Software Evolution as SaaS: Evolution of Intelligent Design in Cloud

Atif Farid Mohammad\textsuperscript{a}, Joumana Dargham\textsuperscript{b*}, Hamid Mcheick\textsuperscript{c}, Attia T. Noor\textsuperscript{a}

\textsuperscript{a} Department of Computer Science, University of North Dakota, Grand Forks, ND USA  
\textsuperscript{b} Department of Computer Science, University of Balamand, Lebanon  
\textsuperscript{c} Department of Computer Science, University of Quebec at Chicoutimi, Quebec, Canada

Abstract

The aim of this paper is to establish an understanding of software evolution as Software as a Service in the Cloud as a concept in comparison to Biological evolution. Software evolution is a concept that requires a deep understanding for the course of our use in the future and is an important knowledge that can bring in a change of view towards software development altogether. There are several issues of software evolvability provided in this paper. Resolution to these issues through time with software dispersion can be understood as biological inheritance. Study of biological evolution to produce compound adaptive distinction could shed the light on how software evolution can be understood in general and in meticulous understanding of evolutionary computation. We study and accumulate the divergences and possible similarities between biological and software evolution toward the end of this paper.

Keywords: Architecture, Cloud Computing, Cloud Services, Services, Service Oriented Architecture, Software as a Service.

1. Introduction

Biological evolution is a well understood and studied concept. It is an important aspect to know that this type of evolution has given us the proof of a common descent for millions of years. The process we call as a common descent means that all species share common ancestors. A well known quote from

* Corresponding author, Tel: +961-6-930250 (3973)  
E-mail: joumana.dargham@balamand.edu.lb
Theodosius Dobzhansky tells us the significance and use of biological comprehension about evolution within the life sciences that “Nothing in biology makes sense except in the light of evolution” [01]. Software as a service Evolution is considered by Lehman is most well-known for its eight “Laws of software evolution”[02]. The first two:

1. Software must continually evolve, or grow useless, and
2. The structure of evolving software as a service tends to degrade.

A software engineer initiates an artifact, which we now call software as a service on the basis of provided requirements by requiring stakeholders in Cloud. Each and every software as a service is designed, developed and later deployed to achieve goals on the basis of given requirements for a certain time period until any change is to occur within the stakeholder organization’s currently provided state. It is an eventual target of every software engineer to keep design simple as well as optimal and precise.

Section 2 contains discussions on biological evolution in relation to population and their crossover and gene mutation. Software as a service evolution and change is given in section 3 with a case study of software as a service instantiation. On the basis of these given sections a detailed discussion is given on an analysis of software as a service evolution and its understanding in section 4.

2. Biological Evolution: An Understanding

Change in living species with the pace of time due to several environmental factors is the basis of biological evolution. Living beings are adaptable to survive in accordance to the different environmental factors, such as:

- Weather
- Plantation
- Availability of food and water
- Terrain
- etc…

This evolution can also be understood on the basis of crossover and mutation. As per authors [03] of textbook of Zoology “this is a phenomenon in which exchange of chromosomal segment or genetic material occurs between the non-sister homologous chromosomes”. To make it understandable to the computing community we can also say that there is an exchange of information process happening between two individuals of same species (such as humans or animals etc.) to generate a child as a parent. A random selection is done by the crossover and operator, which selects one cutpoint from the gross information of both parents. This causes that the resulting child as an individual, which can have a different length, width and breadth than either of its parents. This study explains the evolution of life using and basing on information transmission and the main transfer of energy through living systems compositions. These compositions can be decomposed of following:

- Species and
- Population

A general understanding of population can be as a group of individuals belonging to the same species. The species is a group of organisms that can reproduce with one another and produce members of this population as described earlier in the concept of crossover and mutation [04]. Every species as an organism have a set of detailed distinctiveness, which is called as having various traits. Inheritance of living organisms is controlled by genes. A gene is a portion of DNA molecule. An entire set of genes is called a genotype and is a part of an organism’s genome. An entire set of visible traits that make up the structure and behaviour of an organism is called phenol type, like skin, hair color, height and shape of any species can be few good examples. These traits come from the communication of its genotype with the atmosphere of the species.

