Influence Power Factor for User Interface Recommendation System

Marek Krótkiewicz[1](✉), Krystian Wojtkiewicz[1], and Denis Martins[2]

[1] Faculty of Computer Science and Management, Wroclaw University of Science and Technology, Wroclaw, Poland
marek.krotkiewicz@pwr.edu.pl

[2] ERCIS, University of Muenster, Leonardo-Campus 3, 48149 Muenster, Germany

Abstract. User interface is an important element of software system since it provides the means for utilizing applications' functionalities. There is number of publications that propose guidance for proper interface design, including adaptive approach. Following paper introduces general idea for definition of interface design in a way that allows for easy computing of user interface effectiveness. The introduced factor can be used for recommendation of interface changes and adjustment.

Keywords: User interface design · Interface recommendation Interface components

1 Introduction

The quality and effectiveness of user interfaces relies mostly on the technical knowledge, experience, and talent of the designer. However, with the increasing need for personalization in the Web, particularly after the advent of the Web 2.0, a growing population of non-technical, unexperienced users is required to perform user interface design tasks (e.g., Blogs and personal Website personalization). Unfortunately, these users are commonly not aware of design principles that may assist them during the conceptualization and implementation of user interfaces. Moreover, due to the intrinsic creative nature, this process is frequently characterized as semi-structured (i.e., there is no methodology that can successfully serve as a standard to every design case) and highly interactive, which potentially hinders productivity.

In this work, the authors propose the idea of building and evaluating a recommender system that supports unexperienced designers in an efficient way, allowing to design high-quality interfaces for user-centric applications. Such system considers the appropriate content to be delivered to the user as well as the user interface structure in which content is presented. The process of recommending appropriate design insight is based on ranking and reordering items in a meaningful way in order to both differentiate items by means of an importance degree, and increase the time end-users spend on the platform.

Recommender Systems have been widely used to improve user experience and content personalization in many domains [2,4]. In general, user-centric recommendations aim to promote the discovery of unexpected items that are likely to be relevant to users. For providing such personalization, collaborative filtering has become one of the most representative methods in Recommender Systems. This approach consists of gathering information about several users [13], such as interests, habits, goals, behaviors, preferences and usage statistics in order to select a subset of relevant items to be present to the user [3,15]. The assumption behind collaborative filtering is that users with similar interests and preferences potentially like similar items [9].

Providing such personalization to user interface designers empower them with tools that enhance their capabilities of adapting system's functionality, appearance, or information representation to better suit user needs [11,12,14].

However, at the beginning the designer has little knowledge about users and their preferences as such. Therefore, there is a need to build a system that allows to change the design in time and evaluate it in terms of systems functional assumptions. To build such a system one has to define a proper and arbitrary method to determine whether the system works properly or does it need to be tuned in some way [1]. In order to do that the method presented in this paper was designed. The aim of it is to provide means of design evaluation based on user experience. The method assumes that there are multiple interface (system) states and there are many connections among those states. The system is evaluated on the basis of users states changes. Such an approach is already known [10], however we extend it showing that not only a change from one state to another matters, but rather which component has been used.

In the following section the problem of building the factor is presented in the formal way as well as implementation of structures describing interface using AssoBase – Association-Oriented Metamodel (AOM) is presented. Section 3 presents the algorithm for computing influence power of individual elements that will be used for evaluation of interface design. The summary concludes the paper.

2 Problem Formalization

User interface definition is a demanding task and it relies on many factors e.g. technology, purpose of the system, target user group etc. The intent of authors was to build an abstraction that would allow to introduce an universal factor for assessment of interface definition, i.e. its usability. In order to do that we propose an unambiguous introduction of elements that are further used to describe the proposed factor.

2.1 Structures

Following formalisms are used to identify heterogeneous structures:

$\{e\} = \{e_1, \ldots, e_n\}$ – set of e elements (not changeable, sequence is irrelevant),
$(e) = (e_1, \ldots, e_n)$ – tuple of e elements (not changeable, sequence is relevant),
$\langle e \rangle = \langle e_1, \ldots, e_n \rangle$ – list of e elements (changeable, sequence is relevant).

