-Schmid, 2009) (Shedden, 2004).

The way we approach to new media is not very different than the way we approach to the hypertext as a social step. Also, according to the *Poynter Institute's* "New Media Timeline", it all began with the commence of digital development. As the processors and digital devices evolved, the term "new media" emerged and the need to new media became a part of the development pipeline (Shedden, 2004). The first event in the timeline is the experimental network of computers, called the *ARPANET* (which then evolves into what we currently know as the Internet).

Depending on these, it is easy to say that the new media is a form of how we understand and re-interpret the data in a multi-disciplinary way of creation.

The evolution of multimedia and new media is also a part of the effects of computerization on culture. The new media is not simply a tool for exhibition or distribution but it is a tool for communication itself. It is the change from printed press into a new cultural communication device, starting from still images and continuing towards a type of meta media that contains all sorts of moving images, sound, and various types of completely new structures.

According to Manovich, the way we use the software reshapes even the most basic social and cultural practices. These changes are such an important part of the cultural database that it changes the whole perception of the culture and makes us rethink everything from the beginning (Manovich, Software Takes Command, 2008, p. 17).

> In a software culture, we no longer deal with "documents," "works," "messages" or "media" in a 20th century terms…. We now interact with dynamic "software performances." I use the word "performance" because what we are experiencing is constructed by software in real time. So whether we are browsing a web site, use Gmail, play a video game, or use a GPS-enabled mobile phone to locate particular places or friends nearby, we are engaging not with pre-defined static documents but with the dynamic outputs of a real-time computation… Thus, the final media experience constructed by software can't be reduced to any single document stored in some media (Manovich, Software Takes Command, 2008, p. 17).

Manovich relates the starting of "new media" with the *"Analytical Engine"* device of Charles Babbage, which contains the key elements of today's modern computers. *The Analytical Engine* had punch cards for data inputting, the "engine memory" to store everything and a processing unit which Babbage referred as a "mill" to perform various types of operations to the data in the memory. The final output was being printed (Walker, 1998). Manovich also mentions that the device was not a new invention, but the idea was developed from an older design of J.M.Jacquard's "automatic punch card machine" which was a loom that could be controlled with punch cards.

Manovich is obviously referring the “start of new media” as something other than todays “digital” new media. However, the connection he points out is, the “data” and how it is processed, in other words, how the data becomes visible, through what series of operations. (Manovich, How Media Became New, 2001).

The tools of today give us enough possibility to send and receive the data in infinite possibilities with the usage of new media and meta media. The software which support these functional and operational acts are the basis of this automated translation. It is also possible to see the connection between the starting point and current point (non-ending point) of new media depending on the definitions made by Manovich, as the *Principles of New Media* (Manovich, The Language of New Media, 2001). These principles are: Numerical Representation, Modularity, Automation, Variability, and, Transcoding. These attributes also form the basic structure of apps and the whole logical approach behind them.

The other key element behind this new way of perception that started from the basic understanding of text and evolved into a complex structure is mobility. Since most of the apps are mobile or aiming for mobile-device usage, mobility is another important aspect of these apps of the digital age. Those consumables of the digital culture is highly effected by the mobile use and formed new ways of interacting with the mobility itself instead of only making use of it.

As an example to one of those uses is the new ways of “marking” the location, such as the location based massively multiplayer mobile games which enables a whole new virtual definition to the existing geography data for the players to interact with, or the location based social tools such as *Foursquare*[7] which enables the users to “check in” to venues or defined places nearby. Those newer ways of interacting with the physical environment through a virtual structure would be impossible without the evolution of apps and smart devices as well as the evolution of society’s perception over time. It is also a part of the

---

[7] Foursquare is a web and mobile application that allows registered users to post their location at a venue ("check-in") and connect with friends. Check-in requires active user selection and points are awarded at check-in. As of April 2012, there have been more than 2 billion check-ins with Foursquare. Users are encouraged to be hyper-local and hyper-specific with their check-ins-- one can check into a certain floor/area of a building, or indicate a specific activity while at a venue (TechCrunch, 2012).

cultural software that was defined by Manovich since it is reachable and used frequently by massive amounts of people. Today it is easier to see how these applications affect our life with the help of more complex communication systems such as online social networks which became an important part of our everyday lives.

Network as one of the key aspects of all those systems plays a big role in the transformation of software to "cultural software" Manovich describes. According Castells, a network is a set of connected nodes where nodes that exist as components of networks increase the power and value of a network since they absorb information (Castells, Communication Power, 2009, p. 20).

> Communication networks are the patterns of contact that are created by the flow of messages among communicators through time and space. The concept of message should be understood here in its broadest sense to refer to data, information, knowledge, images, symbols and any other symbolic forms that can move from one point in a network to another or can be co-created by network members (Monge & Contractor, 2003, p. 18).

The networks process the flows that are streams of information between nodes. So a network is a complex structure of communication which is based on the control of the flow of information in this sense.

A social network in this sense is a collection of individuals linked together by a set of relations (Buchanan, 2003, p. 45). It is also possible to see these human-based networks before twenty first century societies where the whole social life is built upon a structure of social information.

However today, the social networks are a part of the technological evolution process of the term, network. As a part of how we interact with the technology through the extensive use of those common devices, they are an important part of this social networking process. These communication devices not only transfer flows of information but also create a database of information which creates the whole logical structure of a digital social network, like cellular companies or online communities such as *Facebook*.

According to Castells, connecting the individuals is the key element of a social network since it is all related to the relationships of the smaller parts of the whole system, the nodes; and the real power as the most fundamental process in society is defined around the values of those relationships.

> Power is the relational capacity that enables a social actor to influence asymmetrically the decisions of other social actor(s) in ways that favor the empowered actor's will, interests, and values. (Castells, Communication Power, 2009, p. 29)

The social networking era started with a series of websites and "friendship" activity over the digital tools of communication. Sites such as *Flickr, Tribe, Friendster, LinkedIn, Facebook* were earlier examples. Most of these sites included creation of personal networks or social circles of individuals with mutual interests or common points. *Flickr*, as an example, connects people according to their common interests in photography and visual aesthetics whereas *LinkedIn* serves as a glue to the people within the same or similar business networks.

These common digital tools are now the base of World Wide Web and social networks. However it is also important to keep in mind that a social network by itself has no or very little power since it is all based on the information gathered from the members of that social network.

This is a semantic web, as originally named by Tim Berners-Lee, is a network which "provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries" (Berners-Lee, W3C: Semantic Web Activity, 2001) In other words, semantic web is a web which is developed using the resource description framework that consists of interlocking statements ("triples"). The semantic web is therefore, a network of statements about resources, which is able to connect data together depending on how it is processed.

The notion of social networking is built upon the semantic web idea. The most important aspect for a social network is the information that is referenced and shared through various formats and places, in infinite conditions and forms. A friend list may seems like a list of "close friends" for an individual but it can be a way to filter the people around defined ages for a company who wants to advertise a certain product to a group of people sharing common interests or geo-location. As a result, the main importance of a social network is how the data is processed (Berners-Lee, Web 2.0 Summit 09, 2009). Because of this reason, we can identify those cultural and social applications as the second key element since they process the data and reshape it depending on how it is requested. This means, applications are the elements which have real power over the data.

It is very similar to how data visualization works in terms of its logic. It is the re-visualization of an existing data, which enables one to see a different aspect of the data by looking at it in a completely different visual approach. Because of this reason, the whole logic behind data visualization could be a proof of the concept of the existing data-application system which works in a quite similar manner in terms of processing or re-formatting the existing information. Since "communication is the sharing of meaning through the exchange of information" the tools which modify the way data is transferred plays a vital role on the whole process. As a result "the process of communication is defined by the technology of communication, the characteristics of the senders and receivers of information, their cultural codes of reference and protocols of communication, and the scope of the communication process" since it is all about nodes that are contained within the network (Castells, Communication Power, 2009, p. 54).

Search engines play a huge role in the processing of data. They use every type of information to gather what is needed between colossal amounts of data to get "descriptions of identity" about what is requested. Before the semantic web the main problem was the "dictionary principle". In order to find a resource, the user must already know exact keywords within the "dictionary" of that location, which consists of the available words and keywords located in that digital address. However, after the semantic Web it became a lot easier to locate the data that we really request since every keyword is connected to each other depending on how they are accessed and processed so far. This network of resources, keywords, and their interconnections create another level of information over the existing one, which is the "relation of information" with each other. With the help of semantic Web, during search the user does not have to write the exact word to find the related resources about the word. The user can also input "related" words since at the background all of those related words are connected to each other as well as what the user is actually looking for through the use of semantic web processing. In other words the whole semantic structure is related to how the information is linked together (Downes, 2005).

It is also important to note the importance and value of an individual in an almost-alive structure like the semantic Web. The individual is responsible to behave as a node since (s)he has the power and ability to send requests (do searches, click links, etc.) or gather data using the whole network. Because of this reason, an individual is what really shapes the whole semantic structure of the Web; creating ties between various elements of the structure and causing the information to flow in a different way. For instance, the web page links in a social network are the elements of semantic Web that we know as the "weak ties", which are "acquaintances who are not a part of your closest circle, and such have the power to act as a bridge between your social cluster and someone else's" (Cervini, 2003). This means even the existence of an individual changes the whole semantic web connected to the individual therefore the real power behind the whole social structure is the power of individuals and their relations. It is not different than the definition of Castells but it is a developed version of what he mentioned with networks in 1995.

Castells also refers this new form of communication which is the result of the networks in general as the "mass-self-communication": It is mass-communication because it has the potential to reach the global audience, just like how a YouTube video reaches to massive amounts of people or how RSS[8] links work; and it is a self-communication because "the production of the message is self-generated, the definition of the potential receiver(s) is self-directed and the retrieval of specific messages or content from the World Wide Web and electronic communication networks is self-selected". Castells refers

---

[8] Rich Site Summary

this whole process as a transformation of the mass-communication to the mass-self-communication (Communication Power, 2009, p. 55).