As per figure 1, Charles Darwin claimed in 1859 that “The origin of species by means of natural selection or the preservation of favored races in the struggle for life”. Biological evolution can also be understood as chaotic elevation and collapse of species, such as humans can be a good example of
elevation or the raise of our species and dinosaurs can be a good example of the collapse or fall of an entire set of species. There are few interesting quotes given below to simplify the understanding of biological evolution. Alfred Russel Wallace 1858 said “On the tendency of varieties to depart indefinitely from the original type” (“The Ternate paper”).

Another expert Ernst Mayr (1904 - 2005) explained the change or evolution as follows “A change in population over time of the proportions of individual organisms that differ in one, or more, genetically-determined traits”. Following figures explains single cross over.

With the pace of time, changes are introduced in any environment and new traits with several variations can appear in all of these time and several of these traits can be useful as well as useless or even harmful. Small change accumulates over time can originate new species with considerable changes, better adapted for the evolution phase of this new species environment. The study of biological evolution claims [04] that all the living species on Earth descend from a common ancestor, by means of a series of genetic variations. It is a fact that living species have not much difference; rather these do exhibit an inner continuity. However, when we study fossils, there is no such continuity can be found as claimed by Darwin, we can distinguish distinct differences in different categories of organisms. Fossil discoveries are the only available foundation that elaborates about the history of life and evolution.

Biological evolution is not just a theory; it is also an observational fact. As we discussed fossils is an observational fact that the species that exist today are not the same as the species that existed in the past. Evolution can also be understood as a scientific "theory." There can be two distinctive ways a scientific theory can be proved wrong.

- The first is to find contradictory evidence. If we hypothesize, for example, that iron can float on water, my theory will be disproved by an experiment which demonstrates that the iron do not float on water. Rather it is a fact that a ship can have tons of iron as part of it, and it floats.
- We can find an internal inconsistency as a second way to prove that the theory is not correct. If we assume that adding potato’s and tomato’s in one plantation to produce from same plant, then this theory will not be correct.

The previous discussion sheds light on a simple understanding of evolution. We can assume that the evolution is based on the thought that mutations accure over a long course of time and millions of years accumulation introduces some changes into any or all living being into a total different form of existence. These mutations are thought to drive the natural selection process, by providing the variety necessary for selection. However if you just take a look at simple gene recombination, you can see that variety comes from this mechanism, not gene mutation. The study of evolution tells us that, this reproductive change...
mechanism cannot start on its own. Mutation is explained in the following figure to establish a general understanding.

![Mutation Example](image)

**Figure 3** Visible changes after Mutation [21]

We can take an example here of mule [18] to explore the idea of a simple gene recombination leading to variety is in the large number of breeds as mule is the offspring of a male donkey and a female horse, which is classified as a kind of F1 hybrid. Where as the much rarer offspring of a male horse and a female donkey, is called a hinny. The term "mule" (Latin mulus) was formerly applied to the infertile offspring of any two creatures of different species. This proves that gene mutations contribute to this variety of breeding experimentation done, but the genetic information for the variety is already in the DNA. Next section will contains description of software evolution on the basis of biological evolution.

### 3. Software Evolution

Software Evolution can be understood as the process of developing software as a service initially on the basis of some requirements (information provided by the stakeholders requiring a solution with the help of computers), then repeatedly updating it for various reasons. As per Georgia Tech Glossary “A continuous change from a lesser, simpler, or worse state to a higher or better state [for a software system]” can be known as software evolution. Another expert Davor Svetinovic explains software evolution as the “change in the essential properties of a software product family over time”.

