THE ROLE OF AESTHETICS IN SOFTWARE DESIGN, DEVELOPMENT AND EDUCATION

Working Paper · April 2016

DOI: 10.13140/RG.2.1.3684.6488

Latchezar Tomov
New Bulgarian University
95 PUBLICATIONS **32** CITATIONS

CSECS 2016, pp. 000 - 000

The 12th Annual International Conference on
Computer Science and Education in Computer Science

July 04-07 2016, Fulda, Germany

THE ROLE OF AESTHETICS IN SOFTWARE DESIGN, DEVELOPMENT AND EDUCATION

Latchezar Tomov, PhD

New Bulgarian University - Sofia, Department of Informatics

***Abstract**:* Aesthetics in software has the same role as in science in general – it is a powerful tool for motivation of the developer, a symptom of expert knowledge, a way to measure quality. You can hire people, based on it, you can estimate directly or indirectly the quality of software and of the process of development and you can teach it to help student reach the expert level earlier.

***Keywords**: aesthetics, software quality, (Keywords are your own designated keywords).*

***ACM Classification Keywords**: A.0 General Literature - Conference proceedings*

Introduction

Computer science is science, but computer programming is an art and as such it requires some notion of aesthetics. The etymology of the world is from Latin “ars”, which means “skill” [Knuth 1974]. A skill is practiced by humans and aesthetics is describing their relationship with the developed products and services. The human – computer interaction is complicated one, where the languages radically differ and communication can exists only within frames of some compromises. Human readability of machine code is low, while computer needs translation of higher level programming languages code into bytecode. Furthermore, software engineers exist in a market environment which actively shapes the nature of that relationship not only by putting constraints on time and effort, but as a driving source of continuous and gradual improvement of product quality [Tomov and Ivanova 2015a]. In this paper we want to show that not only the notion of aesthetics is natural for natural sciences but is also very informative, both for the people that practice them and software development in particular and for the quality of what they accomplish. Aesthetics is related to motivation, which means that is useful for hiring people in the industry, for judging their commitment. It is a sign of expert knowledge which comes with many thousands hours of dedicated practice. It is a tool to overcome the problem of induction and increase software reliability. It is linked to quality and is possible to be measured, directly and indirectly.

Definitions and history of aesthetics in science

Def1. Aesthetics is the philosophical study of beauty and taste [Britannica, 2009]

Def2. [Bychkov, 2010] The ancient themes and problems that have been traditionally considered “aesthetic” are as follows:

- the sense of beauty and awe before certain natural and artistic forms, which lacks any rational explanation and yet is a source of great pleasure and seems to point to values and truths that transcend the human mind;
- the whole area of sensory experience that brings us the feelings of beauty and awe; the area of human production that we call the “fine arts” or the production of aesthetic objects;
- A number of themes and issues associated specifically with the human artistic activity, such as imitation; various literary and rhetorical techniques; and principles of tone, contrast, harmony, and composition in painting and music.”

This definition is not complete and does not recognize the sense of beauty in science [Girod, 2006], which developed simultaneously with its deductive and logical foundations [Luminet, 2011]. We call it “artistic aesthetics”

Def.3 Intellectual aesthetics:

- the sense of beauty and awe before certain natural forms and relationships, as expressions of order and truth following from rational explanation; and yet a source of great pleasure and seems to point to values and truths that originates from the human mind;
- the whole area of sensory experience that brings us the feelings of beauty and awe; the area of human production that we call

the “fine arts and sciences” or the production of aesthetic objects;

- A number of themes and issues associated specifically with the human intellectual activity, such as deduction and logical reasoning; various theory proving techniques; and principles of vastness, simplicity, efficiency, harmony, and composition in mathematics, logic and crafts.”

Intellectual aesthetics cannot be ignored because it is part of science and education from its beginnings in Greece. The Greek paideia incorporated education in both arts (rhetoric, composition, philosophy) and sciences (mathematics, physics, medicine) [Jaeger, 1945]. The developed mind for the Greeks appreciated art and understood science. The academy of Plato had engraved “Let no one ignorant of geometry enter” at its door [Suzanne, 2011]. Geometry and its achievements played central role in forming the intellectual aesthetics of the classical era [Luminet, 2011]. Geometry as a language of beauty and harmony and as a driving force for evolution of mathematics [Yushkevich et.al., 1970] is the early medium that connects order and logic with beauty and aesthetics. Greek mathematics took the root of geometry when Pythagoreans faced the problems of incommensurability of sections and inability to be quantified with ratios and rational numbers. Thus, geometric algebra appeared. Geometry had profound influence over philosophy and other arts, as seen in Plato’s ideas about spherical structure of the Universe, consisting of two worlds – the superlunary world of perfection, expressed by the harmonious, symmetrical and uniform shape of the sphere and the sublunary world of regular polyhedrons - less perfect and less aesthetic shapes [Luminet 2011]. The ideas of Plato and Aristotle (the geocentric model of the world) remained central to the science up to 16th century until Nicolaus

Copernicus replaced that model, with heliocentric. The aesthetic view of the order of the Universe via spheres and polyhedrons was idealistic and incomplete. Ironically, his motivation for the change was also aesthetic: "In this arrangement, therefore, we discover a marvelous symmetry of the universe, and an established harmonious linkage between the motion of the spheres and their size, such as can be found in no other way" [Copernicus, 1543]. Kepler was also influenced by Platonic solids, which he used for construction of his model of the Solar System in search for the geometric order in the Universe as part of God's plan [Caspar, 1993]. Geometric interpretation of abstract ideas was core of science for history of the Western world and continued to be valuable in modern physics and mathematics from 19th, 20th, and 21st century as discussed in the next section.