It is possible to support this idea of mass-self-communication with the way apps evolved over time to enable users to create content and publish them on WWW by various means. Most of the applications of today have a social interaction capacity since it is a necessary part of their existence. Even a simple game application submits the high scores to an online score-board which lets the user to modify the linked content. This enables a social interaction between users and the data, therefore between users and users since they share the same database together through the use of various software.

In other words, applications are part of the network just like the individuals. They do not only enable the individuals to create or share the content but also they have the ability to gather information and share it just like the individuals. The whole system of network that is based on the individuals as the tool users and the applications as the functional tools create a huge structure, that we can call the network of today.

Depending on Castells' and Manovich's arguments on how social structures work, it is also possible to argue about the necessity of "sharing" since sharing is the only way to exist in the digital realm and that is how the whole concept of network is about. The idea of "sharing" is also one of the core ideas of today's social networks such as *Facebook* for "updating the status" or *Twitter* for "tweeting about things" or newly forming *Instagram* to share the visual content that is created by the users. All of these social structures, and many more, are completely related with the notion of sharing and they also put pressure on the users to share to be existent in their structure as without any sort of sharing it will be impossible for a user to interact with someone else. Once again, it is not only related to the individual user, but also to the app that is being used by the user since the apps of today has their own networks of information and data. It is essential to know the difference between the whole semantic structure of the Web and those small networks that work as a part of the WWW.

McLuhan argued that the communication technologies change and re-shape the cognition and social structure (The Gutenberg Galaxy, 1962).

It is very similar to how the development of print moved the society into a visual age, and television, radio and film helped society to move into a mass age of media (Campbell & Park, Social Implications of Mobile Telephony:, 2008). Therefore, the (in)famous argument of McLuhan: "The medium is the message" is transformed into "the network is the message" by Castells during the rise of the network society (2000).

As it was discussed by Berners-Lee the Web is becoming more and more application-oriented everyday with those "closed social models" such as *Facebook* and other social networks. Before these social models, one email address was able to send an email to any other email address being unrelated with their service providers. However, the current social networks form a new structure of communication that it is impossible to communicate with other people that are not connected to the social network, or that are connected to the other social networks. Because of these reasons, the users are limited with the world that the networks present them and all of the defined set of assets that are being presented by the social networks, in other words, the "closed" networks (Berners-Lee, Web 2.0 Summit 09, 2009).

In this case, it is possible to say that a network and the whole structural idea behind how a network works is completely dependent on its applications and what they present to the user. The information is re-shaped, modified or customized and the whole network is depending on that same data and evolving over the same data. This creates a second level of perception of the networks, which is quite different than the base level which was the pure Internet.

The WWW of today contains the apps in itself and in relation to each other, (social apps, mobile apps, web apps, etc.). It is not only about the mobile technology but also about the whole perception of the "web of society" through these applications.

It is also important to remember that in 2015 most of the Internet users will be using smart devices (O'Dell, 2010) (BBC News, 2011) (Alberts, 2012) (Rogers, 2012) to interact with the whole network; so that the use of applications will rise just like the whole rise of the network society. Castells also points this out in the following lines with an example:

> …in July 2007, YouTube also launched 18 countryspecific partner sites and a site specifically designed for mobile telephone users. This made YouTube the largest mass communication medium in the world. (Castells, Communication Power, 2009, p. 67)

Keeping in mind that the applications will be used to access the Web and to socially interact with the others, it is possible to say that the smart devices and app technology will play the role of the "closed networks" of Berners-Lee. The whole idea of social interaction using the apps and smart devices is completely related to (and sometimes based on) the closed social networks and applications that was mentioned above. The "Network Society" of this special condition, makes it an "App Society" since that is how the information and the pure data is filtered. Since most of the Web users will be connected to the Web through mobile devices, the content will be made for the majority in the future just like the web is mainly made for the desktop users today.

> Mobile communication contributes to an entirely new form of social order would be an overstatement… We consider this period of social change as part of a sociotechnological transition already in progress, which Castells and others have described as the network society. Hence, the parameters of our thesis are bound by the delimitation that mobile communication adds a unique new flavor to the social landscape, rather than creates an entirely new one. (Campbell & Park, Social Implications of Mobile Telephony:, 2008)

As mentioned by Campbell and Park (2008) unlike the "progression from McLuhan's mass age to Castells' network age" this progression to the personal age that is driven by applications and mass-self-communication methods is not a radical change from its predecessor but rather a more natural extension of it. When the medium became the network, "the network" had become message. Today it is possible to say that the applications are gaining ground to become the medium of the near future. In this case, depending on both of those assertions of McLuhan and Castells it is possible to say that today "the app is the message".

CHAPTER IV

DESIGN AND FUNCTION

4.1. Technical Artifact

We are surrounded by technical artifacts, material objects to serve practical purposes, and all of these objects are important and an essential part of our everyday life. Clever designs ranging from chairs and glasses to digital devices are ascribed with functions which help us to make use of such objects in our everyday tasks.

We, together with these objects around us, create "webs of significance" built as a structure of meaning and culture (Geertz, 1973, p. 5). This deep structure of culture, science, design, philosophy, technology and basically the whole relationship of everything together (things that form this web of connections where we, individuals, are connected to series of symbolic representations) is carried by those things around us.

Actor-network theory (ANT) as an approach to explain these connections of equally-valued human and non-human units argues that the whole social structure, network in this case, starts with interaction. This heterogeneous network is able to modify the social structure through the objects since "all of our interactions with other people are mediated through objects of one kind or another" (Law, 1992, p. 381). This necessity to objects take place in complex social relations. The *actor-network theory* also argues that those artifacts do participate in our communicative system in an asymmetrical way. For this reason, *actor-network theory* does not accept any sort of reductionism, therefore "in particular cases social relations may shape machines, or machine relations shape their social counterparts" (p. 382).

As a part of this growing structure of artificial members and their relations to each other and us, purpose is one of the key elements to understand this web of objects. Each of the artificial elements in the social structure exists for reasons to be sure, but can we define their pure existence depending on their purposes?

Depending on the *actor-network theory* it might be possible to argue that those non-human actors of the network are there for a purpose if we are keeping them in the network through interaction and communication. In this case, their existence can be explained by looking at their creation process: Their designs.

The primary purpose of design for the market is creating marketable products for selling whereas the social design depends on the satisfaction of human needs. The "market model" and the "social model" should not be taken as two opposites but should rather be seen as two different poles of the same structure since many of the products that are specially designed for the market fulfills the social needs and vice-versa (Margolin & Margolin, 2002). This point of view differs from Papanek's view on product design where he proposes that function and aesthetics are both parts of each other as well as being part of the "rest"of the system.

> 'Should I design it to be functional,' the students say, 'or to be aesthetically pleasing?' This is the most heard, the most understandable, and the most mixed-up question in design today. 'Do you want it to look good, or to work ?' Barricades erected between what are really just two of the many aspects of function. It is all quite simple: Aesthetic value is an inherent part of function (Papanek, 1972, p. 12).

We can surely say that a lot of things has changed since 1972, especially with the technology and the social structure followed by it, but it is still possible to stay connected to both of the attributes of everyday products: The aesthetics and the function. Both of them can surely be expanded here but it is possible to argue that all other attributes related to objects evolve around these two main attributes when the product design is the case, since both of them are hidden in the definition of the design itself. Good design is identified to be the "best solution to a problem", whereas the "best" stands for the realization of numerous criteria, as well as aesthetics.

Using objects as part of our daily lives to achieve goals is a common and natural activity. These actors - objects, gadgets, or things in general is not only the dominant part of our networks, relations and environment, but using them is also the dominant part of our lives. Most of the people do not think about how they wear their shoes or pick up their glasses to perform a certain task during the day. For this reason, the design of object is as important as the use of it since they are related to each other.

Figure 4.1 - Material posessions of a family.

This relationship that *ANT* points out is almost identical with the network society theory of Castells, where this time the whole network is not only dependent on those individual relationships and cause-effect balance but also depending on the power, where power is defined as the "relational capacity" that enables a social actor to influence other social actor(s) (Castells, Communication Power, 2009, p. 10). This, as an expansion over the original *ANT*, is a good way to point out the social (im)balance between objects depending on numerous variables that change the identity and the structure of a single object.

Within this complicated network of actors, one can not simply say that the objects do exist naturally. The main focus should be the design of the objects to understand the relation between the object and the network, in other words, the existence of this object within this network of relations. In this notion of an object's existence as a part of the network, Houkes and Vermaas point out the "use" of an object in terms of an activity.

> Playing with a tennis ball while having a conversation is not naturally described as using the ball. Describing some activity as using of an object x invites the question what x is used for. Using is using with an aim or goal. Similarly, 'designing' carries strong connotations of purposiveness and intentionality (Houkes & Vermaas, 2010, p. 24).

It is possible to see this as definition of design itself, as it is a goal-directed activity. The aims and goals that are embodied by those everyday objects are results of good planning and "goal directed design", as a part of that object's lifespan from its creation to its obsolescence. Houkes and Vermaas also mention that goal-directedness is an integral part of the design process since human beings are "capable of (goal-directed) use and design of the artifacts" with the help of certain level of skill or knowledge (Houkes & Vermaas, 2010, p. 26).

Design research is a way of approach to understand these cases. In this manner, according to Bayazit the design research as the "systematic inquiry whose goal is knowledge of, or in, the embodiment of configuration, composition, structure, purpose, value, and meaning in man-made things and systems" (Archer, 1981) is concerned with the physical embodiment of "man-made things, how these things perform their jobs and how they work" (Bayazit, 2004). Since purpose is the part of Archer's design research definition according to Bayazit, it is important to note the relation of the design of the object to its purpose.

