A Framework for Ranking of Software Design Patterns

Shahid Hussain(✉), Jacky Keung, and Arif Ali Khan

Department of Computer Science,
City University of Hong Kong, Kowloon Tong, Hong Kong
{Shussain7-c, aliakhan2-c}@my.cityu.edu.hk,
Jacky.Keung@cityu.edu.hk

Abstract. Several software design patterns have been familiarized either in canonical or as variant solutions in order to solve a problem. Novice designers mostly adopt patterns without considering their ground reality and relevancy with design problems, which may cause to increase the development and maintenance efforts. In order to realize the ground reality and to automate the selection process, the existing automated systems for the selection of design patterns either need formal specification or precise learning through training the numerous classifiers. In order to address this issue, we propose an approach on the base of a supervised learning technique named 'Learning to Rank', to rank the design patterns with respect to text similarity with the description of the given design problems. Subsequently, we also propose an evaluation model in order to assess the effectiveness of the proposed approach. We evaluate the effectiveness of the proposed approach in the context of several design pattern collections and relevant design problems. The promising experimental results indicate the applicability of the proposed approach.

1 Introduction

In software development life cycle, software design is considered as a most challenging task as compared to other activities. An architecture-based design method is one of main category which can be realized during the evolution process. In this category, different software architectural styles (presented through design components and their relationship) are applied and can be considered as coordination tools among the activities of the software development life cycle. For example, it provides a bridge between satisfactions of system's critical requirements to implementation [1]. The most common architecture-based design method is Attribute Driven Design (ADD) which incorporates the different architectural strategies and patterns to fulfill the quality attributes. In the working procedure of ADD, a software developer selects an architectural style and chooses the desired quality attributes included in the software system. Subsequently, software developers use the description of design requirements related to selected architectural component and employed software design patterns [2]. Over many years, based on their experiences, software developers have encapsulated and suggested the proven solutions to satisfy the recurring design problems. Subsequently, the experience-based solutions are organized and realizes as a standardized form for

L. Barolli and O. Terzo (eds.), *Complex, Intelligent, and Software Intensive Systems*,
Advances in Intelligent Systems and Computing 611, DOI 10.1007/978-3-319-61566-0_20

design patterns. For example, the canonical solution of Composite GoF (Gang-of-Four) design pattern is the result of formulating the recurring structural and compound design problems related to the composition of objects from its nesting and building tree structures [3].

The employment of design patterns in software development is considered as an alternative approach to software refactoring to address the declined quality of the software system. Besides, other advantages to employ the design patterns are; to maintain consistency between design and implementation, enabling the relationship between developers, increase reusability and software modularization [3]. However, due to the existence of an alternative solution of spoiled patterns [4], the challengeable design patterns [5], and interdisciplinary design patterns [6], it is so difficult for a novice designer to find appropriate design pattern(s) and determine its applicability in order to solve a design problem. Since design patterns are formulated on the base of software developer's experience, consequently, a designer with no or little experience with design patterns have concerns how to find the best one.

In order to address this issue, we propose an approach to rank the design patterns on the base of their text similarity with given design problem(s). The proposed approach is formulated through a supervised learning task named 'Learning to Rank' which can help to rank the design patterns on their ground truth. The current efforts to automate the design pattern selection process can be divided into three approaches named UML-Based [7–9], Ontology-Based [10–12] and Text categorization based approach [13–15].

Subsequently, for the effective results of automated systems which based on Text categorization approach, there is need of an ample training sample size or separate classifiers for each pattern class in order to solve the multi-class problem [16]. However, it is not easy in the existence of complex and interdisciplinary design pattern