Aesthetics in mathematics and physics

Platonic ideas influenced modern physicists, such as Niels Bohr, who built his model of the atom based on aesthetic feeling and was called by Einstein "the highest musicality of human thought" [Gustafson, 2004]. It used ideal shapes – spheres, for their aesthetic appeal of simplicity, harmony and symmetry. This is a case where aesthetic feeling can be misleading, and the process of representation of reality with platonic forms and models can lead to too much simplicity, not corresponding to empirical data, a "platonification" [Taleb, 2010]. There are two different aspects of aesthetics and notion of beauty in mathematics and physics – algebraic (symbolic) and geometric and with development of modern science the second decreases its priority [McAllister, 2005]. Geometric aesthetics is related to visualization of theories and is close cousin of aesthetics in art, while algebraic or symbolic aesthetics is related to the quality of theories and their abilities to uncover truth with simple symbolic expressions. Simplicity and vastness are expressed by Poincare [Chandrasekar, 1987]:

*"It is because **simplicity and vastness** are both beautiful that we seek by preference simple facts and vast facts; that we take delight, now in following the giant courses of the stars, now in scrutinizing with a microscope that prodigious smallness which is also a vastness…"*

Contrary to the opinion, expressed in [McAllister, 2005] there are objective measures of aesthetics that have little to no change for the 2500 years of scientific revolution and are related to the way the human brain works. "**Simplicity and vastness**" express that. We will further discuss how to measure aesthetics and what aesthetic feeling means for the work of the scientists and or engineers and for the quality of knowledge they posses. A prime example to that description is the Euler's identity (1)

$$e^{i\pi} + 1 = 0 \tag{1}$$

Richard Feynman called it "*our jewel*" and "*the most remarkable formula in mathematics*" [Feynman, 2011]. Keith Devlin, professor of mathematics in Stanford has said:

"…like a Shakespearean sonnet that captures the very essence of love, or a painting that brings out the beauty of the human form that is far more than just skin deep, Euler's equation reaches down into the very depths of existence" [Nahin, 2006]

It connects five of the most important numbers in a single line with just two operators, low complexity. Furthermore, it is an expression of profound connections between algebra and analysis. It is a prime example for the validity of the explanation of aesthetics which Poincare made. Furthermore, analysis of brains of mathematicians shows that the same region that is active when experiencing beauty in art activates when experiencing mathematical beauty and activation is strongest for (1) [Zeki et. al, 2014]. Aesthetic inclinations, interests and abilities in art are not uncommon among the top scientists, especially Nobel Prize winners, 65 of them [Root-Bernstein, 1989]. Einstein, Feynman and Heisenberg were musicians, Feynman also painted very well and he had success with music too. Marie Curie, Robert Oppenheimer (winner of Fermi award) and Ludwig Boltzmann wrote poetry. Cedric Villani compares mathematics with music, based on his experience in that area. It is argued by [Root-Bernstein, 1997] that science and art shared common underlying aesthetic motive and theory, which is not unfounded in the light of results from neurobiological research. Aesthetic motives were used for evaluation of theories in physics – an example is Paul Dirac and his objection on renormalization techniques in quantum electrodynamics [Kragh, 1990]:

“Most physicists are very satisfied with the situation. They say: "Quantum electrodynamics is a good theory, and we do not have to worry about it anymore." I must say that I am very dissatisfied with the situation, because this so-called "good theory" does involve neglecting infinities which appear in its equations, neglecting them in an arbitrary way. This is just not sensible mathematics. Sensible mathematics involves neglecting a quantity when it is small - not neglecting it just because it is infinitely great and you do not want it!”.

Paul Dirac has expressed his conviction that the value of theory lies beyond simple agreement with experiments [Girod, 2006]:

"It is more important to have beauty in one's equations than to have them fit the experiment"

Teaching aesthetics

This concept we see applied in the case of quantum electrodynamics, but it has deeper roots – in the problem of induction [Hume, 1910]. In the scientific context the problem is as follows:

Def.4 Problem of induction – Arbitrarily large amount of data that supports a theory can be outweighed by a single piece of data that rejects it.

History of physics is filled by short lived theories, replaced with new on that basis. Scientific reliability thus cannot be completely assured by any amount of observations. This is where mathematical beauty has its role – a more coherent and complete theory has higher probability of surviving longer period of time or just being "modified" instead of rejected completely. This directly related to computer science and software reliability being dependant of software design principles.