On the other hand, Neander explains the relation of function and the purpose of an object as a part of teleology, and points out the problem:

> For example, we may offer an explanation of the switch on the wall by saying 'it dims the lights' and in doing so we apparently offer an explanation of the switch that cites its effect rather than its cause. Barring backward causation it looks as though the switch's turning on the light cannot causally explain the switch's being there on the wall, and so it looks as though the explanation is illegitimate. The prima facie problem gets worse, if that is possible, because many purposes, goals and functional effects are never realized: most athletes never win their gold medal, some inventions fail to perform their intended function, hearts occasionally fail to pump blood, and so on. Unrealized effects have no potential as causes of anything at all, yet mention of them is taken to explain the existence of the item for which they are potential but unrealized effects (Neander, 1991).

Neander also points out the ambiguity in talk of purposes within this puzzle by the notion of "intention": Supposing that Hagar refuses to drink alcohol for losing weight; which makes his intention to lose weight. If that purpose is not fulfilled, Hagar still has the purpose. In this case, purpose as a guide to the behavior is a part of this teleological explanation where these explanations are "just species of ordinary causal explanation" (Neander, 1991, p. 456).

This brings Neander to the second step into the explanation of the relation of function and purpose where artifacts are the main concern since they appeal to functions.

> If an artifact's function is the purpose for which it was designed, made, or put in place, then an artifact's function could explain the artifact in the same way that the purpose of some behaviour explains the behaviour (Neander, 1991).

The intention of the person that designs the artifact is in question since the artifact is created with an intention, a purpose; therefore a function. Therefore, if the "agency involved in the solution is transparent, teleological explanations can be understood as a species of normal casual explanations" (p. 456). In this case, the switch on the wall that can "dim the lights" explains its being on the wall because when we learn about its function we also learn about its purpose and the creator's intention, which is making the lights dimmable. In this case where "these agents acted to design, make, or place the artifact thus and so, with the purpose of bringing about the effect that is the artifact's function (or minimally, with the hope of bringing about the capacity for effecting the function)" (p. 457); we can explain the being of everyday artifacts since all of them are created by agents for function(s). In other words : They are “designed”.

This takes the overall focus to the design of objects therefore to the functions that are ascribed to them. Intention and purpose, as the main elements of directing the design process of the objects, is important for the ascription of functions to the objects.

In *The Constuction of Social Reality* (1995) Searle suggests that the assignment -or imposition- of the functions is added by the observers regardless of how or for what purpose they are created. They are never ascribed by the creators and as a matter of fact they are “assigned from outside by conscious observers and users” and “never intrinsic but always observer relative” (p. 14).

> It is, for example, intrinsic to nature that the heart pumps blood and causes it to course through the body. It is also an intrinsic fact of nature that the movement of blood is related to a whole lot of other casual processing having to do with the survival of the organism. But when, in addition to saying “The heart pumps blood” we say, “The function of the heart is to pump blood” we are doing something more than recording these intrinsic facts. We are situating these facts relative to a system of values that we hold. (Searle, 1995, p. 14)

According to Searle, those "nonagentive functions" which do happen as a result of natural being are different from the "agentive functions" which are added to the objects by conscious agents. Searle points that when we say "this stone is a paperweight", "this object is a screwdriver" or "this is a chair" we mark their uses and marking the use of an object is neither a discovery of the function nor a result of a natural happening. Instead, they are "assigned relative to the practical interests of conscious agents" (p. 20).

Searle also argues that some objects are specially made for those agentive functions, just like a hammer, but it is still possible to use the hammer as a paperweight. However, there is no sharp dividing line between the agentive and nonagentive functions since "sometimes an agentive function can replace a nonagentive function, as when, for example, we make an "artificial heart" (p. 20).

Vermaas and Houkes, on the other hand, generalizes the constraints for the ascription of functions to objects for function theories for the technical domain in four main captions which are absent in modern function theories.

The first one, as opposed to Searle's argument of the agentive and nonagentive functions, is the "proper versus accidental" debate. Vermaas and Houkes differentiate the proper function(s) from the accidental one(s). John, as one of the actors of their supposed network, is standing on a folding chair to reach the top shelves of his kitchen. His neighbor is concerned with this situation since the chair is not for standing on but for sitting on. John mentions that his stepladder is broken and he has to make the job with the chair instead.

According to Vermaas and Houkes, a theory of functions should differentiate the proper, which is sitting on the chair, from the accidental, which is other types of functional uses of the chair such as what John did. Proper functions are the functions that are originally ascribed to those artifacts at the creation and intention process whereas accidental functions are ascribed occasionally (Vermaas & Houkes, 2003).

Millikan also defends a quite different notion of proper functions by caring about their historical definitions. According to Millikan, the actual definition lies within that historical turn which looks back to the history of an item to determine its function rather than using the current properties of it (In Defense of the Proper Functions, 1989).

Millikan states that the definition of "proper functions" is recursive. If the A item have the function F as a "proper function", it should hold one of the two conditions.

> (1) A originated as a "reproduction" (to give one example, as a copy, or a copy of a copy) of some prior item or items that, due in part of possession of the properties reproduced, have actually performed F in the past, and A exists because the product of some prior device that, given its circumstances, had performance F to be performed by means of producing an item like A. Items that fall under condition (2) have "derived proper function", functions derived from the devices that produce them. (Millikan, 1989, p. 288)

Millikan's approach is more like the explanation of how those functions are ascribed into the items as "proper functions" whereas Vermaas and Houkes accept the ascription of those functions as a natural result of the design process. In both cases, the proper function of an object is defined by all means and regardless of it being a result of an historical approach or a design principle, it exist as an attribute of the creation. Searle's approach, on the other hand, seems quite different from those other two approaches where design plays role on the intentionality of the ascribed function as a part of agentive functions where those functions are also assigned when we say “that is an ugly painting” since “all these are instance of uses to which agents intentionally put objects” (Searle, 1995, p. 20). By comparing all three of them it might be quite possible to say that the “proper” and “accidental” approach is better for explaining the technical artifacts since design plays a huge role on the final output of those functional apparatuses.

Vermaas and Houkes also underline another attribute of their function theory, which is the “Malfunction” that happens when artifacts are unable to perform their functions, just like the broken stepladder (Vermaas & Houkes, 2003). Malfunction is also frequently used in information technologies as a part of software development.

As the third desideratum; Vermaas and Houkes suggests the notion of “physical structure” which should exist in a function theory since structural conditions are sufficient for the performance of the artifact. They exemplify the condition with another supposed case: John, who wants to put up his tent, lost his pegs. After looking for the most suitable artifact to do the job he decides on the nails since the wooden pins break easily. This phenomenon explains the physical structure in terms of identifying functions because the main reason for John to pick the metal nails over wooden pins is their physical strength over the wooden pins (Vermaas & Houkes, 2003).

The final caption they suggest is the "novelty", where a theory of functions should admit an ascription of proper functions to innovative or atypical artifacts.

They explain the condition again with John, this time holding an artifact that is the blend of a typewriter and a phone, but which is neither of them. The design he is holding is something completely new, which can send a typed message by using the telephone signals. Novelty, as a title to explain this situation focuses on the creation of the innovative function, as well as the creation of the object-body that carries the function. They bind to become the functional object together (Vermaas & Houkes, 2003).

Not all of those titles are directly related to the design process of the object but we can say that all of them are included within the lifespan of the objects. For this reason, it is important to note down these captions for the explanation of virtual objects, such as apps, which contain those four elements as an evident part of their process of creation.

4.2. Interface as a Body

The modernist design formula "form follows function" states that function is a guide for the process of creating the physical object. The functions that are ascribed to the object define the physical structure of the final design or create boundaries and outlines for the design process. But what about the virtual objects? How can we compare their relation of function and form in terms of physicality?

By looking at the simple objects like a spoon or a glass, it is possible to make connections between the function and the form of the object. In those simple objects, the form is almost the solid representation of the function.

A spoon would be meaningless without its form, just like a glass. Similarly, a door would be meaningless if it was made too thin or too small for the decoration or other aesthetic concerns. In this sense, their form is what makes them unique and functional. In other words, their form is the aspect that realizes their function. The progress of the design of these objects is overlapping completely with their functional existence, in other words, their purpose.

Just by looking at their physical appearance, it is possible to make assumptions about simple functional objects. Because of this, "true functional design has very little to do with what we call progress" since it simply tries to realize the function (Grillo, 1960, p. 26).

Technically speaking, for the mechanical objects, as the function gets more complex, the physicality of the object generally follows it. The physical complexity of such devices increase with the ascribed functions' complexity, therefore, the form becomes a representation of the functional aspect of the object. As an example, the whole logic behind a mechanical Turing machine is a very complex combination of physical parts when we compare it to the simple spoon.

However, the technical complexity of a tool might be different than the complexity of use. It is possible to compare a mechanical calculator, which is fairly complex to use as well as its mechanical structure, to a digital calculator, which has the ease of use as well as additional functions but has more technical complexity than the physical device.

In one of his presentations, Donald Norman states that even the simplest everyday objects are not as simple as we think they are (Norman, Stanford University: Don Norman's Complexity, 2011). He shows the audience a picture of a salt and pepper shaker (together) and asks the audience which one is the salt and which one is the pepper. Half of the audience says the one on the left is the salt, and the other half says the exact opposite. Then he states when these people are asked about "why the one on the left is the salt" or "why the one on the right is the salt" depending on their answers to the previous question, the second answer they give is quite interesting: "The one on the left is salt because it has more holes", "the one on the right is salt because it has one hole". According to Norman, it does not matter what the audience thinks. What really matters is what the person who filled these shakers thinks.