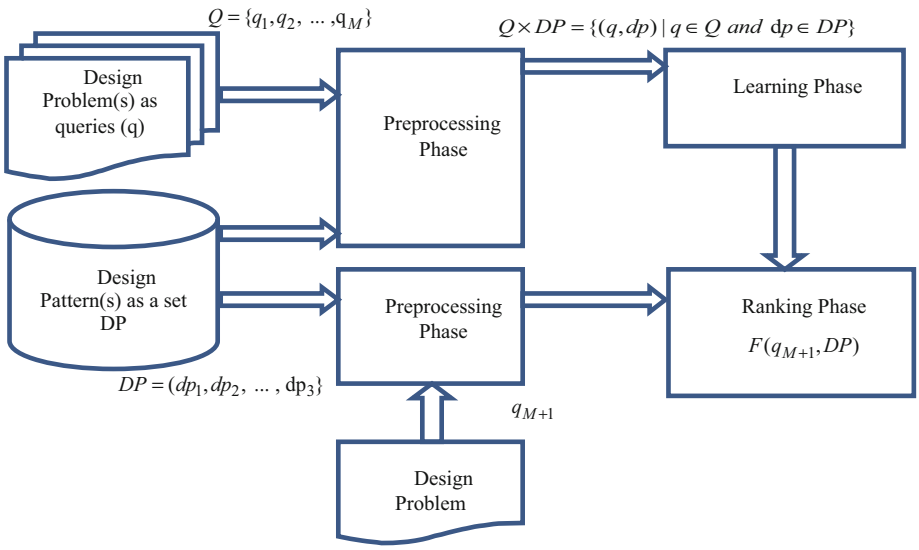

Fig. 1. Overview of proposed approach

collection, where numerous classifiers will be required to make effective learning for each pattern class [5, 7]. In this study, we consider design pattern(s) selection as information retrieval problem [13–15] and proposed an approach to rank the design pattern(s) for given design problem(s) using a supervised learning task named 'Learning to Rank'. There are numerous list-wise ranking algorithms (Coordinate Ascent, AdaRank and LambdaMART) and, point-wise and pair-wise (RankNet, MART and RankBoost) have been applied in the domain of information retrieval applications [17]. However, in this paper; we incorporate only LambdaMART [18] in the proposed approach. The LambdaMART has been applied to combine the strengths of the boosted tree classification and LambdaRank (which is described through neural network models). The framework of the proposed approach is shown in the Fig. 1.

2 Related Work

In 1977, Alexander, the father of patterns describe a roadmap to capture the design knowledge and present through a collection of patterns in order to help the architects and engineers. According to Alexander, a pattern can be described in three sections context, a problem description (we used this section in the preprocessing phase of proposed approach, described in Sect. 4.1) and a corresponding solution. However, the building components of each section vary which depend on the developer's experience and knowledge in the response of handling design problems. For example, in the domain of object-oriented development Gamma et al. [3] present 23 design patterns and classified them into three groups named Creational, Structural and Behavioral. Subsequently, we summarized the researcher's efforts made to automate the selection of design patterns which are UML-Based, Ontology-Based and Text categorization based approaches.

D-K Kim and C.E. Khawand depict an approach to select the design pattern by specifying its problem domain. They reported that problem domain of design patterns can help to describe the known problems in the context. The authors used Role-Based Modeling Language (RBML) as UML based pattern specification language to describe the Meta-model to formalize the design patterns [7]. Hasso and Carlson consider the problem description part of design patterns; perform semantic analysis of sentences, and on the base of words meaning which suggest the classification of design patterns. In their study, the authors perform semantic analysis using linguistic theory and consider the meaning of verbs for the classification of design patterns [10].

S.M.H. Hasheminejad and S. Galileo [13], and S. Hussain et al. [14, 15] consider the design pattern selection as an information retrieval problem, used the text categorization approach, and present the automated method to select the design pattern with respect to the degree of similarity between the description of design problems and problem definitions of the retrieved design patterns.

In both studies, the authors consider the classification and selection of design patterns with respect to the given design problem as a problem of information retrieval and used text categorization approach. Subsequently, the common objective of both studies is to obtain promising results with respect to ground truth. However, for the precise learning, either there is need of enough sample size or numerous classifiers to

overcome the multi class problem. Recently, a new machine learning technique 'Learning to Rank' has been applied in different information retrieval related application such as web search. The 'Learning to Rank' is a supervised learning technique which is used for the creation of the ranking model $f(q, d)$ to retrieve document d with respect to query q. In our study, we consider the design pattern selection as information retrieval problem and proposed an approach based on 'Learning to Rank' to select the appropriate design pattern(s) for given design problem(s).