2.2 Formal Interface Definition

For the purpose of this paper the user interface I is defined as a set of interface states:

$$I = \{s\} = \{s_1, \dots, s_n\} \tag{1}$$

where each state s is defined as tuple built from set of components C and a template t.

$$s = (C, t) \tag{2}$$

where

$$C = \{c\} = \langle c_1, \dots, c_n \rangle \tag{3}$$

and

$$t = \langle r \rangle. \tag{4}$$

The template t, in scope of user interface, is understood as set of regions, where each region r is defined by its position on the screen x and y, width w, height h and influence power i.

$$r = (x, y, w, h, i). \tag{5}$$

The influence power i is assigned to each region based on its location in the list according to the function defined by the designer. However, the concept of the influence power will be evaluated in another section of the paper, thus it will not be explained here.

The component c is understood as tuple built from reference to region r, applied design characteristics d and a link l:

$$c = (r, d, l) \tag{6}$$

Where the design characteristics d of interface component is defined as following tuple:

$$d = (P, a_a, a_m) \tag{7}$$

where P is a list of properties, while a_a is additive influence power adjustment factor and a_m multiplicative influence power adjustment factor. The set of properties is defined as:

$$P = \{p\} = \langle p_1, \dots, p_n \rangle \tag{8}$$

where each property is a tuple of feature list F and value list V:

$$p = (F, V) \tag{9}$$

where:

$$F = \langle f_1, \dots, f_n \rangle \tag{10}$$

$$V = \langle v \rangle = \langle v_1, \dots, v_n \rangle. \tag{11}$$

The link l points to either interface state $l \in I$ or another component $l \in C$, thus $l \in C \cup I$.

2.3 Interface Definition in AssoBase Environment

The proposed method for assessment of interface definition usability aims at possibility not only to define the interface in a formal way but also to provide computational platform for measuring the actual factors. In order to do that authors have used AssoBase, formerly known as AODB [6] as a database layer for a system. The implementation of the structure in AssoBase proves its cohesion. Following the definition of the system in AssoBase formalism [5,7] is provided. Figure 1 presents the definition of the system by the use of AML [8,16] which is a modeling language for AssoBase.

$$FEATURE\left\{\left\langle\begin{array}{ll}+name & : unicode(32),\\ +type & : ascii(2)\end{array}\right\rangle;\right\} \tag{12}$$

$$VALUE\left\{\left\langle +value : byte\right\rangle;\right\} \tag{13}$$

$$CHARACTERISTICS\left\{\left\langle\begin{array}{ll}+add\_power\_adj\_factor & : double,\\ +mult\_power\_adj\_factor & : double\end{array}\right\rangle;\right\} \tag{14}$$

$$REGION\left\{\begin{array}{ll}+x & : double,\\ +y & : double,\\ +width & : double,\\ +height & : double,\\ +influence & : double\end{array}\right\rangle;\right\} \tag{15}$$

$$Property\left\{\left\langle\begin{array}{lcl}[*] & \overset{+Feature}{\bullet\!\!\rule[0.5ex]{4em}{0.4pt}} & [1]\ \square FEATURE,\\ [1] & \overset{+Value}{\bullet\!\!\blacklozenge\!\!\rule[0.5ex]{3em}{0.4pt}} & [*]\ \square VALUE\end{array}\right\rangle;\right\} \tag{16}$$

$$Characteristics\left\{\left\langle\begin{array}{lcl}[1] & \overset{+Characteristics}{\bullet\!\!\blacklozenge\!\!\rule[0.5ex]{6em}{0.4pt}\!\!\blacklozenge} & [1]\ \square CHARACTERISTICS,\\ [1] & \overset{+Property}{\bullet\!\!\blacklozenge\!\!\rule[0.5ex]{4em}{0.4pt}} & [*]\ \diamond Property\end{array}\right\rangle;\right\} \tag{17}$$

$$Component\left\{\begin{array}{l}\left\langle\begin{array}{lcl}[*] & \overset{+Characteristics}{\bullet\!\!\rule[0.5ex]{6em}{0.4pt}} & [1]\ \diamond Characteristics,\\ [*] & \overset{+Placement}{\bullet\!\!\rule[0.5ex]{4em}{0.4pt}} & [1]\ \square REGION,\\ [*] & \overset{+Link(\diamond\{State,Component\})}{\bullet\!\!\rule[0.5ex]{10em}{0.4pt}} & [1]\ \diamond State\end{array}\right\rangle;\\ \qquad\qquad\qquad \xrightarrow{f^v}\triangleright State;\end{array}\right\} \tag{18}$$

