Application of Evolutionary Algorithms in Interaction Design

from Requirements and Ontology to Optimized Web Interface

Maxim Bakaev
Economic Informatics department
Novosibirsk State Technical University
Novosibirsk, Russia
bakaev@corp.nstu.ru

Martin Gaedke
Distributed and Self-organizing Systems department
Technische Universität Chemnitz
Chemnitz, Germany
martin.gaedke@informatik.tu-chemnitz.de

***Abstract*— The paper is dedicated to automated and semi-automated engineering of human-machine interfaces through artificial evolution: the state-of-art, the notable problems, and the preposition of a potentially feasible evolutionary algorithm. The A/B Testing method is already widely used in web interaction design, but it utilizes a single measurable goal and therefore risks random undirected modifications in the interface "phenotype", since its "genotype" is not distinguished. Also, purely interactive assessment of the potentially defective solutions' fitness by real users is undesirable, so the neural networks-based Kansei Engineering (KE) method is increasingly used for fitness evaluation. We reason that KE can be extended to consider any kinds of requirements, not just the emotional aspect, and outline the relevant techniques. Further, as an interface abstract model corresponds to a genotype, we propose a dedicated domain ontology to represent the genome, i.e. the structure of the design space. Finally, we summarize the combination of several existing AI and design methods in an evolutionary algorithm that may be used for automated development and improvement of web user interfaces in the coming era of the Big Interaction.**

Keywords—evolutionary computation; genetic algorithms; Kansei Engineering; model-based interface design; ontologies

I. INTRODUCTION

Charles Darwin's *The Origin of Species* is considered one of the most influential books in terms of its effect on biology, science, and even worldview of many people. Principal components of the modern Theory of Evolution, such as within-species variability, natural selection, heritability of fitness, etc., were initially laid there. Since then, evolutionary algorithms (EAs) traversed from a novel idea of the 1960s to a widely recognized method in Artificial Intelligence (AI), currently having many practical applications. Domains of evolutionary computation traditionally include routing and other optimization tasks [1], programming [2], design and engineering [3], data mining and classification [4], economics and investment [5], as well as more recent applications in natural language processing, robotics, machine learning, intelligent and recommender systems, and more [6], [7].

In our paper we consider the usage of EAs in interaction design – that is, production and improvement of user interfaces for software systems and websites. We believe this application is currently of significant importance, since user interfaces' share in programming code and development time is close to 50% [8]. Moreover, today's diversity of user traits and needs, as well as of interface devices and contexts of use, leads us to what we may call *Big Interaction*, when hand-made design of each and every web interface turns out to be impossible or economically unfeasible, so employment of AI methods becomes inevitable, to aid human creators or even replace them [9].

However, the fundamental possibility of EAs application for interaction design remains unproven, because general feasibility of AI methods in design and engineering seems to depend of how conventional the particular field is. So, for routine problems, which may generally be encountered in technical systems not involving humans, like bridge-building or electrical circuits design, EAs can be highly efficient [10]. On the other side of the spectrum are the so-called "wicked problems" (see, e.g. [11]), solutions for which are notably impossible to automate, and some argue that creative design belongs here too. We believe that, as more and more standards on the web are established and projects completed, web interaction design gradually drifts towards the routine side, but can one hope it already got into the "automatable zone"? Besides, EAs are commonly applied to black-box problems, and the solutions often don't make sense to a human designer, even though they may be minimalist and quite efficient [12]. But in human-computer interaction, interface does have to "tell a story" to human user – it is not said by mere chance that design is communication – so how effective evolutionary solutions could be in this?

So, in our research work we make an attempt to reason about the above matters and outline approaches and existing methods that could contribute to constructing operational algorithms. In Section 2, we analyze how EAs and design processes could work together and we frame the relevant problems more specifically. Then in Section 3, we put together Kansei Engineering, which is extended to perform fitness evaluation, with a Model-Based approach to interface design, supplemented with an ontology corresponding to genome, and explain the apparent benefits. Finally, in Conclusions we summarize our findings with a diagram, as well as outline limitations and prospects for further research.