3 Brief Description of Text Categorization

The text categorization approach is performed either through supervised or unsupervised learning techniques. The former techniques use class labels assigned to documents and latter techniques use attributes of the data and dis(similarity) measures to automate their learning. Text preprocessing is mandatory for the text categorization and should be performed before the learning process [11]. The Preprocessing, indexing and feature selection are the main steps which are performed during the text preprocessing. In the preprocessing step, two activities are performed to transform documents into strings. The first activity is to remove stop words that are more frequent words and carry no information, such as Preposition, Conjunction, and Pronouns. The second activity is word stemming, which is performed to make a group of words that have same conceptual meaning. Subsequently, the numbers of words in the documents collection are reduced. Porter's stemmer is a well-recognized stemming algorithm [19], used to transform English words into their stem iteratively through a set of rules. Subsequently, in indexing step, Vector Space Model (VSM) is used as well-known indexing method to describe the documents. In VSM, each document is described by a vector of words. Commonly, a document collection is described through a word-by-document matrix D, where each entry refers to the word's occurrence in the document.

$$D = (W_{wd}) \tag{1}$$

In the Eq. 1, W_{wd} is the weight of word w in document d. There are many ways to determine the weight W_{wd} of word w in document d. For example, Binary, Term Frequency (TF), Term Frequency Inverse Document Frequency (TFIDF), Term Frequency Collection (TFC), Length Term Collection (LTC), and Entropy are commonly used weighting methods in text categorization [11]. The term weighting schemes are used to improve the performance and remove the noise from the documents. Finally, In feature selection step, different techniques are used to remove the features, however, computational time and classification accuracy are the main issues related to the discriminative power of each technique [16].

4 Overview of Proposed Approach

The rapid development of software design patterns leads to an increase in the amount of its electronic documentation worldwide. Therefore, there is a need to organize these patterns in order to reduce the time required to a novice designer to select the appropriate design pattern(s). This situation motivate us to consider this issue (design pattern organization and selection problem) as an information retrieval problem and use the text categorization approach to rank the design pattern with respect to text relevance with design problems. Though this issue is addressed by [13–15] in their study, however, in the case of complex and interdisciplinary design pattern collections where numerous classifiers are required to make supervised learning for each pattern class or need of a global filter-based feature method to construct a more representative feature set to overcome the multi-class problem.

We look out the capacity of automated text categorization in different domains and use in the proposed approach for two purposes, (1) to rank the design patterns with respect to the text relevancy with design problem(s) description and (2) automatically retrieve a more appropriate design pattern(s) for a given real design problem. The ranking tasks are performed by using local $f(q, d)$ and global $F(q, D)$ ranking models. These ranking are created to rank the set of documents D with respect to user's query q. The existing approaches BM25 model and LMIR (Language Model for IR) are implemented to rank the documents by creating f (q, d) without training, however, in order to achieve our objective, we employ a machine learning technique named 'Learning to Rank' to construct a ranking model for the selection of more appropriate design pattern(s) for a given design problem. As a supervised learning technique, 'Learning to Rank' require training and testing data. We incorporate a widely used list-wise ranking algorithm named LambdaMART in the proposed approach. Subsequently, we formulate the proposed approach in three phases Preprocessing, Learning (on training data) and Ranking (on testing data).

4.1 Preprocessing

The first phase of the proposed approach is preprocessing which applied to problem description of all design patterns of a desired collection and description of the given design problem. The sequence of preprocessing activities is depicted in Fig. 2.

The first two activities are performed to remove the stop words and words stemming consecutively. There are several text mining tools like JPreText (A simple Text

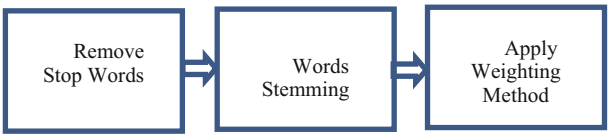

Fig. 2. Preprocessing activities

Preprocessing Tool)[1] and their APIs[2] (Application Programming Interfaces) are available to perform these tasks. The problem description part of each design pattern and design problem description is presented in the form of a feature vector. Subsequently, a Vector Space Model (VSM) includes non-repeated terms of all documents is created. The structure of Vector Space Model (VSM) with for N design pattern and a design problem with M non-repeated terms. Finally, in the last activity, one of weighting schemes (Sect. 3) is applied to the vector space model.