Software as a service is a combination of algorithms with logical conditions written in languages such as C, C++, Java and many more using databases with the use of several conditions as well as several traits. These algorithms use given grammars or we call command structure in a sequence of sentences. These algorithms evolve an illustration of the command structure of any used language to develop the required software as a service in the course of some evolutionary development procedures. This idea is based on the detail that formal command structure of any given language to code any software can easily be characterized as a tree structure as mentioned in section 1 of Darwin’s tree of life of productions. Evolutionary approaches for software as a service development using induction of several command and conditional sentences structures are based on the genetic programming by John Koza et. al [05]. Koza is one of the pioneers of genetic algorithms’ paradigm; he found analogues to the genetic operators as discussed in previous section within the standard set of tree operators and characterized Lisp programs as trees.

Software evolution can also be understood with lots of diverse advances based on evolutionary algorithms have been proposed by Javed and Bryant, et al. [07], Dupont [8], Wyard [9], Zomorodian [10], Kammeyer, and Belew [11], Korkmaz, and Ucoluk [12] and Longshaw [13]. Where as to broaden the range of understanding the study on this work the authors used an evolutionary approach based on genetic programming in Crepinsek and Mernik et al. [06] to structure of the preliminary population of software as a service.

To consider software evolution, the authors Pfleeger [15] and Kemerer and Slaughter [16] have been mainly apprehensive with the characteristics related to the phenomena of evolution, the main concern of the authors are the what and why of software evolution by Lehman’s laws [17]. The goal of this research work is to achieve a clear understanding by categorizing the attributes and convenient to our daily life evolution’s comparative impacts on software evolution on our day to day software as a service design and development processes and its deployed solutions to provide end users ease of use of the provided products together with principal drivers. To explore this idea a small case study is given below. The prescribed practices are to generate an understanding of any software evolution life cycle as its initiation to come to life stage.
These prescribed practices adaptation will give software engineer an edge to make sure to get a Biologist's perspective in the form of following:

1. Initial functionality configuration of the software as a service module to be developed for Biologist containing actual algorithms designed in the form of specifications containing all variables, parameters, any constants and formulae to get results of the analysis on either plants or animals.

2. A relationship is to be established among all requirements with the help of Biologist to make sure even before starting this software as a service module development that if there are any other changes/improvements, if needed by Biologist can be done to achieve the optimal solution.

3. Establishment of a common test-ground is an important factor to make Biologist realize that there are/will be other factors involved in the big picture of the actual software as a service. As this module is going to be a part of the big picture (Overall combination of various related modules).

4. Daily log Maintenance is an important feature and is to be clarified to all stakeholders to make everyone involved in this required solution design and development understand for the traceability towards the root start of the software as a service design in total.

5. Modules development tracking is to be done by both Biologist alongside the software as a service engineer to make sure that all requirements mentioned in point 1 and 2 are followed and are in mentioned sequence.

6. Module inspections is to be done by putting check points in the software as a service development tool before stakeholders involved to have a synergic flow in software as a service development.

7. Disaster recovery plan is to be discussed by all stakeholders to make sure all individual modules have an operational log to be maintained at some other server outside of software as a service development and operational testing zone to restart work, in case of a disaster happens during this software as a service development process.

8. Improve solution safety is an important factor involved in each and every module individually to make sure all modules generate data, which should be hack proof and is available to designated users (Operational users and/or scientists).

9. Reliability check can be understood as point 6 repetition but at the end of each module completion before the delivery to the actual combination of the modules.

10. Code Quality Check is the last and most important factor and is the total responsibility of the software engineer after passing through all points 1 to 9 to make sure that software as a service is crash proof to achieve the goals of this final product, which will be a combination of all scientific as well as operation control software as a service.