This research work was performed under DAAD's (Deutscher Akademischer Austauschdienst) grant, Mikhail Lomonosov program.

978-1-5090-0445-4/16/$31.00 ©2016 IEEE

II. Evolutionary Algorithms

A. Core Parts of Evolutionary Algorithms

Evolutionary algorithms, inspired by biological evolution, are now a subset of AI methods, performing metaheuristic optimization via repeated application of evolutionary operators. A basic EA may be outlined as the following sequence:

Initialization – creation of initial population (first generation), which may be done randomly or with certain considerations, for better algorithm convergence and final solution quality (see review of relevant techniques in [13]).

Selection of best-fit individuals for reproduction (parents) based on **evaluation** of their fitness, which is performed either in real world context or, more commonly, with specially formulated **fitness functions**.

Reproduction of parents to create individuals of the next generation, in which genetic operators such as **mutation** (random or directed variations) and **crossover** (producing child solutions from parents) are used. If termination condition, based on sufficient quality of the final solution or time limit, is not reached, go back to previous step and repeat.

It should be noted, although not all EAs clearly reflect this, that there's an important distinction between the actual individual organism's properties (phenotype) and its hereditary information (**genotype**) [14]. From biological evolution, it is known that genetically identical organisms normally differ in phenotypes, in particular due to development under different environmental conditions. Thus, if evolutionary computation is performed with chromosomes and genes, i.e. as *Genetic Algorithm,* an additional procedure to **generate** individual candidate solutions from hereditary genetic model, similarly to biological epigenesis, may have to be introduced. Another popular subset of EAs, *Evolution Strategy* methods, don't distinguish genotype and rather use recombination of fitter individuals' explicit properties as innovation operator to create new generations.

Among the parts listed above, the fitness function is arguably the most crucial one for practical success of an EA [15] – it must wholly reflect the goal, at the same time being as easily computable as possible, since the ultimate algorithm speed depends mostly upon this. Naturally, for complex problems achievement of both these conditions at once is challenging, so various workarounds are proposed, ranging from fitness approximation to the so-called *Interactive Evolutionary Computation* (IEC) that involves human experts or users. Fitness approximation must be used sensibly, otherwise chances of EA not yielding globally optimal solutions increase significantly, so generally at least for some individuals exact fitness function is still calculated (see comprehensive review in [16]). Interactive/human-based evolutionary computation implies delegating some or all genetic operators to humans, and while it may exhibit fair performance [17], IEC is more commonly used for domains in which the fitness function is not known, like art, preferences analysis, etc. In any case, it's worth remembering that a fitness function is not an end in itself, but means for selection, which in biological evolution is performed by the environment that tests fitness of an individual's phenotype. We will elaborate further on selection with fitness function or otherwise, but first let us consider the prospects of EAs application in design and interaction design in particular.

B. The Design Process

Actually, the classic, *Rational Model* of design process is essentially an evolutionary algorithm. Its general outline, as described, e.g. in [18], can be summarized as the following:

```
Defining the goal
Defining desiderata (secondary objectives)
Forming the utility (i.e. fitness) function
Finding out constraints
Designing tree of decisions
   UNTIL ("good enough") or (time constraint met)
   DO another design (to improve utility function)
        UNTIL design is complete
           WHILE design remains feasible
           make another design decision
           END WHILE
           Backtrack up design tree
           Explore a path not searched before
        END UNTIL
      END DO
   Take best design
   END UNTIL
```

Thus the design process is viewed as a systematic search in the design space that is tree-structured (although it doesn't have to be the only possible structure), seeking to optimize the utility function while satisfying the constraints. It should be noted that one of the motivations for defining this model was to apply AI methods for designing [**Ошибка! Ошибка связи.**:16]. Among obvious disadvantages of the model are presuppositions 1) that the goal, the desiderata, and the constraints (which together largely constitute what we'd call *Requirements* in software engineering), are entirely known at the beginning of the design process, and that all relations between them can be identified and quantified, so that a computable utility function can be formulated; 2) that the combinatorial design space is overseeable and has relatively uncomplicated structure, such as the tree with independent branches. Although being an ideal rather than a real-life thing, the Rational Model was quite widely accepted and adapted in design standards, such as e.g. Verein Deutscher Ingenieure (VDI-2221) [19].