Depending on this starting point, Norman states that complexity occurs in every sort of object that might look simple, or it might be absent in objects that look complex.

> We need complexity…Because life is complex and the tools we build need to match life. When people say "I want simple things" they do not look for simplicity, the look for things that they can understand… So I would like to contrast complicated and complex. I define complicated as confusing. And complex is just like how the world is, with many different parts and features. The difference between complicated and complex is in the head (Norman, Stanford University: Don Norman's Complexity, 2011).

Depending on Norman's approach to complexity, it is possible to define the "use complexity" as the "level of complicatedness". Norman states that, the complexity is directly related to the technical detail of the object. In this sense, the object is complicated because of the human perception but it is complex because of the technical details.

Vermaas and Houkes mention that skills are essential to using objects. An artifact becomes complicated when they involve more than one action or steps. In this sense, learning to use an object, therefore the skill, is directly related to doing appropriate actions and using the appropriate order. As an example, making a toast with a toaster requires taking the object from where it is stored, plugging it in, putting bread into the rack, setting the toasting time, pulling down the toast rack or the switch and waiting the required amount of time. Missing one of the steps will not result with a toast therefore all of these steps should be applied in the right order. The description is only a sequential form of actions applied in a natural use process of the toaster. Moreover, some of those steps require the object to be manipulated, such as the timer and the switch. This series of actions lead one to the realization of the goal, in this case, the toast (Houkes & Vermaas, 2010, p. 26).

> We seek to reconstruct the process of deliberation that underlies artefact use. Therefore, we need to look at the series of actions that constitutes use, and we need a concept that is tailor-made to describe such series. This concept is that of use plan: a goal-directed series of actions, including manipulations of the artefact and its components (Houkes & Vermaas, 2010, p. 17) .

Houkes and Vermaas also points out the value of use and use plans within the process of design. According to them, the use plan approach allows analyzing the evident relation between the users and designers. Their characterization of the use plan does not require "designing to result in material objects that were previously non-existent" since according to this characterization, activities that result in new material objects are a subtype of designing called product designing (Houkes & Vermaas, 2010, p. 26).

The advancement of technology in this sense, which helps the transposition of mechanical parts into digital extensions of the device that need less space and less technical knowledge to use, is one of the most important aspects of today's product design. In this sense, the use plans get smaller for the devices with same (or similar) functions as the technology develops in the design of these objects. A fully physical and mechanical coffeemaker would need the skills and knowledge to use the device as a whole as well as the separate parts for various options, such as the parts to generate the foam or to cook the milk; whereas a modern coffeemaker would do the same coffee (arguably) with only one switch and almost no skills.

Figure 4.2 – Coffeemakers: Old vs. new.

If technology affects the design so much in terms of simplicity and functionality, can we make a connection of designing the products to the virtual machines of today? Looking back to the virtual machines of everyday life, it is possible to say that they exist in a very similar sense to the tangible functional objects. A calculator, exist directly for its functional use; or a full operating system is a completely functional base for everything within a system. Despite their virtual existence, they are very similar to physical objects in a sense of their functional uses.

An application is a fully functional tool that is almost similar to the tangible tools in terms of use plans and process of design. It needs certain skills, certain knowledge and similarly a sequence of actions to be used properly. In this sense, an application can be compared to a physical object in terms of representation of the function.

A physical object's physical being reflects its function through its form as mentioned above. The function that is carried within its design is reachable through the form of the object. In other words, the object is open to interaction. With the help of this interaction, the object is used and the function is revealed. A glass can help someone to hold a liquid with the help of its form and physical existence, a mechanical compass can help someone to navigate, or a scissors can cut papers if the correct use plan is applied.

As a result of digitalization, miniaturization and electronics, the contemporary technology is nontrivial and its operation remains incomprehensible. Because of the need to allow their use by non-specialists, another type of technology emerged on the dynamic border between the complex technological devices and their users: Human-machine interfaces.

Widespread use of human-machine interfaces in all sorts of technology creations from the control rooms of atomic power plants and airplane cockpits to handling of the extraordinary tools created for specific purposes became a crucial part of those creations that can not be separated from those human-machine interfaces any more.

This change must be seen as "enabling what humans do with them". This technology shifted the designers' attention from a concern of the appearance and makeup of technology to what mediates between them (Krippendorff, 2006, p. 9).

> The three most important features of interfaces are their interactivity, dynamics, and autonomy. Interactivity refers to the action–response sequences, command and execution cycles, or the give and take that is inherent to the human use of machines. Dynamics implicates time and refers to the human use of artifacts, which rarely ever returns to its point of departure. Autonomy emphasizes the containment of the process (Krippendorff, 2006, p. 9).

Evolution of computers is quite a similar process in this sense. The interface, as a new kind of artifact of its time, became a more obvious part of the computers. Typically, personal computers are extremely complex electronic devices with an operational architecture that escapes from users' understanding. As a result, the purpose of the physical computer interfaces (like keyboard, mouse, buttons, etc.) is "to render usable that which can no longer be understood in functional details" (Krippendorff, 2006, p. 78).

On the other hand, the notions of interface and form are quite different for applications. A compass app in a smart device is - just like the mechanical compass - a tool for navigation but its physicality is completely different than the mechanical compass. The way they work are almost completely different since the mechanical compass uses simple magnets whereas the app has access to the magnetometer or GPS of the smart device to calculate the magnetic fields or the exact position of the device. It is also important to remember that the other applications on the same smart device can also use the same physical parts of the device, such as the GPS. The functional hardware that belongs to the smart device is similar to a modular toolbox of useful functions which can be reached and used whenever they are needed. All of the functional parts of a compass (the glass, the needles, the magnets, etc.) belong to the device itself whereas the hardware that is used by the application belongs to the actual smart device, not the application.

At this point, the smart device as a whole, with all of its internal parts and functional hardware, is a host to the guest applications that use the device's hardware. Therefore, the notion of interface is completely different for media devices such as smart devices.

> Today a typical information device such as a mobile phone has two kinds of interface. One is a physical interface such as buttons and the phone cover. The second is a media interface: graphical icons, menus, and sounds. The new paradigm that treats interaction as an aesthetic and meaningful experience applies equally to both types of interface (Manovich, Interaction as an Aesthetic Event, 2006, p. 69).

"Form follows function" dictum does not fully work considering the physical aspect of the application at this point. Form, as the form of the smart device, works well to hold the internal hardware (functional parts) together; however, it is almost completely unrelated to the applications that are installed in the device. The real form of the application is hidden in its interaction.

A GUI[9] is type of human-computer interface that visually represents interactive elements of the software. A GUI is in contrast with command line interfaces (CLIs) which use the text based commands to control the machine (LIP, 2004). The first graphical user interface was developed by Alan Kay, Douglas Engelbart, and a group of other researchers at *Xerox PARC* in 1981, with the idea of turning the

---

[9] Graphical User Interface.

computers into a "personal dynamic media". It is commercialized with the *Apple*'s Lisa computer in 1983 (ComputerHope). The GUI is a vital element of an application since it is the way of interaction that we use in today's computer technology. Alan Kay and his group not only developed a simple tool to visualize the functional properties of the device, but also made a completely new invention by carrying these functional attributes to a non-physical medium – the screen.

> The media interfaces[10] themselves – icons, folders, sounds, animations, and user interactions - are also cultural software, since these interface mediate people's interactions with media and other people (Manovich, Software Takes Command, 2008, p. 15).

According to Manovich, the emergence of the graphical user interfaces (and media interfaces that followed GUIs) are what transformed the culturally invisible computer technology into "new engine of the culture" (Manovich, Software Takes Command, 2008, p. 21). Technical knowledge, or in other words, the use plan is always necessary to be

---

[10] While the older term Graphical User Interface, or *GUI*, continues to be widely used, the newer term "media interface" is usually more appropriate since many interfaces today use all types of media besides graphics to communicate with the users (Manovich, Software Takes Command, 2008, p. 15).

able to use a device. Because of this very simple reason, GUI step is crucial since it made the technology accessible and usable by the non-technical users, which are the dominant part of the society all the time, just like the time when computers emerged into the cultural network.

Today media interfaces are one of the most important parts of an application. The virtual machines that represent the functional use through various formats of interfaces are completely dependent on those interfaces since the normal user has no reach to the functions or any sort of content that is hidden in the application unless there is an interface for it. It is very similar to the concept of physical interfaces in this manner; however, the form is only a sum of flexible visual representation of the interactive elements. In this context, applications are virtual machines with virtual interfaces. The interface is not only effective for the functional use and the concept of that virtual machine, but also very important for the feel of the object which can be adjusted with the help of design; just like the feel of a car that depends almost completely on its product design.

This is a very important point of the whole discussion of apps since the whole realization of the functional properties of an app lies within its visual interface. Every kind of interaction, every sort of possibility is a part of the applications interface through buttons, sliders, pointers, dropdown menus, and various evolving types of user interface elements. For the user, existence of the application's functions depends completely on this visual preference since if there is no interface element for a function it is impossible for the user to reach that function for use.

As a result, the user interface is not only a simple visualization of the contents of an application, or virtualization of something that already exist (physical interface) but rather the whole form of the application in a very similar relation compared to a spoon and its physical form. Just similar to how a spoon's form reveals its functional use, the interface of an application reveals its functional being in terms of usability.

It is also possible to see the three features of interfaces that Krippendorff mentioned in the media interfaces: Interactivity, dynamics and autonomy (Krippendorff, 2006, pp. 9-10).

In one of his presentations, Donald Norman mentions that there is a dynamic sweet spot for the complexity of the interaction of products' designs. As the functions increase with desirability, the capability of the design increases since it has more of the desired functions; whereas it becomes complicated and usability drops. The sweet spot for an interaction design is around the intersection point of the usability and the capability lines (Norman, Stanford University: Don Norman's Complexity, 2011).