11. Whenever any required software as a service by any stakeholder(s) or a system is released to software engineers and goes into the life cycle mentioned above. All other features enhancements in the future will be put in next major releases of this software as a service. From time to time after the some major release is completed, the software as a service becomes a bit more different from the first release. This process can be considered as software evolution. In general it is taken as software as a service evolvability, which happens gradually through the change with the pace of time of its utilization.

4. Rants and Raves

In a remote town of Philadelphia, in a cold December night with 4 meters of snow, a computer running Microsoft Windows 7 accidentally connected to one of the Linux Server in Jet Proportion Laboratory (JPL) of NASA working under the wing of Army Confidential Research Project.
This uncommon “Cross-Over” generates a program, that is one of its kind, never seen before, intelligent, adaptive and aggressive. Program has the great immune system against viruses. This program can decide which command to follow or reject by humans. There are some points given below to ponder.

A) A function was updated by an undergrad student at Florida Tech should we call it evolution? This certainly is not evolution as it has been a change to a certain function, this might give us incorrect data in the end, and the sequence of input-process-output will remain the same.

B) If one of us changes the hair color to Red, would it be called evolution? This change will not be transferred in our children as it has nothing to do with our inheritance.

5. Equivalency between Human and Software

There is an interesting question arises after all of this biological and software evolution discussion: What is the equivalent of “Humans” in Software?

- Operating System?, Applications?, Programs?
- Functions?, Classes?, Modules?

In the following discussions, we try to find the commonalities of human traits with any given software as a service as mentioned above.

An operating system/OS is typical software as a service designed to make sure the hardware (computer) is working according to its defined capabilities and OS [20] is also to run other programs on a computer. We can not find any crossover and mutation here to get further evolved as specie of its kind.

A software as a service application is a program or group of programs designed for anyone such as OS can use an application in its batch processing to maintain certain feature available for the end user to utilize, without separately executing it. An application such as Facebook can be an example, is used for our virtual community creation etc. Facebook was intended as an application to run as a forum for student interaction and information flow on college campuses. Since Facebook opened to the general public, it has grown to several millions of active users [14]. We can see other similar web applications, once again to reiterate there is no genetic change can happen here in relation to our human genome system.

A program or programming paradigm [18] is a set of an organized list of commands of a language’s designated grammar that, when executed, causes the computer to behave in a preset way. It is simple fact that if there is no OS available, physical computer will not work. There are many programming languages available with defined grammar styles of writing instruction sets in a sequence such as C, C++, Pascal, BASIC, FORTRAN, COBOL, and LISP are just a few. A tree structure can be formed of the languages as given below:

![Figure 4 A Computer Programming Language Tree](image)

A function is a tenet of a language defined with two or more variables. These variables are to store the input values to output a certain result. For example a Sum(x,y) can be a function to give us an answer of 3 if x has a value of 1 and y has a value of 2. A change in a function can obviously happen, there is no way that this function of Sum mentioned above can further evolve.

A class can be understood or compared with biological evolution discussion. An Object-oriented language allows a software engineer to define classes [18] that can generate objects having characteristics and methods or we can call these ways of working of the derived object that are associated to the parent class of the child object(s). For example, we can consider a class of vehicles, a car, a bus and a train can be few good examples to understand. As vehicle goes from a point A to point B and has a certain shape to carry
people. This type of instantiation happens in most of the species and the properties are combined of two same species to produce a third of same with all or few same characteristics of theirs own to evolve.

Modules are subroutines in a code to achieve a given target to either calculate some factors of the input or to pass internal values to other modules for hybrid results, if required by the users.

7. Conclusion

This paper establishes an understanding of software evolution as Software as a Service in the Cloud as a concept in comparison to Biological evolution. Software evolution is a concept that requires a deep understanding for the course of our use in future and is an important knowledge that can bring in a change of view towards software development altogether.

We survey several issues of software evolvability based on biological inheritance. This survey accumulates the divergences and possible similarities between biological and software evolution toward. In the future research, we will apply this approach with a large case study and to develop a framework based on the comparison of human and software.

References