C. A/B Testing in Interaction Design

A popular implementation in practice of the above process's *DO another design* part in interaction design, particularly web development, is *A/B Testing* – a technique that takes different versions of a design and measures their performances in the real environment. Usually differences between the designs are minor, the "fittest" design survives, becoming a "parent" for subsequent improved versions, and considerable effect in performance is achieved after a lot of "generations" [20] – so A/B Testing may well be considered

an evolutionary approach. The two major problems it has are similar to the disadvantages of the Rational Model.

First, the issue of a measurable goal function remains, and it's recommended to use the method for websites that have a single measurable KPI, though it surely would be beneficial to consider requirements more wholly [21]. A/B Testing has to be performed in a real environment, which is both its greatest strength and a substantial weakness, as running potentially poor versions of the design on an operational website would increase the risk of user abandonment, harm to the owner's reputation or even direct economic losses. Second, without any further support, the method generally doesn't provide qualitative insights to the designer responsible for creating new versions. With the lack of a systematic approach to recombination and the low number of interactions, the method may quite easily turn into a random walk. Thus, to avoid subjecting real users to "A/B test lottery", it is recommended to combine A/B Testing with user research [22] and use it only as a supplement to qualitative usability engineering methods that actually deliver better results.

D. Summary of the Problems

So, in terms of an evolutionary approach currently prevalent interaction design methods seem to have notable ambiguities with selection of better candidates and with the reproduction operators. Evaluation with a fitness function is troublesome as defining computable expression of the design utility is problematic, especially since the function is very much likely to have its own evolution, due to changing requirements. At the same time, straightforward evaluation of manifold and potentially low-quality design solutions in real environments with real users, in accordance to A/B Testing, is also effectively impossible.

The problems with reproduction via crossover and mutation arise first of all due to practically inadequate representations of the entangled design space, or lack of them whatsoever. In A/B Testing, de-facto phenotype features of designs are being recombined, somewhat blindly, and genotype is not distinguished. That does not only make it difficult to keep track of changes and make sense out of them, but it means that if the creation of new designs is to be automated, the reproduction operators would have to be platform-dependent. So, modification of technologies employed in interaction, or for example, changes in HTML and CSS conventions would make these significant parts of an EA obsolete.

Thus, in the current state of interaction design, human participation would seem to be indispensable in most parts of EAs. The situation deteriorates by the fact that the human designers working process is still largely unknown, unknown but it's said that it definitely doesn't follow the Rational Model [18]. Still, we believe it may describe well how computers should design [23], so in the subsequent section of the paper we propose a Kansei Engineering-based method for efficient selection and a Model-Based Ontological approach for representing genotype, which should allow initialization and generation of interfaces (phenotypes), as well as reproduction with both crossover and mutation.

III. Evolutionary Algorithm for Interaction Design

A. Kansei Engineering for Fitness Evaluation

Currently, *Kansei Engineering* (KE) is a set of methods and techniques – some authors identify already as many as 6 types of KE [19] – relating customers' feelings and impressions with existing or prospective products or their certain features. KE application started in Japanese automotive industry in the 1980s, and its *analytical* method includes the following principal steps [24]:

1. Creating the list of concepts describing the emotional sphere of potential customers or users of the product. If no accepted catalogue exists for the domain, preliminary investigation may be performed to narrow down the initial hundreds of terms extracted from literature, experts, and users to dozens of most characteristic ones, which then constitute emotional scales (*Kansei Words*).

2. Developing the general set of a product's attributes and design-related decisions that can be made regarding them. Effectively, this is design space that we mentioned above – a tree-like or network-like structure, where each possible design resolution is represented as a pair: category (e.g. color or size) and value (e.g. 20 px).

3. Selecting existing products or their prototypes that will be assessed, and then running the experimental research – generally a survey, when user representatives evaluate the artifacts per emotional scales, e.g. from 1 to 5, or from -3 to +3.