Algebraic aesthetics or mathematical beauty as in (1) is an attempt at solving the problem of induction. **Geometric aesthetics** is an attempt at increasing scientific understanding and helping scientists to escape what Richard Feynman called "**fragility of knowledge**" – the inability to apply theoretical knowledge to real world problems due to lack of deep understanding. Examples of visual, geometric aesthetics are the Feynman diagrams originally invented to help with QED and renormalization, but are used to describe behavior of subatomic particles in a simpler, easier to comprehend way. They also help speed up calculations by replacing multiple lines of algebra and hundreds of distinct terms with intuitive and simple approach. It is combining useful for teaching power of expression with the practical side of computation [Kaiser, 2005]. These two examples contradict, since renormalization was objected on aesthetic terms, while the diagrams were created with aesthetic motives, along the practical considerations. The use of tricks for ignoring infinities in algebraic form was deemed unaesthetic by Dirac, while the diagrammatic calculations that did the same are aesthetically founded, as they bring understanding, clarity and ease of computation. To solve such problems in computer science, we formulate:

Def 5. **Principle of precedence** – The aesthetic of an entity precedes the aesthetic of its presentation.

According to this principle, an entity cannot be considered aesthetic solely on its presentation (a geometric description of itself).

In the next part we will see how the ideas from mathematics and physics translate to computer science and what is achieved so far. The latter is younger than the former two branches of science and can borrow ideas from them.

Aesthetics in computer science

The role of aesthetics in computer science is not different than in mathematics and physics from which it is derived with the help of Alan Turing and Johnny Von Neumann. In the seminal paper, A.P. Ershov points several different aspects [Ershov, 1972]:

1. Aesthetics of programming as a process, the pleasure that the developer obtains from solving complex problems for which both high level of mathematical knowledge and purely engineering talent are need and requirements for accuracy are

highest from all engineering fields. This is the driving force of his intrinsic motivation.

2. Programming as creative job, close to mathematics and creative writing is naturally bound to aesthetic values and judgments about the products of it.
3. Aesthetic aspects related with programming as a social and public functions, the perspectives for coding skills as a requirement for literacy in long term (programming is the new literacy) and the social status quo of the programmer (the last is important in Eastern Europe and former Soviet Union)

The influence of A.P.Ershov in this topic is profound – Donald Knuth in [Knuth, 1974] analyzed the differences between art and science and discussed the place of art in engineering, a.k.a. “the crafts”. The need of developing “taste” and “style” in the art of programming is stressed by him, as he draws on from earlier sources like [Dijkstra, 1971]:

“*When I speak about computer programming as an art, I am thinking primarily of it as an art form, in an aesthetic sense. The chief goal of my work as educator and author is to help people learn how to write beautiful programs.”*

Aesthetics for him is related to the experience of both programmers and users, therefore closely connected to software quality (program correctness, performance, usability, readability, maintainability). One more point stressed in this article is the importance of style in design of programming languages and as we will see in next chapters, aesthetic motives have main role in evolution of languages from lower to higher level of abstraction (case in point – the evolution of assembly language into C and then in C#).

Research in the field of aesthetics and its applications are in three different directions:

- ➢ Taste and style in coding
- ➢ Aesthetic computing
- ➢ Elegant software systems design

"Taste and style" are concept related to the work, started by Donald Knuth in [Knuth, 1992] and Brian Kernighan in [Kernigan and Plauger, 1978], later followed by [Stepanov and Marcus, 2004]. Knuth teaches his students how to write code as literature, or a text-based approach, in which the program serve as documentation by itself. The goal is to minimize the distance between the formal language and the natural one. This is a "*text-based aesthetic*" [Fishwick, 2000] or as we define in this article, **algebraic aesthetics in programming**. This is a concept related to estimation of aesthetics in objective way which we consider foundation for our current and future research and will use in next parts of this article to define complexity.

Aesthetic computing is treated as *embodied formal language* in [Fishwick, 2013] which is an example of **geometric aesthetics in programming**:

"*The Aesthetic Computing Hypothesis is that given the embodied nature of cognition, we should realize this embodiment through novel human-computer interfaces for learning formal languages.*"

The author teaches course in aesthetic computing in which the direction of work is reversed – instead of using software to develop aesthetic artifacts, he uses aesthetic artifacts to develop software. The idea is not completely novel; LOGO language is an example of such idea [Papert, 1980], quoted in [Fishwick, 2013]:

"We have stressed the fact that using the Turtle as metaphorical carrier for the idea of angle connects it firmly to body geometry”

This approach for teaching has its merits and is similar to the way Feynman taught physics [Feynman, 2011], in order to develop “**non-fragile knowledge**”. It is one possible solution to the problem of understanding abstract knowledge. Better solution is to teach students to think in abstract realm first, to achieve one day expert level where they can understand it on intuitive level and can generate geometric aesthetic embodiments on their own. Visual models are embodiments of concepts, thus a subset of them. The relationship is not bijective and using the representations to form understanding for the concepts is very useful and practical, but incomplete. Algebraic and geometric aesthetics together form dynamic systems in which understanding of abstract knowledge leads to aesthetic felling and to generating aesthetic embodiment with geometric models; they increase the understanding of the abstract knowledge which is the feedback in this system (Figure 1.). Graphical representation of scientific knowledge like this is also a meta-example. Our goal is to develop algebraic aesthetics, which is complementary to this approach and is related to **elegant systems design** [MacLennan, 1997-2006], [Madni, 2012]. Elegant design is directly linked to different aspects of software quality in these works, and the idea, that in structural design **function follows form**. This is valid especially for the design the user can see. We are interested in what the user **cannot see**, since we want to escape from the problem of decision – making, called “judgment based on availability” [Tversky and Kahneman, 1974], on which only what is visible to a person is counted by him or her, because is easier to be recalled.