The whole graph is completely dynamic and it depends on the hardware used since some sort of hardware might be better to hold functions than the other. The "coffee machine" is a good example of this sort of a comparison. The transformation from the completely mechanical coffeemaker to the newer coffeemakers with electronic hardware; where most of the manual mechanical functions that need complicated use plans are simplified with the usage of digital technology. The interface of the electronic machine is far more simple than the mechanical version with the exact capability. Depending on this equation, it is possible to say that as the digital device handles the manual work that had to be done by the user; therefore, this increases the usability as well as the capability of the device.

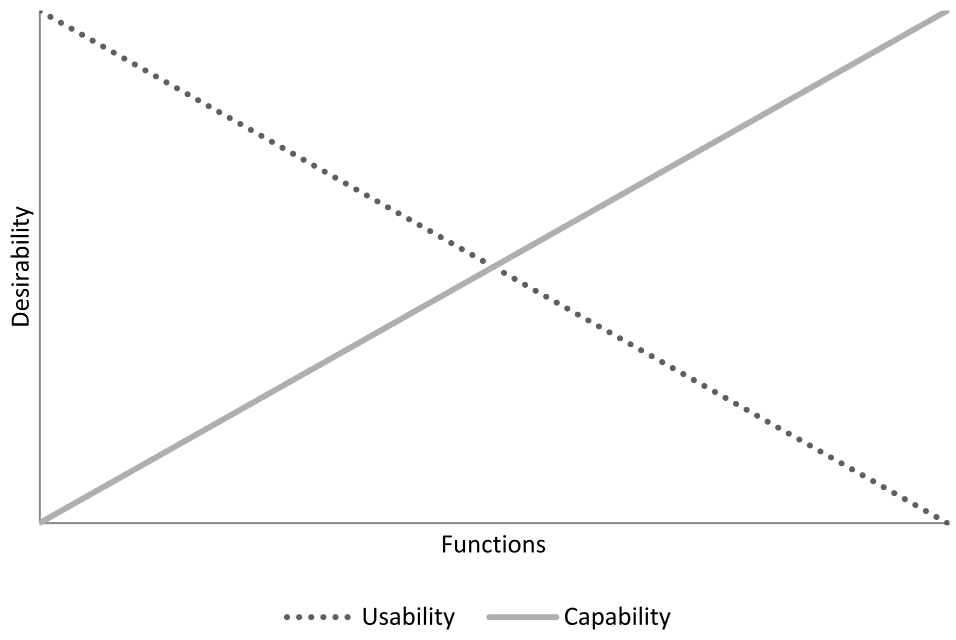

Figure 4.3 - Usability and Capability.

It is very similar with the applications as well. The way users interact with the application still depends on the hardware used. With the creation of touchscreen technology which simply revealed the use of fingers instead of a pointer hardware like a mouse, it became even easier to interact with the hardware since the way we use our fingers is directly related to interaction. In this sense, touchscreen is a more direct way of interacting with the media interfaces. The touchscreen is a hardware just like the mouse pointer and both of them are used to set the virtual point of interaction however the way touchscreen works is completely WYSIWYG[11] whereas the mouse needs additional skills to learn and interact with the virtual elements.

Just like the touchscreen, it is possible to talk about the multi-touch[12] technology which helped the whole interaction methods to evolve once again depending on the new rule set. With the help of this technology the users became able to perform natural gestures by interacting with the multi-touch interface. Those gestures such as “pinch to zoom”, “swipe to navigate”, “three finger drag”, “rotate” are realized with this technology and provided the users a new way of interacting with their device, in a more natural manner. To rotate an image on the screen, the gesture is the simple “rotate” with two

[11] Acronym for “What You See is What You Get”.
[12] Functionality allowing a touch screen, trackpad, etc., to register multiple points of contact made on the surface simultaneously.

fingers, just like how it is performed to an actual photograph in a physical space. This natural responsive behavior of the interface made it almost invisible as well as making the applications very close to the actual functional objects in the sense of they are interacted with (Apple, 2011).

As a result of this technology, the design of the virtual interface is similar to the design of a mechanical interface from a point of view. However, there is a certain freedom in this instance since it is all virtual, and everything is possible. As an example, "sliding" a paper to see more of the text is a natural behavior that can be realized virtually with the help of this technology with incredible amount of control.

This sort of a control gives the developers the ability to change the behavior of the whole design like the friction of the paper on the virtual surface by controlling the acceleration / deceleration amount and such. Similarly, a completely new type of gesture-based interaction could be designed to match the needs of the application, in other words, the product.

This type of approach in order to design the behavior of the interaction of things would be very hard (sometimes impossible) to create in actual objects with the physical design of the product. This hyper real condition is almost like the perfection of the existing interaction methods therefore it gives amazing possibilities to both users and developers.

These important aspects of the virtual interfaces also widened up the doors of a newer design branch: the user experience design (UX Design, UE Design, UXD or UED), which unites various branches of design together to create a better usability. Donald Norman, the inventor of the UX defines user experience it as follows:

> "User experience" encompasses all aspects of the end-user's interaction with the company, its services, and its products. The first requirement for an exemplary user experience is to meet the exact needs of the customer, without fuss or bother. Next comes simplicity and elegance that produce products that are a joy to own, a joy to use. True user experience goes far beyond giving customers what they say they want, or providing checklist features. In order to achieve high-quality user experience in a company's offerings there must be a seamless merging of the services of multiple disciplines, including engineering, marketing, graphical and industrial design, and interface design (Norman, User Experience).

In this sense, the UXD is an extension of user-centered design. Luke Wroblewski defines the structure of this user-centered design in three headings: Information architecture, interaction design and visual design. Information architecture defines the general structure of the information, which can be in many forms. That information is important to be the basis of the whole UXD. Interaction design as the second important step is essential for the manipulation of the information. It enables people to interact with the existing information therefore it is the bridge between the information and the visual elements of the UXD. And finally, the visual design communicates these possibilities to the people visually and creates desirability (Wroblewski, 2009).

In this sense, UXD is directly related to the virtual interface since it is also the summation of all of the three elements of UXD that are mentioned above. Therefore, it is possible to say that user experience design is an essential process of designing the interface of the application.

As a result, the virtual interface of the application is not only a supporting extension of the software but also the way software is interacted and its function(s) is revealed; therefore, the virtual interface is the way the software is realized. Without the interface the function is still there without the access of the user. Interface, as a functional part of the software, is what behaves as a bridge between the user and the function. Each element of the interface is functional through the operation of interaction. In this sense, the form does not simply follow the function but it also has a function: To functionalize the function(s) of the application. Therefore, the interface is the body and the existence of the application; it is how the function(s) of the application comes alive for the user.

4.3. The Prestige: Function as an Object

Function based design is the key to make a product attain certain levels within the social structure. Complexity is another aspect of products because as the use of a product becomes detailed or needs modification(s) it needs specialization on its functional use to overcome the issue of being too complicated.

If a user is looking for a pair of scissors for a basic functional use, to cut, (s)he can get one to cut something into pieces. However, if the user is going to use it for a special condition they need to know exactly which scissors they are going to use. In this sense, a hair thinning scissor is functionally (therefore design-wise) very different than kitchen scissors, grass scissors or tin snips. This difference which is important to create functional diversity in a tool is one of the key elements of the tool's survivability.

The specialization that happens to objects is very similar to how mutation and evolution happens in biology during the natural selection process and helps some species to survive where others disappear. As a result, some objects survive longer than the others with the help of the product development and evolution as some of them slowly fade into the past.

Figure 4.4 - Various types of scissors.

Before the evolution process of the object, the modifications need to be specified since without specifications it is impossible to determine the point of possible change.

> Specifications are a concept which we all know from everyday life. A specification can be defined as a description which can unambiguously transfer needs or intentions from one group of people to another (Hvam & Mortensan, 2008, p. 18).

As Hvam and Mortenson states, specifications are directly related to the user(s) of the product. In this manner, specifications are important for a product's life cycle from the identification of its need to its disposal. Hvam and Mortenson describes the whole cycle in eight different steps from the identification of the need to the disposal of the object. In the case of a functional device, the need is either related to the function or related to the design therefore the modifications that happen are connected to the use of the device (Hvam & Mortensan, 2008, p. 19).