4. Using formal methods to analyze the obtained data and establish the relations between the emotional scales and the product's attributes. The most widely used types of KE and the corresponding approaches that are used for analyzing the data may be outlined as the following [19]:

- Type I (KE simple category classification): uses statistical approach, most suitable for small data sets with simple relationships among variables.
- Type II (KE Computer System): uses soft computing to find patterns in data, usually applied for large data sets with rather complex and dynamic relationships between variables.
- Type III (KE Modeling): uses mathematical model approach, suitable for large data sets with highly complex system relationships.

The "inverse" task, or may we say *synthetic* method of KE, is obtaining the list of the prospective product's attributes and design resolutions from the desired Kansei (emotional feeling) of the target customer. This can be done if the knowledge about their relationships was already extracted and formally specified, thus synthetic KE is essentially an AI method. Certainly Kansei knowledge bases already existed by the time when the Mazda Miata car was designed with the KE approach. Open publications related to practical KE expert/intelligent systems became rife in the mid-90s, see e.g. [25]. In this KE Type II, probably the most widely used AI method that kept its popularity to the present day and proliferated to the web design domain, was neural networks ([26], [27], [28]). The attractiveness of this method is due to its self-learning capability that is a natural requirement for KE (supervised learning would not be possible, since desirable

correct teacher patterns are not known in advance) [26], [25], and due to reasonable computation effectiveness compared to other AI or statistical methods [29].

At the same time, neural networks have become recognized means to evaluate fitness in EAs, when neither precise values of fitness function can be determined due to obscurity or computational complexity, nor IEC is feasible [30]. Relevant hybrid methods imply accurate calculation of fitness function only for some of the candidate individuals, the ones that passed preliminary step of selection carried out with approximate values estimated by neural network [15]. Moreover, the natural adaptability of neural networks [31] allows them to accommodate fitness functions that may fluctuate over generations (mutable).

All of the above considerations suggest that KE approach in combination with neural networks could be effectively applied for selection in web interaction design. So, if candidate designs are first evaluated automatically, and only reasonably fit ones are subsequently used in A/B Testing in real a environment, the risk of causing aversion in website users would decrease significantly, while the evolution pace could speed up, as time required for IEC diminishes [32]. Furthermore, evaluation by adaptive neural networks could handle mutability in fitness functions that arises due to changing requirements, whose inevitability in software design is a substantial objection against the Rational Model, as well as due to adjustments in target users' experience and preferences [33].

However, it seems obvious that traditional KE that considers only emotional aspect of user's interaction with a product would be considerably limited in evaluation of a web interface, a design that must both follow non-functional (qualitative) requirements and embrace functionality of web applications. So in the next chapter we put forward our ideas regarding an extension of the Kansei (emotional) Engineering approach to the two other interaction aspects, which we identify as physical (considers use of interfaces as real-world objects, their ergonomic characteristics) and cognitive (relates to information processing, thesaurus, memory and learning, etc.) [34].

B. *Extending Kansei Engineering for Requirements*

There seems to be no fundamental obstacles why KE couldn't be extended from its conventional emotional aspect to the two other, physical and cognitive (of course, all the aspects are rather interrelated in real interaction, but we use them to illustrate transition to requirements). The most straightforward way of course would be to append additional Kansei words, e.g. "prompt" (system response time – physical aspect), "feeling of being in control" (cognitive aspect), "exhausting" (physical and cognitive load), "work-effective" (completeness of functional requirements), etc. However, we'd like to note that KE in fact uses the subjective evaluation method to measure customers' emotions not due to its accuracy or robustness, but because it currently remains the only reasonable approach to detect complex feelings in humans. It is not by accident that neural networks, famously effective for solving problems represented as black-boxes, are the most popular AI method in KE.