Madni distinguishes two types of elegance:

- Systemic elegance – related to actual **systemic complexity**
- Perceived elegance – the elegance, showed to the user by hiding systemic complexity from the user

We shall use those two definitions for the internal design, by enlarging the vector of users with the addition of programmers. Object oriented programming is example of perceived elegance which hides systemic complexity from the user. In general the "level" of programming languages is associated with the increase of perceived elegance as long it is minimizing the distance between formal and natural languages.

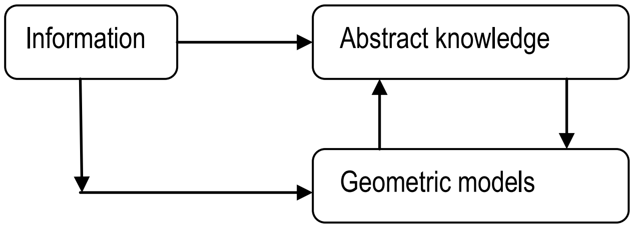

Figure 1 *Abstract knowledge and geometric representation as dynamic system*

Aesthetics and the expert mind

Aesthetics in computer science and software development has two main applications:

- It gives us information about the quality of software design, both internally and externally, from interaction with users and programmers.

- It gives us information about the people who are involved in the field, who experience this feeling.

As we saw, many of top scientists in mathematics, physics and computer science have talent for the arts and appreciate the aesthetic aspects of their research and development. As mentioned before, experiments show that their reaction to beautiful equation is the same neurologically as the reaction of artists and musicians, enjoying beautiful painting, sculptures and music. Professionals on the highest level often have taste for aesthetics, but can we use aesthetic appreciation as a sign or symptom of the level of knowledge, experience and understanding in science and engineering? Aesthetics and science are linked through intuition [Sinclair, 2004]:

"*From this perspective, which I adopt, a student's aesthetic capacity is not simply equivalent to her ability to identify formal qualities such as economy, unexpectedness, or inevitability in mathematical entities. Rather, her aesthetic capacity relates to her sensibility in combining information and imagination when making purposeful decisions regarding meaning and pleasure*."

Aesthetic forms with advancement of knowledge and achieving expert status. As [Sinclair, 2014] points:

"…*surely the mathematician's appreciation of structure, closure and order is a function of their mathematical knowledge, experiences and social acculturation*"

Experts have intuitive understanding or problems for which they have trained in predictable environment with regularity and fast feedback from the consequences of their decisions [Kahneman and Klein, 2009]. These skills are learned through machine learning or the "System 0" [Dreyfus, 2014] which does not think, but "knows how" and are automatic. Aesthetics experience is emotional experience, result from intuition [Root-Bernstein, 1997], [Burton, 2004]. Software development is such an area in which experts are formed, just like mathematics and physics and some of them can be recognized by their aesthetic appreciation for their work and the value of beauty they assign to it. Furthermore, aesthetic appreciation is a sign of motivation for continuous improvement of both code and education – the quest for elegance is one for optimality, since in **algebraic aesthetics** beauty is recognized as perfection. Desire for beauty leads to striving for excellence. Aesthetic feeling is an intrinsic reward for programming efforts and can significantly accelerate goal-related learning and external reward intake [Schmidhuber, 2010]. It creates conditions for intrinsic motivation and self-supporting process of lifetime education. Learning trough experience with emotional feedback is important [Dewey, 1997] and aesthetics is such a feedback. Furthermore, the advantage of expert chess players is in the first few seconds of thought [Ross, 2006] and intuition plays key role in expert knowledge.

Thus we can make the following suggestions how this aesthetic feeling can be used to manage human resources:

- Appreciation of beauty and elegance in internal design is a sign of expertise and a valid criterion for filtering candidates. It has to be remembered however, that **lack of appreciation of beauty is not proof for lack of expertise**.
- Appreciation of beauty and elegance is strong sign of intrinsic motivation which is another valid criterion for filtering candidates.

- Human recourse staff and managers should try to develop this aesthetic feeling during firm education and should cherish and support a culture of appreciation of beauty. The same suggestion is valid for academic institutions and the teaching course in Aesthetic Computing in Florida is an example for that, in the case of **geometric aesthetics,** as well as the Literate programming lectures by Knuth, as example **in algebraic aesthetics.**

Measuring aesthetics

As much as aesthetic tells us about people, it also tells us about programs. Just like in mathematics and physics where the beauty of an equation and theory is the amount of order they express with as little symbols as possible and speaks for their validity, in programming aesthetics is related to the order that a function or an algorithm (or object) imposes with as minimal complexity as possible thus is related to reliability, readability, maintainability. If we can measure aesthetics, we can automate the process and give another tool to developers of programming and programming languages that can guide not only the first writing of a code, but its evolution.