$$State\left\{\left\langle\begin{array}{lcl}[*] & \overset{+Template}{\bullet\!\!\rule[0.5ex]{4em}{0.4pt}} & [1]\ \diamond Template,\\ [1] & \overset{+Component}{\bullet\!\!\blacklozenge\!\!\rule[0.5ex]{4em}{0.4pt}} & [*]\ \diamond Component\end{array}\right\rangle;\right\} \tag{19}$$

$$Template\left\{\left\langle [1]\ \overset{+Region}{\bullet\!\!\blacklozenge\!\!\rule[0.5ex]{3em}{0.4pt}}\ [1..*]\ \square REGION\right\rangle;\right\} \tag{20}$$

$$UserInterface\left\{\left\langle [1]\ \overset{-State}{\bullet\!\!\blacklozenge\!\!\rule[0.5ex]{3em}{0.4pt}}\ [1..*]\ \diamond State\right\rangle;\right\} \tag{21}$$

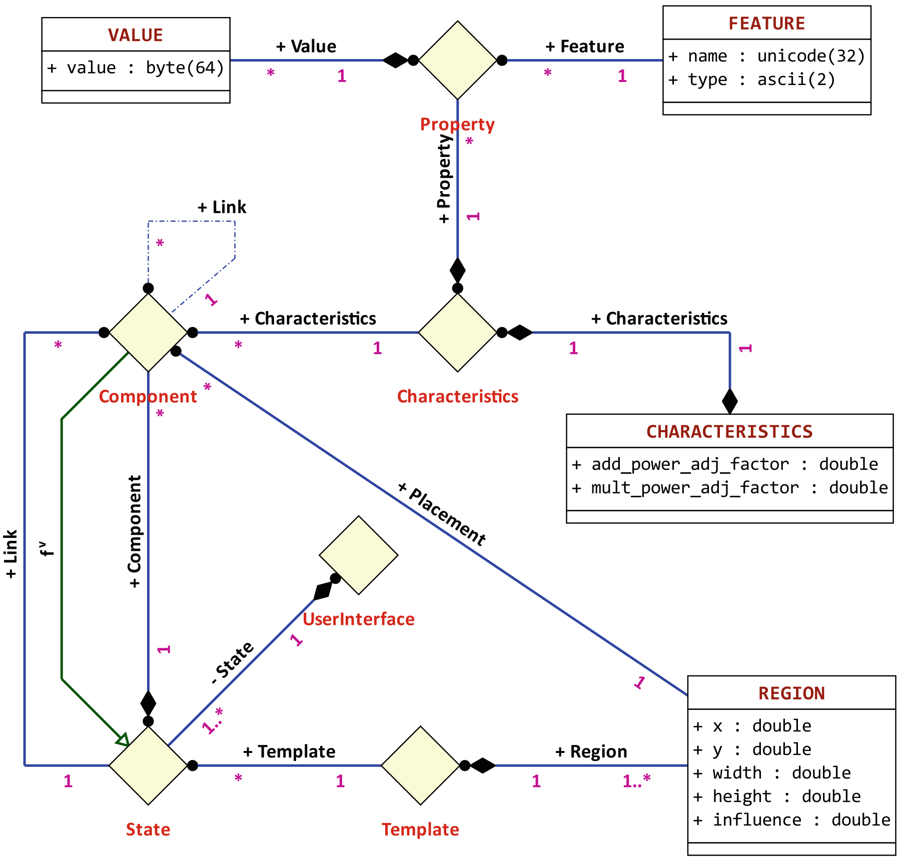

Fig. 1. Database structure for data retrieval system expressed in AML

3 Component Influence Power

The algorithm used for assessing *component influence power* is mainly based on computing it from both design characteristic of the component and the position of the component on interface canvas. These two interlate with each other but they are also quite different. The interface is defined as a set of interface states that are organized as graph. Individual interface states are understood as graph nodes, while links connecting these states are edges. In most common approach, i.e. traffic statistics, advertisement statistics, this level of abstraction is sufficient. However, for the purpose of interface design evaluation it is important to assess performance of each individual element that directs to another state of the interface. It is due to the fact that it is the only way to understand how users interact with system and how to enhance desired users activities.