There are, however, more unprejudiced approaches to measure interaction quality in physical and cognitive aspects, and we believe that supplementing KE with them would lead to more holistic and faster evaluation. Relevant techniques for automated evaluation of user interfaces quality are reasonably well developed for the physical aspect – HTML and CSS validation, website response time monitoring, GOMS models, optimization of sizes and distances between interface elements (see e.g. [35] for Fitts' law-based optimization). Programmed consideration of cognitive aspect in interaction so far remains rather rudimentary, but there are endeavors to develop quantitative measures (e.g. to assess interface "information capacity" [36]), to automate qualitative methods, such as usability testing, and to determine integral usability of web interfaces – e.g. WaPPU [21] does this based on user interactions. However, since we'd like to be able to predict usability qualities without recourse to either user testing or real interaction, it seems to leave us with the only remaining feasible option – modeling.

Indeed, having a model of the interface could both aid in evaluating its usability [37] and in dealing with the problem of a complex and entangled design space that was noted already in the Rational Model. The latter problem was also persistent in A/B Testing that was unable to supply designers with sensible insights on feasible changes in future interface versions without reliance on added qualitative methods. As for KE, its step #2 responsible for construction of the design space is also far from being negligible, unless a well-established industry like car manufacturing already possesses the technological knowledgebase. In EA terms, interface model would correspond to its genotype, enabling crossover and mutation operators that are well-established in genetic algorithms. So, in the following chapter we'd like to elaborate on a model-based approach in interface design and relate it to EAs.

C. *Model-Based Interface Design and the Design Space*

The so-called model-based approach to interface design generally implies that the code of the actual interface is generated based on knowledgebase rules or on optimization of certain interface parameters or expected quality indexes [35] – from its model. The model may be either specified by designer or somehow derived – e.g. from existing programming code, database models, domain-specific languages, high-level user tasks, or requirements. The core parts of the model-based approach can be divided into: 1) requirements-related models: Tasks and Domain, 2) per se interface models: Abstract UI, Concrete UI, and Final UI, and 3) context of use models: User, Platform, and Environment [8]. In some cases the parts are composed differently, e.g. as Domain model, UI Representation model, Application (functionality) Linkage model, and Dialog Scenario model in [37]. In a restricted sense, *interface model* relates to Abstract UI, which is modality- and implementation-independent (also, a review of UI description languages may be found in [38]).

The model-based approach seeks to decrease laboriousness for designers, programmers, and maintainers when the user interface is developed or ported to different platforms. However, there is a fundamental drawback in the approach: effort required for detailed formal specification of a complex interaction may well exceed the one needed for making an actual user interface. So, it is no coincidence that currently existing tools implementing this approach are mostly aimed to

create user interfaces for relatively simple tasks or for special contexts of use (see their review in [9]).

As we mentioned before, in EA-based interaction design, interface model would correspond to genotype, and indeed there are recent research works such as [28] or [27] that put forward the use of chromosomes to represent candidate interfaces. However, the traditional linear structure of chromosomes seems to be quite disadvantageous, since it doesn't contribute to representing the design space and interdependencies between its elements – the "genes". Accordingly, the procedures for reproduction, generation, etc. have to contain knowledge about the design space, otherwise the EA may end up trying combinatorial matches and producing too many candidate solutions, which is unwelcomed in IEC-based algorithms. Thus it is sensible to separate out this knowledge as a user interface meta-model, and we believe its practical implementation could be a domain ontology, effectively representing the design space – i.e. in EA terms corresponding to **genome**.

Ontologies currently have varying definitions, but they generally contain concepts, hierarchical and associative relationships between them, and attributes for concepts. Ontologies may contain instances created from concepts via assigning concrete values to attributes – that allows specifying interface models, similarly to genome-genotype dualism in EA. As a popular AI method, ontologies have already made their way both into model-based interface design and KE [39]. In our own previous research we successfully used a dedicated web design ontology as the core part of a web design support intelligent system [34], and later justified its application in KE to represent design resolutions [40]. The ontology allows incorporating Domain model (websites), Abstract UI, User model, Platform model (as HTML+CSS), and Environment model. The Tasks model is automatically derived from project requirements, and Concrete UI and Final UI are auto-generated. As an example from our ontology, we show concepts/genes for *Interface element* and *Design property* classes/chromosomes in Fig. 1.