The first mathematician, who tried to measure aesthetics, is George D. Birkhoff [Birkhoff, 1933]. He measures geometric aesthetics as a simple ratio between order and complexity (2)

$$M = \frac{O}{C} \tag{2}$$

Order in this case is a notion for symmetry and harmony in the visual sense. For **algebraic aesthetics** we will use the notion of **simplicity and vastness** as expressed by Poincare [Chandrasekar, 1987]. Since simplicity is the inverse of complexity, we can substitute and derive:

$$M = \frac{V}{C} \tag{2}$$

This measure can be used **explicitly** with computing numbers or **implicitly** with comparing programs of different vastness and complexity.

The next step is to try to define vastness for computer programs.

Def 6. **Vastness** of a function as defined in [Manev, 1998] is the sum of cardinality of its domain and co-domain.

Def.7 **Computer program** is a system of functions, where system is defined as:

Def.8 A **system** is a network of interdependent components working to achieve common goal.

From Definition 7 it follows that the vastness of a program is at least the sum of vastnesses of its functions.

Functions with increased generality of both data types and operations have larger vastness. Function that finds the complex roots of a polynomial with real coefficients for example has larger vastness than a function that finds only the real roots. Function that can calculate determinant of matrix of specified size of functions in symbolic algebra and gives analytical expression have larger vastness than numeric function that finds determinant of matrix of the same size where the numbers are concrete values of the functions. Function that calculated the determinant of matrix of real numbers is more general than a function that calculates it for integers.

An open question here remains how to define metrics for vastness, since cardinality in set theory is usually infinite number, like the set of natural or real numbers, and explicit assignment of “size” of a set when the set is not finite is a task to be done. Cardinality, however, can be compared and all domains and co-domains of programming functions can be compared. Two or more programs with the same vastness and different complexity have different aesthetics. For finite cardinality we can compare programs with different cardinality and complexity. An open question is the comparison of programs with different infinite cardinality and complexity.

Another open question, despite the substantial work done in the field [M. Debbarma et.al, 2013] is what software complexity is and how it is measured. There are different approaches to solve that problem

- Probabilistic or statistical approach.
- Cognitive approach, or what the human mind consider complex.

The first probabilistic definition of complexity of an object is from Andrey Kolmogorov [Kolmogorov, 1998], [Fortnow, 2001] as the shortest computer program in given language that produces the object as output. It is maximized for random strings as objects and is closely related with information entropy. Due to incomputability of this measure, other measures are needed. One such a measure is **long range power law correlation**, used in [Kokol et.al, 1999] to compare computer and human text. Since texts written in natural languages are more complex than programs in computer languages, the idea of Knuth for **text based aesthetics** in which the distance between a program and the equivalent text is minimized (and the distance is measureable in this metrics) is controversial with the statistical approach and shows that for humans the minimal statistic complexity and the minimal **cognitive complexity** differ and decreasing statistical complexity under some value will make a program harder to comprehend.

Def.9 **Cognitive complexity of a program** – the mental effort, required to understand a program.

Here are the measures, related to **program volume** which does not capture the problem of level of abstraction a language uses or the layers of readability [Tomov and Ivanova, 2015b]. Here is **cyclomatic complexity** which is useful for testing purposes since it estimates the number of linear independent paths in the control flow graph of the program but says little about volume or level of abstraction. In the search of measure of aesthetics we need to develop ways to capture all properties of the cognitive complexity in a formula or algorithm that is both complete and computable. This is an open problem in the field of software complexity.

Here we need to distinguish some types of complexity to extend the notion of Madni:

Def.10 **Perceived complexity** – the complexity of the abstract layer of program code with which the developer directly interacts.

An example is a programming language of higher level like C# in which the developer has no direct interaction with memory addresses and address arithmetic and has many functions for data manipulations compressing a lot of complexity in one or two words; another example is SQL, which is very close to the natural language. Programs written in such languages have low perceived complexity for the same functions, realized in lower level languages. For the sake of measuring aesthetics we are interested in perceived complexity.

Def.11. **Actual complexity of a program** – the complexity of the operations done by the program on the level of binary code.

Def.12. **Local complexity of software system** – the complexity of a program.

Def.13. **Systemic complexity** is the complexity of the software system. – It is an important notion, since efforts in refactoring code can decrease complexity on one location by shifting it to another thus not solving the bigger issue.

In order to have meaningful results from measuring and controlling software aesthetics, we need to compute or compare the aesthetics of every program or module that we affect with rewriting.

Teaching aesthetics

Long term social and economic development is in the direction of ever growing automation of work and the usage of software in every part of the industry, education and services. The demand for software developers will continue to grow in the foreseeable future, while the supply will remain limited by the state of school education in mathematics in which only few are excellent students. Although this is not engraved in stone and quite possible to change [Mighton, 2008], [Willingham, 2009], we cannot wait and count on that. The solution is to help interested students achieve expert status earlier and to increase the probability for them to do so. The first goal can be achieved with teaching intuition both in school [Ben-Zeev and Star, 2001] and university in mathematics and computer science. The second goal can be met with teaching the appreciation of beauty that may come with expert knowledge. This is a dual task, comprised by the subtask of teaching beauty and the subtask of teaching the relation of beauty to computer programs and functions, for the algebraic aesthetics. Geometric aesthetics or “aesthetic computing” is already taught as we mentioned before and the practice can be embraced by other institutions.

Teaching beauty is best done in early childhood with practice of music and drawing as the first enhances IQ as well [Schellenberg. 2004]. This is a task primary for the school, but not only for it; the brain is a complex system that can form new intelligence throughout the life of a person [Sternberg, 2008], [Buschkuehl and Jaeggi, 2010.