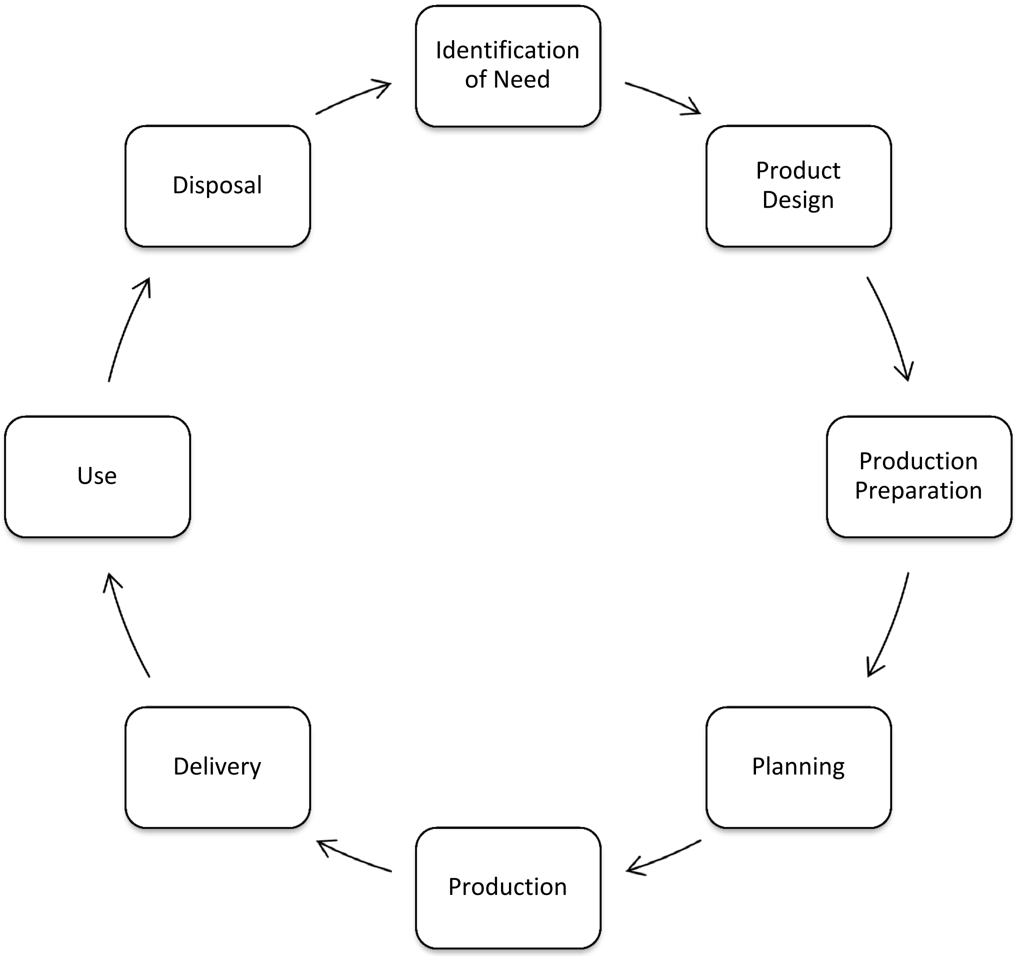

Figure 4.5 - Specifications in the course of a product's life cycle.

In this sense, specification is important for the customization of the product for a special situation. Customization modifies (or reconfigures) the product design of the object depending on the specifications.

The term “mass customization” emerges at this point. Mass customization is a revolutionary method for mass production, which is defined as “the application of technology and new management methods to provide variety and customization through flexibility and quick responsiveness” by Gilmore and Pine II ("Making Mass Customization Work", 1993, p. 112). Duray and Ward define it as "building products to customer specifications using modular components to achieve economies of scale” (Approaches to Mass Customization: Configurations and Empirical Validation, 2000, p. 611). It has also been defined by numerous authors at different times and it is possible to see that those descriptions vary over time with the change of technology. However, as a general definition that stands at the intersection zone of existing definitions, it is possible to say that mass customization is technology which customizes the product for the user.

What happens during the process is surely technology-dependent. Sometimes with the help of a modular design the final product is formed for individuals; and sometimes a generic system re-creates the whole product from the very beginning. The main logic behind the whole strategy is the personalization and customization of the product depending on the individual: The user.

In one of his presentations about mass customization and UX design, Donald Norman states that the term "users" sounds very cold and they should be referred as something else: "People" (UXWeek2008: A Speak with Don Norman, 2008). This funny argument is true since the whole notion of customization is directly related to the UXD of the product therefore, the personalization.

Norman also states that the physical customizations of the objects are not emotionally compelling even though numerous manufacturers have tried to overcome this. It is simply because none of them really guarantees the emotional attachment.

> Things do not become personal because we have selected some alternatives from a catalog of choices. To make something personal means expressing some sense of ownership, of pride. It means to have some individualistic touch (Norman, Emotional Design, 2004, p. 220).

Adaptive customization, as a branch of mass customization, is very important in this sense of personalization since it gives the opportunity of making customizations to the end user (Oliver, 2007). This simple idea of mass customization is extremely powerful since it is a win-win situation for both sides: The users and the manufacturers.

Mobile phone is an important step for the whole adaptive customization idea. It is used by massive amounts of people and it is a personal communication tool so it is directly related to the individual and the concept of identity. Castells explains this situation in the following lines:

> "Wearability" makes the mobile phone an item of fashion, ready to be personalized to reflect the identity of the owner. The built-in capacity for customization is a major breakthrough which allows users to play a more active role in the shaping of this particular consumer culture (Castells, Fernández-Ardèvol, Linchuan, & Sey, 2007, p. 107).

This situation supports the mobile technology to become deeply affected by the consumers and social groups, such as women. It is simply because, the users are no longer just users but also producers or “co-creators” (Katz & Sugiyama, 2005, p. 79). In this sense, user participation influences and re-shapes the whole mobile consumerism.

Castells states that “entertainment is a fundamental dimension of the media world” (Castells, Fernández-Ardèvol, Linchuan, & Sey, 2007, p. 109). The major components of mobile entertainment include mobile gaming, chat, information services, location services and media content consumption (icons, ringtones, images, etc.). According to him, entertainment becomes a new reality when it is applied to the telephony since it changes the traditional telephony and traditional entertainment with the new mobile communication devices that have better audiovisual capabilities and faster development. As a result, entertainment is becoming an important function of the mobile communications (p. 111). As a result of this abundance of creative effort in the entertainment area and the consumption cycle, users have more options to modify their mobile devices in terms of adaptive customization.

However, there are significant amount of people who buy the mobile phone as an "off-the-shelf" item and directly use without any customization of these entertainment and fashion components for personal identity. Katz and Sugiyama uses the term "passive" for this type of users, since they do not affect the mass customization scene as much as the user type who cares about the personalization of the device (2005, p. 81). Those passive users need something more than the fashion and identity to customize.

Therefore, one of the most important things for the mobile technology of today is that users have the ability to customize their gadgets on their personal smart devices. Therefore, the adaptive customization process exceeds the manufacturer and becomes heavily affected by the end user.

In this sense, *AppStore* was an important step to re-define the applications within the social structure. It is important for the whole commodification process of the apps as how we know them today. It is arguably stronger than any previously realized aspect of their marketing process by turning them into essential pieces of the smart devices, therefore making them a fundamental part of the mobile industry. Apps of any virtual store includes various categories from entertainment to shopping and they can be picked freely by the user depending on the functionality (s)he is looking forward to see on the customized smart device.

When we go back to the desktop applications and customization in this manner, it is possible to see that most of the customization happens through the interface of the software. As an example, a *Photoshop*[13] user can change the layout items of the software by dragging and dropping the panels around the screen. This way, the user can completely customize the whole interface and create a personal layout depending on the functional needs. If (s)he needs the typography tools and the brush tools on the screen, it is completely

[13] Adobe Photoshop is a graphics editing program developed and published by Adobe Systems.

possible to re-arrange the layout to fulfill the needs of the user. This is a successfully executed use of customization through the use of dynamic interfaces.

However, the customization of the whole device (computer, or any sort of smart device) is completely another type of customization compared to the layout or interface customization. We have already established that form is function since it is the only way to reach the functional use of an object. In this sense installing a new app to the smart device is something more than that. It is not only the visual change of the device, but the implementation of the whole function to become executed as a function of the smart device, therefore it is directly a way of adding new functions to the physical being of the device.

Without the applications that are installed, the device would be functionless and meaningless; therefore, the applications add function(s) and purpose to the device. As a result, the actual device becomes the carrier of every function it contains through the apps and the usage of dynamic interfaces and shared hardware. Its capabilities are expanded with the software and its physical existence becomes almost like a transparent rail which keeps everything in track. As a matter of fact, the smart device's functionality is completely app-dependent.

Figure 4.6 - Various apps.

Depending on this approach to the function and device relation, it is possible to state that during the process of customization of the device by adding and removing apps, function becomes something that can be ascribed by the user. This ascription, compared to the accidental use of the chair (by stepping on it instead of sitting) is not an "accidental function" but a "proper function". Therefore what happens during the process is a true ascription of the function.

In other words, function is not simply an extension of the object. Function behaves as an object by itself. It can be ascribed, changed, modified, specialized, customized, removed, commoditized, exchanged or completely become obsolete just like actual objects.

In this sense, an app is a virtually realized function therefore it is an object. It completely changes the purpose of the smart device. As an example, a compass application that is installed on the device changes the way device works both logically and technically. The effect is not only happening to the device but also to the user, since the whole use-plan logic depends on the way a user approaches a product. Therefore, the way we approach the compass application is completely different than the way we approach to a music player, a game, or a virtual bookstore.

The physicality of the device is still important since without the hardware, the existence of software would not be possible. But it is also possible to assert that the physicality of the smart device is not directly in relation with its software-contents in a similar sense as a mechanical device and its relation to its counterparts.

The physical device, which behaves as the carrier for the app(s) and function(s) is somewhat like a rack of tools which can hold every category of object from the utilities to entertainment together in an organized way. This shift happens within the boundaries of proper function's definition since the form of the object depends on the proper function.

As a result, it is possible to say that function can be realized as either virtually or actually as the main purpose of creation of the object. In this sense, the actual or the virtual form of the object becomes a shell that is required for the function's realization and existence.

This radical shift of the physicality of the actual object into something completely virtual happens exactly at the line between the actual and the virtual, in other words, the physical and the logical. Software, does not only behave as the logic of the operation any more, but also behaves as a complete object since its existence contains and shelters everything necessary to be labeled as an object. The way we approach and perceive these seemingly-virtual elements of today's technology is very different than how we approach and perceive them two decades ago. Therefore, object's definition can be expanded to include this shift of software from being something completely logical and virtual into being something that is way more related to actual than it was before.

CHAPTER V

CONCLUSION

This study offers a new way of understanding the concept of function in the context of apps. During the research process on the function-object relation, technology which forms the basis of this study is reconsidered and partly recontextualized to make some definitions possible and connect the dots between each step.