The employment of ontologies in EA may also mitigate the most severe drawback of the model-based approach noted above – the advanced increase of the interaction model complexity – so initialization may result in a simple or incomplete interface model, which would gradually obtain deficient parts. The interface code generation procedure would meanwhile create several varying solutions based on the same model, just as different phenotypes develop from one genotype. The understanding that something is missing from the model and must be completed – randomly, through auto-analysis of requirements, or by certain rules – can be obtained via comparison against the meta-model (e.g. compulsory attributes in the ontology that are nevertheless not specified in the model). Then there must be a mechanism for "phenotype fixing" (even though nonexistent in biological evolution) to complement the model with the "genes" of best fit interfaces, which should also speed up the interaction design EA convergence to near-optimal result.

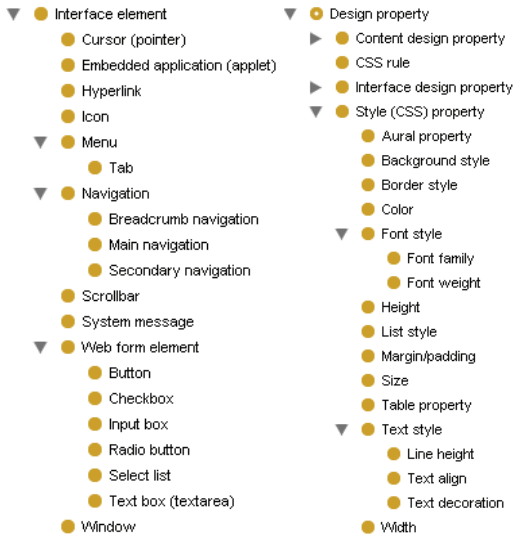

Fig. 1. Structure of *Interface element* and *Design property* classes of web design domain ontology [40]

IV. Conclusions

In our paper, we discussed the applicability of evolutionary algorithms for designing human-computer interfaces, reviewed the state-of-art in the field, and put forward combination of several existing AI and design methods to resolve the identified problems. We present the resulting process in Fig. 2.

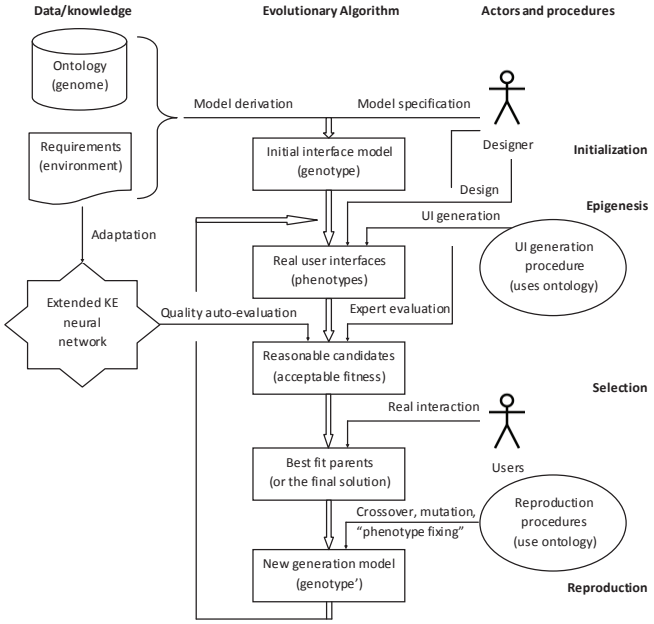

Fig. 2. Proposed approach for EA in interaction design

We found that Kansei Engineering in combination with neural networks is already increasingly used for evolutionary web design [27], [28], but considering almost solely emotional aspects, so the matter of incorporating the other kinds of requirements remains a research challenge. We believe that this could be done via addition of logical inference rules and quantitatively determined interaction quality indexes, which would correspond to gradual "box-whitening" in the neural network.

Regarding the limitations of our study, we'd like to note that it's mostly applicable to web interaction, since Web Design KE currently seems to be better developed compared to general Interaction Design. Websites are more abundant than software applications and are easier accessible via online channels, which is an important feature, since neural networks need lots of diverse data for self-learning. Also, web interface properties can be more effectively analyzed inside the KE framework, given higher typicality in terms of user tasks, as well as technologies, standards and platforms employed. In addition, generation of web interfaces is less of a challenge, since the web browser is partially responsible for their presentation, which would likely allow more undemanding and computationally effective code generation procedure. So, future research prospects include validation of KE applicability for all kinds of requirements, which should entail a corresponding experimental set-up with existing Kansei words for the web design domain. Another direction for research is analysis of existing ontologies (e.g. [39]) that could be used in model-based interface development and representation of genome for EAs.