4.2 Learning Phase

In the domain of information retrieval, three approaches of 'Learning to Rank' are followed in order to overcome the corresponding problem. In the learning phase of proposed approach, we analyzed the list-wise ranking algorithm named LambdaMART due to the existence of design patterns either as variant or a spoiled form of canonical solution and ranking them with respect to text relevancy with the given design problem. Subsequently, LambdaMART satisfied the properties (a) consistency, (b) soundness, (c) convexity and computational efficiency regarding the Loss function point of view. Usually, two tasks are performed in the learning phase. The first task is performed to conduct learning on the pairwise (Problem-Pattern) preference and construct a training set which includes design patterns, design problems and their relevance levels. There are several ways to describe the relevance levels between design patterns and design problem, such as use of similarity measures [20] or use of labels to grade the patterns associated with the design problem. In this study, we assumed the latter approach in order to evaluate the pattern relevance with respect to a design problem. The formulization of training set for learning is shown as follow:

We assume that Q is design problem set (that is $\{q_1, q_2, \ldots, q_M\}$), where q referred to a design problem), DP is the pattern set (that is $\{dp_1, dp_2, \ldots, \mathrm{d}p_N\}$, where dp referred to a design pattern), and L is the label set (that is $\{1, 2, \ldots, l\}$) denotes different grades. The label set $L = \{l_{1,1}, l_{1,2}, \ldots l_{1,N}\}$ and $DP = \{dp_{1,1}, dp_{1,2}, \ldots, \mathrm{d}p_{1,N}\}$ pattern set are created with respect to the association with a given design problem q_1 and are ordered $l \succ l-1 \succ l-2 \succ \ldots \succ 1$ accordingly. Where $\succ$ depicts the order relation. Subsequently, a feature vector $v_{i,j} = \phi(q_i, dp_{i,j}),\ i = 1, 2, \ldots, m\,;\, j = 1, 2, \ldots, n$ is created from each problem-pattern pair $(q_i, dp_{i,j})$ where ϕ referred as feature function for a problem-pattern pair. Finally, the training dataset is created according to Eq. 2.

$$Training\ Set = \{(q_i, DP_i), L_i\}_{i=1}^{M} \tag{2}$$

The second task is performed to construct a local ranking model $f(q, dp)$ in order to assign score to a problem-pattern pair (q, dp) and global ranking model $F(q, DP)$ to assign a list of scores to problem-patterns pairs (q, DP). We assume that $F(.)$ is mapping function which maps a list of feature vectors $V = \{v_1, v_2, \ldots v_3\}$ to list of grades.

[1] http://sites.labic.icmc.usp.br/torch/msd2011/jpretext/.

[2] http://www.opencalais.com/opencalais-demo/.

The design problem and its associated documents form a group which is represented through i.i.d. data, while members of each group have no i.i.d [16].

4.3 Ranking Phase

In the ranking phase of proposed approach, we define the ranking list that is permutation $\pi_i \in \Pi_i$ for a given design problem q and relevant patterns DP_i. Where Π_i referred to the list of permutations π_i on P_i to present the position (or rank) of a pattern. For example, the position of the j^{th} pattern can be represented as $\pi_i(j)$. The testing dataset consist of a new design problem q_{M+1} and the associated patterns DP_{M+1}, shown in Eq. 3.

$$Testing\ Dataset = \{(q_{M+1}, DP_{M+1})\} \tag{3}$$

Subsequently, we create the feature v_{M+1} and used the ranking model (Sect. 4.2) to assign the grades to the patterns DP_{M+1} and output the ranking list π_{M+1} in sorted form.

4.4 Design Pattern Groups

In the domain of software engineering, some groups of experts have categorized the design patterns on the base their intent, motivation, and applicability. In this study, we consider two well-known design pattern collections in order to evaluate our proposed approach.

A. **Gang-of-Four (GoF) Design Pattern Collection**

The GoF collection includes 23 object-oriented design patterns which are divided into three groups named Creational, Structural, and Behavioral. This case study includes 1465 non-repeated words of all 23 design patterns after removing the stop words and words stemming [3].

B. **Douglass Design Pattern Collection**

The Douglass collection includes 34 real time systems relevant design patterns which are divided into five categories named Concurrency, Safety, and Reliability, Distribution, Memory, and Resource. This case study includes 1271 non-repeated words of all 34 design patterns after removing the stop words and words stemming [21].

4.5 Design Pattern Groups

Subsequently, we consider 20 real design problems from different resources [4, 22, 23] in order to evaluate the effectiveness of proposed approach; however, we mention the description of only one design problem due to space limitation.