Teaching expertise is possible with focused practice and the "ten year rule" applies. [Ross, 2006], [Willingham, 2004]. Ten years of heavy labor are needed **on average** to achieve expertise. Universities and institutes should therefore start students on extensive practice from the first year of their education until graduation with a structured and possibly iterative approach to acquire knowledge and understanding. The goal in mind is to increase the efficiency of practice beyond the trial-and-error algorithm extensive labor provides in the market with its time and money constraints. This can be done with some guidance to accelerate convergence towards optimal solutions of algorithmic and software engineering problems. Instead of directly presenting the best design pattern or algorithm, the educator can guide the students to develop it in several iterations thus giving practice time. This is also an excellent opportunity to teach algebraic aesthetics by comparing solutions from different iterations both visually and functionally. We have used similar approach to teach good software practices by refactoring during a course in which they have to produce commercial software [Tomov and Ivanova, 2014]. The next logical step is to propose such an algorithm with added training of the feeling of "beauty":

```
Step 1. Problem definition.
Step 2. Problem solution by students, unguided
in the first iteration.
Step 3. Automatic testing and simulation of
usage for a predefined period of time, chosen by
the teacher, in respect with the complexity of
the problem - more complex problems requires
more testing and larger period of
simulation.
```

```
Step 4. Analysis of bugs and performance. Compare current problem solution with previous solution, by predefined measures. IF satisfactory level is achieved, GO TO Step 6.
Step 5. Unguided or guided refactoring by the choice of the students. GO TO Step 2.
Step 6. Compare initial and final problem solutions, by predefined measures.
```

Algebraic aesthetics is trained by explicitly associating the functionality with beauty on each comparison and implicitly – by the act of comparison of older and new cleaner, simpler and shorter program code.

Conclusion

The role of aesthetics in computer science is much like the role of aesthetics in mathematics and physics

- it is a sign of expert knowledge and intuition;
- it drives intrinsic motivation and stable process of self-improvement in long term as an emotional award strategy
- It is a way to increase understanding and accelerate achieving expertise both with algebraic and geometric versions.
- It is a way to measure software quality, related to complexity, readability, and maintainability (an open for research field)

Beauty of a formula, a theory or a program is in general the same thing – the feeling of satisfaction from completeness, vastness, strong logic and harmony of relations. Many of the brightest minds in these fields have the same appreciation for it as artists have for their work and strive for the same perfection. Software engineering and computer science are relatively new in engineering and science fields and can use more rigors, and therefore, more beauty, to achieve the quality of products other engineering fields have and the beauty of theories as profound as the general relativity theory or abstract algebra. It is very practical to appreciate human appreciation of beauty and to try to use its power for innovation and production of *state of the art* software.

Bibliography

[Ben-Zeev and Star, 2001] Talia Ben-Zeev, Jon Star, Intuitive Mathematics: Theoretical and Educational Implications, Understanding and Teaching the Intuitive Mind: Student and Teacher Learning, Routledge (January 1, 2001).

[Birkhoff, 1933] George D. Birkhoff. Aesthetic Measure. Cambridge, Harvard University Press, 1933

[Britannica, 2009] Encyclopedia Britannica. 8th ed. 2009

[Bychkov, 2010] Oleg V. Bychkov, Greek and Roman Aesthetics, Cambridge University Press, 2010.

[Buschkuehl and Jaeggi, 2010] Martin Buschkuehl, Susanne M. Jaeggi,Improving intelligence: a literature review, Swiss Med Wkly 2010 ; 140(19–20):266–272 , http://www.isdbweb.org/documents/file/4bfd263b26c95.pdf

[Burton, 2004] Leone Burton, Aesthetics, Intuition/Insight and the feelings associated with mathematics, Mathematicians as Enquirers Volume 34 of the series Mathematics Education Library pp 63-89, 2004

[Caspar, 1993] Caspar, Max. Kepler; New York: Dover, 1993. ISBN 0-486-67605-6

[Chandra, 2014] Vikram Chandra .Geek Sublime: The Beauty of Code, the Code of Beauty, Graywolf Press; 1st edition (September 2, 2014)

[Chandrasekhar, 1987] S.Chandrasekhar, Truth and beauty: Aesthetics and motivations in science,1987. The University of Chicago Press.

[Copernicus, 1543] N. Copernicus, On the Revolutions of the Celestial Spheres, I, 10, trans. E. Rosen, John Hopkins University Press, 1992.

[M. Debbarma et.al, 2013] Mrinal Kanti Debbarma, Swapan Debbarma, Nikhil Debbarma, Kunal Chakma, and Anupam Jamatia A Review and Analysis of Software Complexity Metrics in Structural Testing, International Journal of Computer and Communication Engineering, Vol. 2, No. 2, March 2013

[Dexter et al., 2011] Scott Dexter, Melissa Dolese, Angelika Seidel, Aaron Kozbelt./ On the Embodied Aesthetics of Code. Culture Machine, Vol 12 (2011) ISSN 1465-4121 http://www.culturemachine.net

[Dewey, 1997] John Dewey, Experience and nature, 2nd ed, reprinted 1997. Chicago: Open Court.