ACKNOWLEDGMENT

This research work was performed under DAAD's grant, Mikhail Lomonosov program, 2015-2016.

REFERENCES

[1] C.M. Fonseca and P.J. Fleming. "An overview of evolutionary algorithms in multiobjective optimization." Evolutionary computation 3, no. 1, 1995, pp. 1-16.

[2] Z. Michalewicz. Genetic algorithms + data structures = evolution programs. Springer Science & Business Media, 1999.

[3] P. Bentley. Evolutionary design by computers. Morgan Kaufmann, 1999.

[4] A.A. Freitas. Data mining and knowledge discovery with evolutionary algorithms. Springer Science & Business Media, 2002.

[5] M.G.C. Tapia and C.A.C. Coello. "Applications of multi-objective evolutionary algorithms in economics and finance: A survey." In IEEE congress on evolutionary computation, vol. 7, 2007, pp. 532-539.

[6] C.A.C. Coello. "Multi-objective evolutionary algorithms in real-world applications: Some recent results and current challenges." In Advances in Evolutionary and Deterministic Methods for Design, Optimization and Control in Engineering and Sciences, 2015, pp. 3-18.

[7] Y.J. Gong et al. "Distributed evolutionary algorithms and their models: A survey of the state-of-the-art." Applied Soft Computing 34, 2015, pp. 286-300.

[8] G. Meixner, G. Calvary, and J. Coutaz. Introduction to Model-Based User Interfaces. W3C Working Group Note 07 January 2014, 2014.

[9] M. Bakaev and T.Avdeenko. "Defining and optimizing user interfaces information complexity for AI methods application in HCI." In HCI: Interaction Technologies, LNCS 9170, 2015, pp. 397-405.

[10] G. Renner and A. Ekárt. "Genetic algorithms in computer aided design." Computer-Aided Design 35, no. 8, 2003, pp. 709-726.

[11] J. Conklin. Dialogue mapping: Building shared understanding of wicked problems. John Wiley & Sons, Inc., 2005.

[12] T. Pratchett, I. Stewart, and J.S. Cohen. The science of Discworld. EBURY PRESS/Random House, 1999.

[13] B. Kazimipour, X. Li, and A.K. Qin. "A review of population initialization techniques for evolutionary algorithms." In IEEE Congress on Evolutionary Computation (CEC), 2014, pp. 2585-2592.

[14] D.B. Fogel. "Phenotypes, genotypes, and operators in evolutionary computation" In IEEE Int. Conf. on Evolutionary Computation, vol. 1, 1995, pp. 193-198.

[15] Y. Jin and B. Sendhoff. "Reducing fitness evaluations using clustering techniques and neural network ensembles." In Genetic and Evolutionary Computation–GECCO, 2004, pp. 688-699.

[16] Y. Jin. "A comprehensive survey of fitness approximation in evolutionary computation." Soft computing 9, no. 1, 2005, pp. 3-12.

[17] C.D. Cheng and A. Kosorukoff. "Interactive one-max problem allows to compare the performance of interactive and human-based genetic algorithms." In Genetic and Evolutionary Computation–GECCO, 2004, pp. 983-993.

[18] F. Brooks. The design of design: Essays from a computer scientist. Pearson Education, 2010.

[19] V. Čok, M.D. Fikfak, and J. Duhovnik. "Integrating the Kansei Engineering into the Design Golden Loop Development Process." In ICoRD'13, 2013, pp. 1253-1263.

[20] J. Nielsen. "A/B Testing, Usability Engineering, Radical Innovation: What Pays Best?" NNGroup, March 26, 2012.

[21] M. Speicher, A. Both, and M. Gaedke. "Ensuring web interface quality through usability-based split testing." Web Engineering, LNCS vol. 8541, 2014, pp. 93-110.