- Design Problem

"An arithmetic expression consists of an operand, an operator (+ − */), and another operand. The operand can be a number, or another arithmetic expression. Thus, 2 + 3 and (2 + 3) + (4 * 6) are both valid expressions."

5 Evaluation Model for Proposed Approach

In the text categorization approach, Precision and Recall for the learners can be estimated using micro-averaging or macro-averaging equations, however, the precision of micro-averaging (used in this study) is better than macro-averaging. In order to select the best weighting method for ranking algorithms used in the proposed approach, the effect of each method such as Binary, TFIDF, TFC, LTC and Entropy on ranking algorithm is computed in term of Precision, Recall and F-measure [11]. Subsequently, weighting method with highest F-measure is chosen for the ranking algorithm.

$$P = \frac{\sum_{c=1}^{N} TP_c}{\sum_{c=1}^{N} (TP_c + FP_c)}, \quad \text{Micro - Averaging} \tag{4}$$

$$R = \frac{\sum_{c=1}^{N} TP_c}{\sum_{c=1}^{N} (TP_c + FN_c)}, \quad \text{Micro - Averaging} \tag{5}$$

$$F = \frac{2 \times P \times R}{(P + R)} \tag{6}$$

In the Eqs. 4 and 5, N is the number of design pattern classes decided to evaluate the performance of learning techniques used in the proposed approach. For example, in the case of GoF design pattern collection N is 3 (Sect. 4.3). The term TP is the number of design problems which are correctly identified for each design pattern class, FP is the number of design problems which are incorrectly identified for each design pattern class, and FN is the number of design problems which are missed from each corresponding design pattern class. The values of P, R, and F are computed using Eqs. 4–6 respectively.

Usually, the performance of ranking models is evaluated through a comparison between their computed ranking lists (computed permutations) and the ranking list given on the ground truth. There are numerous widely used performance measures in the domain of Information Retrieval (IR). For example, Mean Average Precision (MAP), Kendall's Tau, Discounted Cumulative Gain (DCG), and Normalized Discounted Cumulative Gain (NDCG). However, in this study, we used NDCG measure to evaluate the performance of the ranking function created through LambdaMART used in the proposed approach. The Normalized Discounted Cumulative Gain (NDCG) is an improved form of DCG, which can be computed through the Eq. 7.

$$NDCG(k) = G_{\max,i}^{-1}(k) \sum_{j:\pi_i(j) \le k} G(j)D(\pi_i(j)) \quad (7)$$

Where $G(.)$ and $D(.)$ referred as gain function and position discount function of pattern $DP_{i,j}$ in π_i. The summation is computed over the top k positions in the corresponding ranking list.

6 Evaluation Results

The proposed approach is evaluated according to the evaluation model described in the Sect. 5. In order to improve the homogeneity and completeness features, the most appropriate weighting method (in term of F-measure) has been used for the ranking algorithm LambdaMART used in the proposed approach. However, weighting method can be explicitly revealed. Though, we applied the proposed approach for 20 design problems related to the GoF and Douglass pattern collections. However, we give the detail analysis of only one problem given in Sect. 4.4. For this problem, the Composite, Decorator, Adaptor, and strategy are the candidate design patterns; however, the Composite design pattern should be more appropriate choice for a novice designer with respect to expert point of view. The performance of ranking algorithm used in the proposed approach depends on the creation of effective ranking list ranking with respective to relevance to the design problem. In terms of NDCG ranking metric criterion, the ranking of top 4 design patterns (Composite, Decorator, Adaptor and strategy) metric out of 23 design patterns indicate the effectiveness of the ranking of algorithms used in the proposed approach. We observed the average NDCG as 0.85 and 0.78 in case of GoF and Douglass design pattern collection respectively. These results suggest the applicability of the proposed approach.

7 Evaluation Results

In our study, we also find some threats. The first threat is related to a number of design pattern collection and design problem(s) which are considered during the evaluation process of our proposed approach. In order to evaluate the significance of results of proposed methodology, we will consider more design pattern collection (such as a security and Duynei's HCI design patterns group) and more examples of design problem(s).

The second threat is related to the incorporation of ranking algorithms. Our promising result based on the ranking function created through LambdaMART, we will consider more list-wise and pair-wise algorithm to improve the effectiveness of the proposed approach.