From the relation of software and hardware to how society relates silent revolutionary changes that happen during the technological evolution of devices, one of the main concerns of this study is to provide a ground and inspiration for related researches. Also note that this way of thinking to understand technology was the main inspiration for this study.

Smart device as a concept, is one of the key elements of the whole structure of this research since it serves as a practical proof of the concept that an object can be ascribed with proper functions during the adaptive customization process. This concept as the fuel of this research may point out a new era where it is necessary to redefine the objects as they were not defined before. It would be extraordinarily impossible to determine the next step without this type of technology.

Finally, it is also important to note that cloud computing as the continuation of this era promises another radical change in the close future. However, to maintain the direction of the study, the whole concept was not discussed at depth but only used as a tile to define the current boundaries of this technological step.

As a conclusion, the approach to relate functions with objects in a way that functions are designed just like objects is only an evolved idea of design. The possibility to understand the functions as objects of today opens new doors to interpret the whole system of objects in a completely new way.

SELECT BIBLIOGRAPHY

Acunetix. (n.d.). *Web Apps*. Retrieved April 05, 2012, from Acunetix Web Application Security: http://www.acunetix.com/websitesecurity/web-applications.htm

Alberts, S. (2012, July 6). *Mobile Web Usage Statistics*. Retrieved from Business2Community: http://www.business2community.com/mobile-apps/13-killer-mobile-web-usage-statistics-0212852

American Dialect Society. (2010). *ADS: Word Of The Year*. Retrieved from ADS: http://www.americandialect.org/app-voted-2010-word-of-the-year-by-the-american-dialect-society-updated

Apple. (2008, July 8). *Apple Introduces the New iPhone 3G*. Retrieved from http://www.apple.com/pr/library/2008/06/09Apple-Introduces-the-New-iPhone-3G.html

Apple. (2011). *Gestures That Feel Real*. Retrieved from Apple: http://www.apple.com/osx/what-is/gestures.html

Archer, B. (1981). Systematic Methods for Designers.

Baudrillard, J. (1994). *Simulacra and Simulation.* Éditions Galilée (French) & University of Michigan Press (English).

Bayazit, N. (2004). Investigating Design: A Review of Forty Years of Design Research. *Design Issues (Vol. 20)*, 16-29.

BBC News. (2011, Aug 31). *Mobile Internet Use*. Retrieved from BBC News Technology: http://www.bbc.co.uk/news/technology-14731757

Berger, C., & Piller, F. (2003). "Customers as co-designers". *Manufacturing Engineer, August 2003, 82*(4), 42-45.

Berners-Lee, T. (2001). *W3C: Semantic Web Activity*. Retrieved from W3C: http://www.w3.org/2001/sw/

Berners-Lee, T. (2009, Oct 24). Web 2.0 Summit 09. (T. O'Reilly, Interviewer)

Berthon, P., Pitt, L., McCarthy, I., & Kates, S. (2007). When Customers get Clever: Managerial Approaches to Dealing with Creative Consumers. *Business Horizons, 50*(1), 39-47.

Blustein, J. (2003, April 30). *Definition of Hypertext*. Retrieved from A Part of Hypertext Materials: http://web.cs.dal.ca/~jamie/hypertext/defn.html

Boradkar, P. (2010). *Designing Things: A Critical Introduction to the Cuture of Objects.* NY: Berg.

Buchanan, M. (2003). *Nexus: Small Worlds and the Groundbreaking Science of Networks.* NY: W.W. Norton & Company.

Cambridge Dictionaries Online. (n.d.). *Software*. Retrieved 05 17, 2012, from Cambridge Dictionaries Online: http://dictionary.cambridge.org/dictionary/british/software

Campbell, S. W., & Park, Y. J. (2008). Social Implications of Mobile Telephony:. *Sociology Compass*.

Campbell, S. W., & Park, Y. J. (2008). Social Implications of Mobile Telephony:. *Sociology Compass*.

Caprio, G. (2006, 08 16). *Virtual Machines: Virtualization vs. Emulation*. Retrieved from Strange Loops:

http://www.griffincaprio.com/blog/2006/08/virtual-machines-virtualization-vs-emulation.html

Castells, M. (2000). *Rise of The Network Society* (2nd Edition with a New Preface ed., Vol. 1). Malden: Oxford.

Castells, M. (2009). *Communication Power.* Oxford University Press.

Castells, M., Fernández-Ardèvol, M., Linchuan, J., & Sey, Q. A. (2007). *Mobile Communication and Society: A Global Perspective.* London: The MIT Press.

Cervini, A. L. (2003). *Network Connections: An Analysis of Social Software That Turns Online Introductions to Offline Interactions.* Retrieved from http://stage.itp.tsoa.nyu.edu/~alc287/thesis/print.html

Chen, W. (2008). Chapter 8: Turing Machines. In *Discrete Mathematics* (pp. 83-89). Macquarie University.

ComputerHope. (n.d.). *GUI*. Retrieved 08 03, 2012, from ComputerHope: http://www.computerhope.com/jargon/g/gui.htm

Deleuze, G. (1986). *Cinema 1: The Movement Image.* London: The Athlone Press.

Deleuze, G. (1990). *Logic of Sense.* (C. V. Boundas, Ed., & M. Lester, Trans.) London: Colombia University Press.

Deleuze, G. (1991). *Bergsonism.* (H. Tomlinson, & B. Habberjam, Trans.) NY: Zone Books.

Dictionary.com. (n.d.). *Software*. Retrieved 05 17, 2012, from Dictionary.com: http://dictionary.reference.com/browse/software

Downes, S. (2005). Semantic Networks and Social Networks. *The Learning Organization (Vol 12)*, 411-417.

Duray, R., & Ward, P. T. (2000). Approaches to Mass Customization: Configurations and Empirical Validation. *Journal of Operations Management, 18*(6), 605-625.

Geertz, C. (1973). *The Interpretation Of Cultures.* NY: Basic.

Gilmore, J. H., & Pine II, B. J. (1993). Making Mass Customization Work. *Harvard Business Review*, 108-119.

Gralla, P. (2010, Feb 22). *PCWorld Tech Industry: Apple vs Google vs Microsoft*. Retrieved April 19, 2012, from PCWorld: http://www.pcworld.com/article/189963/apple_vs_google_vs_microsoft_who_will_win.html

Grillo, P. J. (1960). *Form Function & Design.* New York: Dover Publications.

Hart, C. W. (1995). "Mass Customization: Conceptual Underpinnings, Opportunities and Limits". *International Journal of Service Industry Management, 1995, 6*(2), 36-45.

Hart, C. W. (1996). "Made to Order: Technology is Making it Feasible to Reach That Market of One". *Marketing Management, Summer 1996, 5*(2), 10-22.

Hazaël-Massieux, D. (2010, Aug 2010). *DOM: What is a Wb Application ?* Retrieved from DOM: http://people.w3.org/~dom/

Houkes, W., & Vermaas, P. E. (2010). *Technical Functions: On the Use and Design of Artefacts.* London: Springer.

Hsu, M.-H., & Chiu, C.-M. (2004). Internet Self-Efficacy and Electronic Service Acceptance. *Decision Support Systems, 38*(3), 369-381.

Hvam, L., & Mortensan, N. H. (2008). *Product Customization.* Lyngby: Springer.

Ian Jacobs, N. W. (2004, December 14). *Architecture of World Wide Web: Volume One*. Retrieved from W3C: http://www.w3.org/TR/webarch/

I-Symbian. (1999). *Symbian Device - The OS Evolution.* Retrieved from Symbian: http://www.i-symbian.com/wp-content/uploads/2009/11/Symbian_Evolution.pdf

Jansen, T. (2007, March). *Ted Talks: Theo Jansen: My Creations, a New Life Form.* Retrieved from Ted Talks: http://www.ted.com/talks/theo_jansen_creates_new_creatures.html

Jansen, T. (2010). *Strandbeest: Leg System.* Retrieved from Strandbeest: http://www.strandbeest.com/beests_leg.php

Jobs, S. (Performer). (2007). *2007 iPhone Keynote.*

K.R.Rao, Bojkoci, Z. S., & Milovanovic, D. A. (2006). *Introduction to Multimedia Communications.* New Jersey: John Wiley & Sons. Retrieved from http://www.dcl.hpi.uni-potsdam.de/LV/Components04/VL1/02_middleware.pdf

Kakati, M. (2002). "Mass customization - needs to go beyond technology". *Human Systmes Management, 2002, 21*(2), 85-93.

Katz, J. E., & Sugiyama, S. (2005). Mobile Phones as Fashion Statements: The Co-creation of Mobile Communication's Public Meaning. *Computer Supported Cooperative Work (Vol. 31)*, 63-81.

Kay, A. (2008, March). *Ted Talks: Alan Kay Shares a Powerful Idea About Ideas.* Retrieved from Ted Talks: http://www.ted.com/talks/alan_kay_shares_a_powerful_idea_about_ideas.html

Koninklijke Bibliotheek. (2009). *What is Emulation ?* Retrieved 05 12, 2012, from Koninklijke Bibliotheek: http://www.kb.nl/hrd/dd/dd_projecten/projecten_emulatiewatis-en.html

Krippendorff, K. (2006). *The Semantic Turn: A New Foundation for Design.* London: CRC Press.

Law, J. (1992). Notes on the Theory of the Actor Network: Ordering, Strategy and Heterogeneity. *Systems Practice*, 379-393.

Leonhardt, D. (2000, July 28). *John Tukey, 85, Statistician; Coined the Word "Software".* Retrieved May 20, 2012, from The New York Times: http://www.nytimes.com/2000/07/28/us/john-tukey-85-statistician-coined-the-word-software.html

LIP. (2004, October 1). *The Linux Information Project: GUI*. Retrieved from Linfo: http://www.linfo.org/gui.html

Lozano-Nieto, A. (2007). *Radio Frequency Identification (RFID).*

Lyotard, J.-F. (1984). *The Postmodern Condition: A Report on Knowledge.* (G. Bennington, & B. Massumi, Trans.) UK: Manchester University Press.