[Dijkstra, 1971] Edsger W. Dijkstra, EWD316: A Short Introduction to the Art of Programming. T. H. Eindhoven, The Netherlands, Aug. 1971.

[Duch, 2012]? Włodzisław Duch, Mind-Brain Relations, Geometric Perspective and Neurophenomenology, APA Newsletters;Fall2012, Vol. 12 Issue 1, Special section p1.

[Dreyfus, 2014] Stuart E. Dreyfus- System 0: The overlooked explanation of expert intuition, Handbook of Research Methods on Intuition, Edward Elgar, 2014

[Ershov, 1972] Andrei P. Ershov, Aesthetics and the human factor in programming Communications of the ACM Volume 15 Issue 7, July 1972 Pages 501-505

[Faye 2006] Jan Faye, The Role of Cognitive Values in the Shaping of Scientific Rationality, 2006, http://philsci-archive.pitt.edu/3738/

[Feynman, 2011] Richard P. Feynman, The Feynman lectures in physics, 2011, Basic Books

[Feldman, 2014] Stuart Feldman, A Methodology for measuring elegance in engineered artifacts, PhD Thesis, 2014 www.stuartfeldman.com/s/Feldman-Dissertation.pdf

[Feldmand and Crutchfield, 1997] David P. Feldman, James P. Crutchfield Measures of Statistical Complexity: Why? Physics Letters A Volume 238, Issues 4–5, 9 February 1998, Pages 244–252

[Fishwick, 2000] Paul A. Fishwick, Aesthetic Programming, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.8539

[Fishwick, 2013] Paul A. Fishwick, Aesthetic Computing, Encyclopedia of Human-Computer Interaction, https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/aesthetic-computing

[Fortnow, 2001] L. Fortnow, Kolmogorov Complexity. In Downey, R.; and Hirschfeldt, D., editor(s), Aspects of Complexity, Minicourses in Algorithmics, Complexity, and Computational Algebra, NZMRI Mathematics Summer Meeting, Kaikoura, New Zealand, January 7--15, 2000, volume 4, of de Gruyter Series in Logic and Its Applicatiaons. De Gruyter, 2001.

[Girod, 2006] Mark Girod. A conceptual overview of the role of beauty and aesthetics in science, Studies in Science Education, Volume 43, Issue 1, 2007.

[Gustafson, 2004] John R. Gustafson Wolfgang Pauli 1900 to 1930: His Early Physics in Jungian Perspective, Minneapolis, Minnesota July 2004

[Hume, 1910] David Hume, An Enquiry concerning Human Understanding. 1910 P.F. Collier & Son, http://www.gutenberg.org/files/9662/9662-h/9662-h.htm

[Jaeger, 1945] Werner Jaeger, Paideia: The Ideals of Greek Culture, vols. I-III, trans. Gilbert Highet, Oxford University Press, 1945.

[Kaiser, 2005] David Kaiser, Physics and Feynman's Diagrams, American Scientist, Volume 93, Number 2, March-April 2005.

[Kahneman and Klein, 2009] Daniel Kahneman, Gary Klein, Conditions for intuitive expertise: A failure to disagree.American Psychologist, Vol 64(6), Sep 2009, 515-526.

[Kernigan and Plauger, 1978] Brian W. Kernighan, P.J.Plauger, The Elements of Programming Style, McGraw-Hill; 2nd edition (1978)

[Knuth, 1974] Donald Knuth: Computer Programming as an Art, CACM, December 1974

[Knuth, 1992] Donald Knuth. Literate Programming. Stanford University Center for the Study of Language and Information (Lecture Notes, No. 27), 1992

[Kokol et.al, 1999] P. Kokol, V. Podgorelec, M. Zorman, T. Kokol, T. Njivar, Computer and Natural Language Texts - A Comparison Based on Long-Range Correlations, Journal of the American Society for Information

Science, John Wiley & Sons, vol. 50, num. 14, pp. 1295-1301, December 1999.

[Kolmogorov, 1998] Andrey Kolmogorov, On Tables of Random Numbers. Theoretical Computer Science 207 (2): 387–395, 1998

[Kragh, 1990] Helge Kragh, Dirac: A Scientific Biography. Cambridge: Cambridge University Press, ISBN 0-521-38089-8.

[Luminet, 2011] Jean-Pierre Luminet, Science, Art and Geometrical Imagination, The Role of Astronomy in Society and Culture, Proceedings of the International Astronomical Union, IAU Symposium, Volume 260, p. 248-273

[Madni, 2012] A.M.Madni. Elegant Systems Design: Creative Fusion of Simplicity and Power. Systems Engineering, Volume 15, Issue 3 Autumn (Fall) 2012 pp 347–354.