8 Conclusion and Future Work

In this paper, we proposed an approach using text categorization and exploit it through a supervised learning technique 'Learning to Rank'. We incorporate LambdaMART, a widely used list-wise ranking algorithm in the proposed approach. Subsequently, we also propose an evaluation model in order to determine its effectiveness. We evaluate the proposed approach through GoF and Douglass pattern collections and relevant design problem(s). In terms of NDCG performance metric criterion, we observe the significant performance (NDCG = 0.85 in case of GoF and NDCG-0.78 in case of Douglass pattern collection) of the proposed approach. In the future, we will consider more list-wise, point-wise, and pair-wise ranking algorithm and design pattern collections to improve the effectiveness of the proposed approach.

Acknowledgments. This research is supported by the City University of Hong Kong research funds (Project No. 7004683, 7004474 and 7200354).

References

1. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, 3rd edn. Addison-Wesley Professional, Upper Saddle River (2012)
2. Wood, W.G.: A practical example of applying attribute-driven design (ADD), Version 2.0, Technical report, Software Engineering Institute (2007)
3. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Boston (1995)
4. Bouhours, C., Leblance, H., Percebois, C.: Spoiled patterns: how to extend the GoF. Softw. Qual. J. **23**, 661–694 (2015)
5. Booch, G.: Handbook of Software Architecture (2006). http://handbookofsoftwarearchitecture.com/
6. Baraki, H., Kurt, G., Voigtmann, C., Hoffman, A., Kniewel, R., Macek, B-E., Zirfas, J.: Interdisciplinary design patterns for socially aware computing. In: Proceeding of 37th International Conference on Software Engineering (ICSE) (2015)
7. Kim, D.K., Khawand, C.E.: An approach to precisely specifying the problem domain of design patterns. J. Vis. Lang. Comput. **18**, 560–591 (2007)
8. Hsueh, N.L., Lin, C.-C., Kuo, J.-Y.: Object-oriented design: a goal-driven and pattern-based approach. J. Softw. Syst. Model. **8**(1), 1–18 (2007)
9. Kim, D.K., Shen, W.: Evaluating pattern conformance of UML models: a divide and conquer approach and case studies. Softw. Qual. J. **16**(3), 329–359 (2008)
10. Hasso, S., Carlson, C.R.: A theoretically-based process for organizing design patterns. In: Proceedings of 12th Pattern Language of Patterns (2005)
11. Hotho, A., Nurnberger, A., Paab, G.: A brief survey of text mining. J. Comput. Linguist. Lang. Technol. **20**, 19–62 (2005)
12. Khoury, P.E., Mokhtari, A., Coquery, E., Hacid, M.S.: An ontological interface for software developers to select security patterns. In: Proceedings of 19th International Conference on Database and Expert Systems Application, (DEXA 2008), pp. 297–301 (2008)
13. Hasheminejad, S.M.H., Jalili, S.: Design patterns selection: an automatic two-phase method. J. Syst. Softw. **85**, 408–424 (2012)

14. Hussain, S., Khan, A.A.K., Ebo, K.B.: A methodology to automate the selection of design patterns. In: Proceeding of 40th Annual Computer Software and Applications Conference (COMPSAC) (2016)
15. Hussain, S., Khan, A.A.K.: Software design patterns classification and selection using text categorization approach. Appl. Soft Comput. (2017). doi:10.1016/j.asoc.2017.04.043
16. Uysal, A.K.: An improved global feature selection scheme for text classification. Expert Syst. Appl. **43**, 82–92 (2016)
17. Li, H.: A short introduction to 'Learning to Rank'. IEICE Trans. Inf. Syst. **94**(10), 1854–1862 (2011)
18. Wu, Q., Burges, C.J.C., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. 'Learning to Rank' for Information Retrieval, Microsoft Research (2009)
19. Porter, M.F.: An algorithm for Suffix Stripping. J. Prog. Electron. Libr. Inf. Syst. **40**, 211–218 (2006)
20. Huang, A.: Similarity measures for text document clustering. In: Proceedings of NZCSRSC (2008)
21. Douglass, B.P.: Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems. Addison-Wesley/Longman Publishing Co. Inc., Boston (2002)
22. Silberschatz, A., Galvin, P.B., Gagne, G.: Operating System Concepts, 6th edn. Addison-Wesley, Reading (2002)
23. Shalloway, A., Trott, R.: Design Pattern Explained: A new Perspective on Object Oriented Design. Addison Wesley, Reading (2001)