Malek, M. (2002). Computer Architecture: Introduction. *Computer Architecture.* Berlin: Humboldt-Universität Zu Berlin.

Mamčenko, J. (2008). On Oprating Systems. Vilnius Gediminas Technical University.

Manovich, L. (2001). How Media Became New. In L. Manovich, *The Language of New Media* (pp. 45-48). London: MIT Press: Cambridge, Massachusetts.

Manovich, L. (2001). *The Language of New Media.* London: MIT Press: Cambridge, Massachusetts.

Manovich, L. (2006). Interaction as an Aesthetic Event. *Receiver (17)*, 67-73.

Manovich, L. (2008). *Software Takes Command.*

Margolin, V., & Margolin, S. (2002). A "Social Model" of Design:. *Design Issues: Volume 18*.

McLuhan, M. (1962). *The Gutenberg Galaxy.* University of Toronto Press.

Millikan, R. G. (1989). In Defense of the Proper Functions. *Philosophy of Science (Vol. 56, No. 2)*, 288-302.

Monge, P. R., & Contractor, N. S. (2003). *Theories of Communication Networks.* Oxford University Press.

National Public Radio. (n.d.). *How Steve Jobs Affected the Music Industry*. Retrieved 03 14, 2012, from All Songs Considered: http://www.npr.org/blogs/allsongs/2011/08/24/139929797/how-steve-jobs-changed-the-way-we-listen

Nations, D. (2010). *About: Web Applications*. Retrieved April 5, 2012, from About: http://webtrends.about.com/od/webapplications/a/web_application.htm

Neander, K. (1991). The Teleological Notion of 'Function'. *Australasian Journal of Philosophy*, 454-468.

Nelson, T. (2005, 10 24). *Indirect Documents At Last ! Now for a Humanist Computer Agenda*. Retrieved April 21, 2012, from XANADU: http://xanadu.com/trollout.txt

Nelson, T. (n.d.). *Ted Nelson's Computer Paradigm, Expressed as One-Liners*. Retrieved 04 08, 2012, from Ted Nelson - Project Xanadu: http://xanadu.com.au/ted/TN/WRITINGS/TCOMPARADIGM/tedCompOneLiners.html

Nelson, T. (n.d.). *The ZigZag® Database*. Retrieved April 15, 2012, from XANADU: http://xanadu.com/zigzag/

Norman, D. (2004). *Emotional Design.* NY: Basic Books.

Norman, D. (Performer). (2008). *UXWeek2008: A Speak with Don Norman.* Retrieved from UXweek: https://vimeo.com/2963837

Norman, D. (2011, April 8). *Stanford University: Don Norman's Complexity*. Retrieved from https://www.youtube.com/watch?v=flRuSn0df8Q

Norman, D. (n.d.). *User Experience*. Retrieved 08 08, 2012, from http://www.nngroup.com/about/userexperience.html

O'Dell, J. (2010, April 13). *Mashable Tech: Mobile Will Rule*. Retrieved from Mashable: http://mashable.com/2010/04/13/mobile-web-stats/

Oliver, M. (2007). *HKUL: Adaptive customization*. Retrieved from HKUL: http://sunzi.lib.hku.hk/ER/detail/hkul/4346255

Oxford Dictionaries. (n.d.). *Software*. Retrieved 05 17, 2012, from Oxford Dictionaries: http://oxforddictionaries.com/definition/english/software

Pachal, P. (2012, July 3). *Mashable Tech: PC and Mac Sales*. Retrieved from Mashable: http://mashable.com/2012/07/03/mac-vs-pc-sales/

Papanek, V. J. (1972). *Design for the Real World.* Pantheon Books.

Paraskevakos, & G., T. (1971, March 18). *Patent No. 3,727,003.* US. Retrieved from http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=3,727,003.PN.&OS=PN/3,727,003&RS=PN/3,727,003

Paraskevakos, & G., T. (1972, June 7). *Patent No. 3,812,296.* US. Retrieved from http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=3,812,296.PN.&OS=PN/3,812,296&RS=PN/3,812,296

Paraskevakos, & G., T. (1972, July 11). *Patent No. 3,842,208.* US. Retrieved from http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PALL&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.htm&r=1&f=G&l=50&s1=3,842,208.PN.&OS=PN/3,842,208&RS=PN/3,842,208

PCMag Encyclopedia. (n.d.). *Desktop Application*. Retrieved July 5, 2012, from PCMag: http://www.pcmag.com/encyclopedia_term/0,1237,t=desktop+application&i=41158,00.asp

Pine II, B. J., Victor, B., & Boynton, A. C. (1993). "Making Mass Customization Work". *Harvard Business Review, September/October 1993, 71*(5), 108-118.

Popek, G. J., & Goldberg, R. P. (1974, July). Formal requirements for virtualizable third generation architectures. *Communications of the ACM, 17*(7), 412-421. doi:10.1145/361011.361073

Remick, J. (2011, January 30). *AppStorm: What Is a Web App ?* Retrieved from AppStorm: http://web.appstorm.net/general/opinion/what-is-a-web-app-heres-our-definition/

Rogers, P. (2012, March 30). *GPMD - 2012 Mobile Internet Usage Statistics*. Retrieved from GPMD: http://www.gpmd.co.uk/blog/2012-mobile-internet-statistics/

S.Robins. (2012, Feb 13). *Stackoverflow: Simulation vs Emulation*. Retrieved April 21, 2012, from Stackoverflow: http://programmers.stackexchange.com/questions/134746/whats-the-difference-between-simulation-and-emulation

Science Daily. (n.d.). *Application Software.* Retrieved June 8, 2012, from Science Daily: http://www.sciencedaily.com/articles/a/application_software.htm

Searle, J. R. (1995). *The Construction of Social Reality.* London: Penguin Books.

Shedden, D. (2004, December 16). *New Media Timeline (1969-2010)*, 4. Retrieved March 20, 2012, from Poynter Institute: http://www.poynter.org/latest-news/business-news/transformation-tracker/28803/new-media-timeline-1969-2010/

Singh, K. (2009). *Security on the.* IBM.

Smith, J., & Nair, R. (2005). Virtual Machines. In J. Smith, & R. Nair, *The architecture of virtual machines* (pp. 32 - 38). Dept. of Electr. & Comput. Eng., Wisconsin Univ., Madison, WI, USA.

Smith, R. (1993). *Comdex Report*. Retrieved from ReviewsOnline: http://www.reviewsonline.com/ARTX2.htm

Socha, B., & Eber-Schmid, B. (2009). *What is New Media*. Retrieved from NewMedia: http://www.newmedia.org/what-is-new-media.html

Spriensma, G. J. (2012). *The Four-Year Anniversaryof the AppStore*. Retrieved from Distimo: http://monitor.distimo.com/app-store-analytics-blog

Stockholm Smartphone. (n.d.). *History of Stockholm Smartphone*. Retrieved April 11, 2012, from Stockholm Smartphone: http://www.stockholmsmartphone.org/history/

TechCrunch. (2012, May 4). *Foursquare Gets Its Own Searchable Timeline With New History Page*. Retrieved from TechCrunch Social: http://techcrunch.com/2012/05/04/foursquare-gets-its-own-searchable-timeline-with-new-history-page/

The Economist. (2005, May 10th). The Rise of the Creative Consumer. *The Economist*.

Thesaurus. (n.d.). *Software*. Retrieved 05 17, 2012, from Thesaurus: http://thesaurus.com/browse/software

Vermaas, P. E., & Houkes, W. (2003). Ascribing Functions to Technical Artefacts: A Challenge to Etiological Accounts of Functions. *British Journal for the Philosophy of Science (Volume 54, Issue2)*, 261-289.

VirtualBox. (1999). *Virtual Machines*. Retrieved 03 01, 2012, from VirtualBox: https://www.virtualbox.org/wiki/Virtualization

Vocabulary.com. (n.d.). *Software*. Retrieved 05 17, 2012, from Vocabulary.com: http://www.vocabulary.com/dictionary/software

W3C. (2001). *W3C: Semantic Web Activity*. Retrieved from W3C: http://www.w3.org/2001/sw/

Walker, J. (1998). *The Analytical Engine*. Retrieved April 11, 2012, from Fourmilab: http://www.fourmilab.ch/babbage/contents.html

White, R. (1994). Language of Computers. In R. White, *The Journey Inside: Digital Information* (pp. 26-52). Ziff-Davis.

Wikipedia. (n.d.). *Hypertext*. Retrieved 03 17, 2012, from Wikipedia: http://en.wikipedia.org/wiki/Hypertext

Wikipedia. (n.d.). *New Media*. Retrieved April 10, 2012, from Wikipedia: http://en.wikipedia.org/wiki/New_media

Wikipedia. (n.d.). *Virtual Machine*. Retrieved 03 05, 2012, from Wikipedia: http://en.wikipedia.org/wiki/Virtual_machine#VM_Definitions

Wiktionary. (n.d.). *Software*. Retrieved 05 17, 2012, from Wiktionary: http://en.wiktionary.org/wiki/software

Wroblewski, L. (2009, March 29). *Defining User Experience*. Retrieved from LukeW: http://www.lukew.com/ff/entry.asp?797

Yahoo Education. (n.d.). *Software*. Retrieved 05 17, 2012, from Yahoo Education: http://education.yahoo.com/reference/dictionary/entry/software

Zipkin, P. (2001). "The Limits of Mass Customization". *MIT Sloan Management Review, Spring 2001, 42*(3), 81-87.