[22] J. Cardello. "Define Stronger A/B Test Variations Through UX Research." NNGroup, April 20, 2014.

[23] E. Stolterman. "The nature of design practice and implications for interaction design research." Int. J. of Design 2, no. 1 2008, pp. 55-65.

[24] S.T.W. Schütte, J. Eklund, J.R.C. Axelsson, and M. Nagamachi. "Concepts, methods and tools in Kansei Engineering." Theoretical Issues in Ergonomics Science 5, no. 3, 2004, pp. 214-231.

[25] S. Ishihara, K. Ishihara, M. Nagamachi, and Y. Matsubara. "An automatic builder for a Kansei Engineering expert system using self-organizing neural networks." Int. J. of Industrial Ergonomics 15, no. 1, 1995, pp. 13-24.

[26] Y.C. Lin, C.H. Yeh, and C.C. Wei. "How will the use of graphics affect visual aesthetics? A user-centered approach for web page design." Int. J. of Human-Computer Studies 71, no. 3 (2013): 217-227.

[27] Q.X. Qu. "Kansei knowledge extraction based on evolutionary genetic algorithm: an application to e-commerce web appearance design." Theoretical Issues in Ergonomics Science 16, no. 3, 2015, pp. 299-313.

[28] F. Guo, W.L.Liu, Y. Cao, F.T. Liu, and M.L. Li. "Optimization design of a webpage based on Kansei Engineering." Human Factors and Ergonomics in Manufacturing & Service Industries, 26, no. 1, 2016, pp.110-126.

[29] S. Ishihara, K. Ishihara, M. Nagamachi, and Y. Matsubara. "An analysis of Kansei structure on shoes using self-organizing neural networks." Int. J. of Industrial Ergonomics 19, no. 2, 1997, pp. 93-104.

[30] J. Dias, H. Rocha, B. Ferreira, and M.C. Lopes. "A genetic algorithm with neural network fitness function evaluation for IMRT beam angle optimization." Central European J. of Operations Research 22, no. 3, 2014, pp. 431-455.

[31] X. Yao. "Evolving artificial neural networks." Proceedings of the IEEE 87, no. 9, 1999, pp. 1423-1447.

[32] H. Takagi. "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation." Proceedings of the IEEE 89, no. 9, 2001, pp. 1275-1296.

[33] A. Milani. "Online genetic algorithms." Int.Conf.ICT&P, Sofia, 2004.

[34] M. Bakaev. "Razrabotka intellektualnoj sistemi dlya podderzhki proektirovaniya cheloveko-kompyuternogo vzaimodeistviya." PhD dissertation, NSTU, Novosibirsk, Russia, 2012.

[35] K.Z. Gajos, D.S. Weld, and J.O. Wobbrock. "Automatically generating personalized user interfaces with Supple." Artificial Intelligence 174, no. 12, 2010, pp. 910-950.

[36] M. Bakaev and T. Avdeenko. "A Quantitative Measure for Information Transfer in Human-Machine Control Systems." IEEE Int. Siberian conference on control and communications (SIBCON), 2015.

[37] V. Gribova. "A method of estimating usability of a user interface based on its model." Information Theories & Applications 14, no. 1, 2007, pp.43-47.

[38] J. Guerrero-Garcia, J.M. Gonzalez-Calleros, J. Vanderdonckt, and J. Muoz-Arteaga. "A theoretical survey of user interface description languages: Preliminary results." In Web Congress, 2009. LA-WEB'09. Latin American, 2009, pp. 36-43.

[39] Q. Qiu, H. Cai, L. Jiang, and K. Omura. "Ontology-based feature modeling and combination for Kansei Engineering." 7th IEEE Int. Conf. on Ubi-Media Computing and Workshops (UMEDIA), 2014, pp. 26-32.

[40] M. Bakaev, T. Avdeenko. "Ontology-Based Approach for Engineering Web Interface Design Resolutions." 5th Int. workshop on Computer Science and Engineering (WCSE 2015-IPCE), Moscow, 2015, pp. 196–201.