The proposed factor is built according to following formula:

$$cip = pip + dip \tag{22}$$

where:

cip – component influence power,
pip – position influence power,
dip – design characteristic influence power.

As it was pointed out earlier, the influence power derived from the position of the element and its physical characteristic have different origin and properties. The idea how to understand them is presented in following subsections.

3.1 Position Influence Power

Each system that have human interaction component uses some kind of interface. For the sake of further deliberation the graphical interface will only be evaluated, since other types of interfaces are not in the thematic scope of presented method. It is assumed that the individual interfaces are build with the use of templates. Each template provide means for component distribution over interface canvas by the use of regions. The regions are used as containers for components and it is assumed that each and every component assigned to a given container have the same influence power, i.e. *pip*. The power distribution that defines influence power values assigned to region lays in discretion of the designer. However, Fig. 2 provides an example, where vertical and horizontal power distribution were shown.

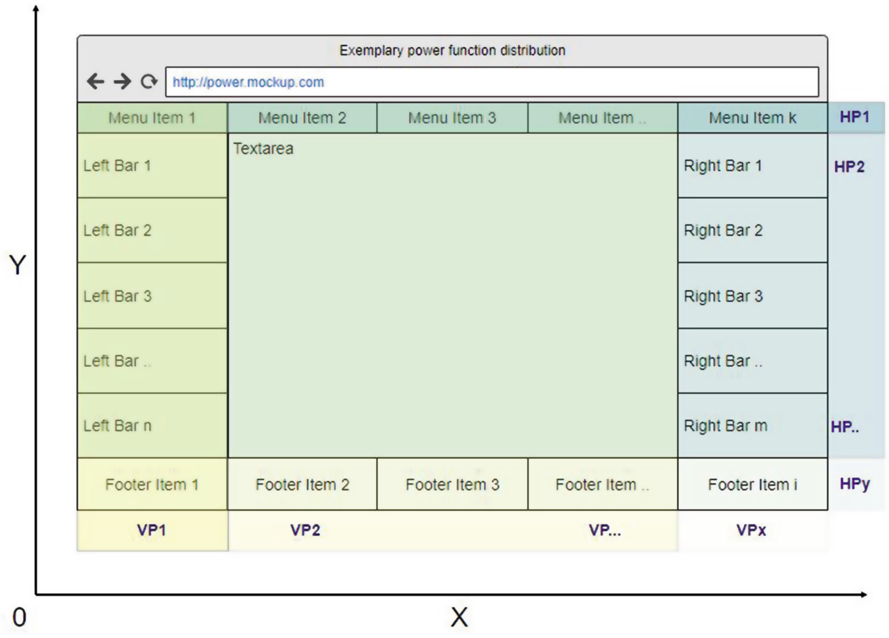

Fig. 2. Exemplary template with power distribution assignments

The function that defines power of each element may be defined according to either absolute position of this element on canvas or its relative position in reference to other elements. Those two approaches might be mixed for optimal power assignment.

3.2 Design Characteristic Influence Power

The second element *dip* in Eq. 22 refers to component design characteristics. It is assumed that physical appearance of components affect the influence that this component have on user decision, whether or not to use it. The definition of components (Eq. 6) assume that its design characteristics is derived from properties assigned to this component. For each property there might be additive or multiplicative power adjustment factor defined. Values of those factors are in the discretion of the designer, however the basic idea of why those should be applied is presented in the Fig. 3. The picture consists of five items aligned horizontally. One of the item differs from the other in spite of its color and an image positioned next to it. Those properties may imply presence of adjustment factors. The actual values of them depend on designer.

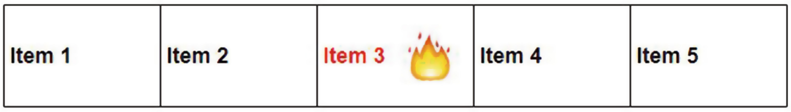

Fig. 3. Menu with one item having adjustment characteristic applied

3.3 Influence Power Evaluation

The influence power evaluation shall be assessed for each interface separately since the definition 22 and the overall construction of the interface definition system presented in this paper implies that the influence power is relative, rather than absolute. Therefore, for each interface state an influence power matrix L is created.

$$L = [l_{ij}] \tag{23}$$

where each l_{ij} is computed according to following equation

$$l_{ij} = (c_s, c_e, cip) \tag{24}$$

where:

c_s – start component in the link,
c_e – end component in the link,
cip – computed influence power.