[MacLennan, 1997] Bruce MacLennan, "Who Cares About Elegance?" The Role of Aesthetics in Programming Language Design (1997), ACM Sigplan Notices february 1997

[MacLennan, 2006] Bruce MacLennan, Aesthetics in software engineering, Encyclopedia of Information Science and Technology, Second Edition 2009. Information Resources Management Association, USA

[Manev, 1998] Introduction to discrete mathematics 2 ed, Arts, 1998 (in Bulgarian)

[McAllister, 1990] J.W. McAllister. Dirac and the aesthetic evaluation of theories, Methodology and Science, 23, 87 - 102 (1990)

[McAllister, 2005] J.W. McAllister. Mathematical Beauty and the Evolution of the Standards of Mathematical Proof, M. Emmer, ed., The Visual Mind II, 15 - 34 (2005)

[Mighton, 2007] John Mighton, The End of IgnoranceL Multiplying our Human Potential, Vintage Canada (August 12, 2008)

[Nahin, 2006] Paul, J. Nahin, Dr. Euler's Fabulous Formula: Cures Many Mathematical Ills (Princeton University Press, 2006)

[Papert, 1980] Seymour Papert. Children, Computers and Powerful Ideas. Basic Books, New York, 1980

[Rigau et al 2007] Jaume Rigau, Miquel Feixas, and Mateu Sbert, Conceptualizing Birkhoff's Aesthetic Measure Using Shannon Entropy and Kolmogorov Complexity, Computational Aesthetics in Graphics, Visualization, and Imaging (2007)

[Root-Bernstein, 1989] R. Root-Bernstein, Discovering. Inventing and solving problems at the frontiers of science. Cambridge, MA: Harvard University Press.

[Root-Bernstein, 1997] R. Root-Bernstein, The sciences and arts share a common creative aesthetic. In A.I. Tauber (Ed.), The elusive synthesis: Aesthetics and science, p. 49-82. Norwell, MA:Kluwer.

[Ross, 2006] Philip E. Ross, The expert mind, Scientific American, August 1, 2006

[Schellenberg. 2004] E. Glenn Schellenberg, Music Lessons Enhance IQ, Psychological Science August 2004 vol. 15 no. 8 511-514

[Schmidhuber, 2010] J¨urgen Schmidhuber, Formal Theory of Creativity, Fun,

and Intrinsic Motivation (1990-2010), IEEE Transactions on Autonomous Mental Development (Volume:2 , Issue: 3), 2010.

[Shalizi 2001] C.R.Shalizi, Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata, PhD thesis, http://bactra.org/thesis/

[Sinclair, 2004] Nathalie Sinclair, The Roles of the Aesthetic in Mathematical Inquiry, Mathematical thinking and learning, 6(3), 261–284, 2004

[Stepanov and Marcus, 2004] Alexander A. Stepanov and Matthew A. Marcus:. Notes on the foundations of programming, www.stepanovpapers.com/PAM.pdf

[Sternberg, 2008] Robert J. Sternberg, Increasing Fluid Intelligence is Possible After All. PNAS, 105, no. 19, 6791- 6792, 2008, http://www.pnas.org/content/105/19/6791.full.pdf

[Suzanne, 2011]. Bernard Suzanne, Frequently Asked Questions about Plato 2011. http://plato-dialogues.org/faq/faq009.htm,

[Taleb, 2010]. Nassim Taleb, The Black Swan, 2010 Random House

[Tomov and Ivanova, 2014] Latchezar P. Tomov, Valentina Ivanova, Teaching good practices in software engineering by counterexamples, CSECS 2014, June 4-7 2014, Sozopol, Bulgaria

[Tomov and Ivanova 2015a]. Latchezar P. Tomov, Valentina Ivanova, Software quality from systems perspective, CSECS 2015, June 4-7 2015, Boston, MA USA

[Tomov and Ivanova, 2015b]. Latchezar P. Tomov, Valentina Ivanova, Software understandability model, CSECS 2015, June 4-7 2015, Boston, MA USA

[Tversky and Kahneman, 1974] Amos Tversky; Daniel Kahneman, Judgment under Uncertainty: Heuristics and Biases, Science, New Series, Vol. 185, No. 4157. (Sep. 27, 1974), pp. 1124-1131.

[Villani, 2015] Cedric Vellinai, Birth of a Theorem: A Mathematical Adventure, Farrar, Straus and Giroux (April 14, 2015)

[Willingham, 2004] Daniel T. Willingham, Practice Makes Perfect—but Only If You Practice Beyond the Point of Perfection, American Educator, Spring, 2004 http://www.aft.org/periodical/american-educator/spring-2004/ask-cognitive-scientist#sthash.mMAc5a4B.dpuf

[Willingham, 2009] Daniel Willingham, Is it true that some people just can't do math?, American Education, winter 2009-2010, http://www.aft.org/sites/default/files/periodicals/willingham.pdf

[Yushkevich et.al., 1970] Adolf P. Yushkevich (ed), Boris Rosenfeld. A. I. Volodarsky, E.I. Berezkina, I.G.Bashmakova, History of mathematics from ancient times until 19th century, vols. I-III, “Science”, Moscow, Russia, 1970

[Zeki et. al, 2014] Semir Zeki1, John Paul Romaya, Dionigi M. T. Benincasa and Michael F. Atiyah The experience of mathematical beauty and its neural correlates, Front. Hum. Neurosci., 13 February 2014, http://journal.frontiersin.org/article/10.3389/fnhum.2014.00068/full

Authors' Information

Your photo here:
Height: 2,58 cm
Width: 1,84 cm

Latchezar Tomov, PhD, assistant professor in Deprartment of Informatics, New Bulgarian University, Sofia, luchesart@gmail.com..

Major Fields of Scientific Research: *control theory, software engineering, software project management*