3.4 Algorithm Definition

The complete algorithm for assessing and evaluating of interface design is presented in the Fig. 4. The first phase consist of several steps, each of which focuses on definition of elements building the interface, namely templates, regions, characteristics, components and states. Once the system is defined in terms of its states the second phase of algorithm is set in motion. The power of edges is computed and formed as influence power matrices, according to formula 23. Following the user experience data is acquired and stored. It is used for computing *Interface Design Effectiveness* – *IDE* matrix for each of states as:

$$IDE = [ide_{ij}] \tag{25}$$

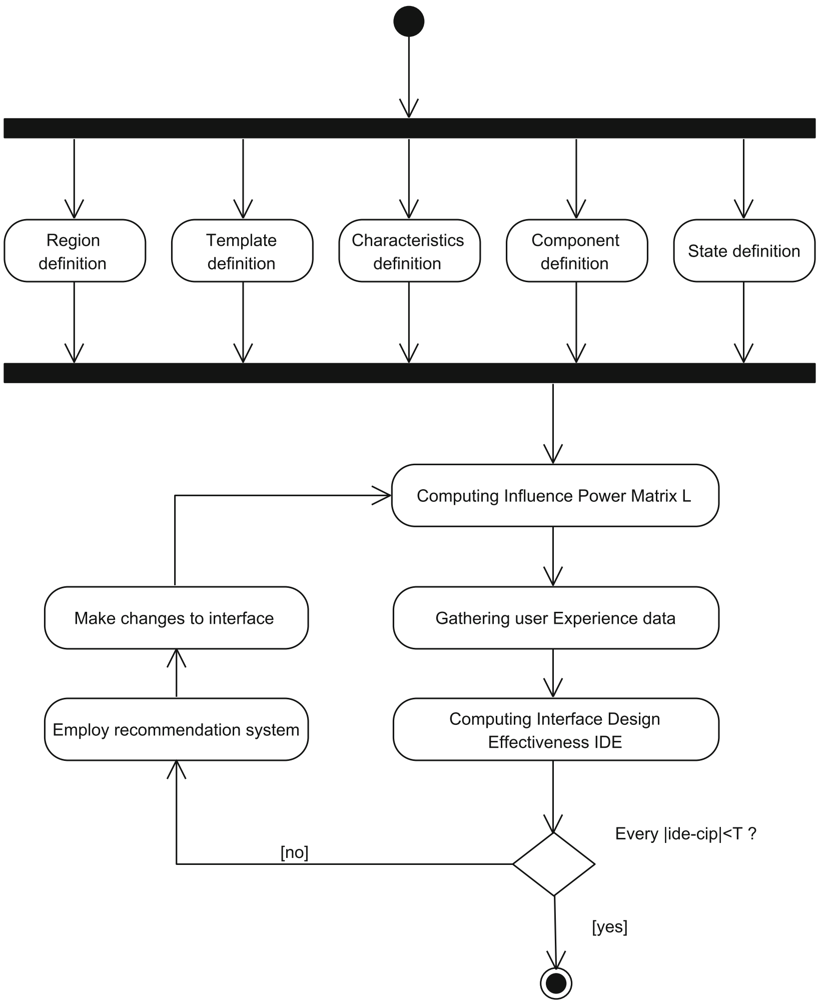
Region definition
Template definition
Characteristics definition
Component definition
State definition
Computing Influence Power Matrix L
Make changes to interface
Gathering user Experience data
Employ recommendation system
Computing Interface Design Effectiveness IDE
Every |ide-cip|<T ?
[no]
[yes]

Fig. 4. Algorithm for user interface effectiveness assessment

where each ide_{ij} is computed according to following equation

$$ide_{ij} = (c_s, c_e, uu) \tag{26}$$

where:

c_s – start component in the link,
c_e – end component in the link,
uu – user usage.

User usage is calculated as number of times the link was used divided by the sum of all outgoing links ware used from a given state. Having both L and IDE matrices it is possible to conclude whether the difference between computed influence power *cip* and interface design effectiveness *ide* go beyond the scope of threshold set by the designer. In case of that the optional phase of employing the recommendation system is executed. The algorithm is concluded if all components in all states are defined in the way that their *interface design effectiveness* states within the boundaries set by the threshold.

4 Summary

The paper is dedicated to the user interface definition in the aspect of building a recommendation system that would allow easy adaptation of interface design to obtain optimal effectiveness. This is important due to the fact that user interface gives means to utilize applications' functionalities. There is number of publications that propose guidance for proper interface design, including adaptive approach. However, authors in the paper did not put the main emphasis on recommendation system, but rather on building the framework that may be used for evaluating of interfaces design and algorithms that aim at its recommendation. Thus, the paper introduces general idea for definition of interfaces based on elements such as templates, regions, characteristics components and interface states. Furthermore, the component influence power – *cip* and interface design effectiveness *ide* are defined in order to assess the effectiveness of interface design. Last but not least, the general algorithm for interface recommendation system evaluation has been proposed.

References

1. Beel, J., Breitinger, C., Langer, S., Lommatzsch, A., Gipp, B.: Towards reproducibility in recommender-systems research. User Model. User Adap. Inter. **26**(1), 69–101 (2016)
2. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. Knowl. Based Syst. **46**, 109–132 (2013)
3. Cooley, R., Mobasher, B., Srivastava, J.: Web mining: information and pattern discovery on the world wide web. In: Proceedings Ninth IEEE International Conference on Tools with Artificial Intelligence, pp. 558–567, November 1997. https://doi.org/10.1109/TAI.1997.632303

4. Isinkaye, F., Folajimi, Y., Ojokoh, B.: Recommendation systems: principles, methods and evaluation. Egypt. Inform. J. **16**(3), 261–273 (2015)
5. Krótkiewicz, M.: A novel inheritance mechanism for modeling knowledge representation systems. Comput. Sci. Inform. Syst. (2017). https://doi.org/10.2298/CSIS170630046K
6. Krótkiewicz, M.: Association-oriented database model – n-ary associations. Int. J. Softw. Eng. Knowl. Eng. **27**(02), 281–320 (2017). https://doi.org/10.1142/S0218194017500103
7. Krótkiewicz, M.: Cyclic value ranges model for specifying flowing resources in unified process metamodel. Enterp. Inform. Syst., 1–23 (2018). https://doi.org/10.1080/17517575.2018.1472810
8. Krótkiewicz, M., Jodłowiec, M.: Modeling autoreferential relationships in association-oriented database metamodel. In: Świątek, J., Borzemski, L., Wilimowska, Z. (eds.) ISAT 2017. AISC, vol. 656, pp. 49–62. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-67229-8_5
9. Kukla, E., Nguyen, N.T., Sobecki, J., Danilowicz, C., Lenar, M.: Determination of learning scenarios in intelligent web-based learning environment. In: Orchard, B., Yang, C., Ali, M. (eds.) IEA/AIE 2004. LNCS (LNAI), vol. 3029, pp. 759–768. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24677-0_78
10. Malski, M.: A Method for web-based user interface recommendation using collective knowledge and multi-attribute structures. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) ICCCI 2011. LNCS (LNAI), vol. 6922, pp. 346–355. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23935-9_34
11. Mican, D., Tomai, N.: Association-rules-based recommender system for personalization in adaptive web-based applications. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 85–90. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16985-4_8
12. Montaner, M., López, B., de la Rosa, J.L.: A taxonomy of recommender agents on the internet. Artif. Intell. Rev. **19**(4), 285–330 (2003). https://doi.org/10.1023/A:1022850703159
13. Shahabi, C., Banaei-Kashani, F.: A framework for efficient and anonymous web usage mining based on client-side tracking. In: Kohavi, R., Masand, B.M., Spiliopoulou, M., Srivastava, J. (eds.) WebKDD 2001. LNCS (LNAI), vol. 2356, pp. 113–144. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45640-6_6
14. Sobecki, J.: Ant colony metaphor applied in user interface recommendation. New Gener. Comput. **26**(3), 277 (2008). https://doi.org/10.1007/s00354-008-0045-9
15. Srivastava, J., Cooley, R., Deshpande, M., Tan, P.N.: Web usage mining: discovery and applications of usage patterns from web data. SIGKDD Explor. Newsl. **1**(2), 12–23 (2000). https://doi.org/10.1145/846183.846188
16. Wojtkiewicz, K., Jodłowiec, M., Krótkiewicz, M.: Association-oriented database metamodel: modelling language (2017). https://doi.org/10.13140/RG.2.2.18483.73769