Calhoun: The NPS Institutional Archive
DSpace Repository

Faculty and Researchers

Faculty and Researchers' Publications

2000

Software Reuse Technology Practices and Assessment Tool-Kit

Nada, N.; Luqi; Shing, M.; Rine, D.; Damiani, E.; Tuwaim, S.
IEEE

<https://hdl.handle.net/10945/43611>

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

<http://www.nps.edu/library>

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

Software Reuse Technology Practices and Assessment Tool-Kit

N. Nada, L. Luqi, M. Shing
Naval Postgraduate School
Computer Science Department
Monterey, CA 93943
Phone: (831) 656-4075
Fax: (831) 656-3225
nnada@cs.nps.navy.mil

D. Rine, E. Damiani, S. Tuwaim
George Mason University
Computer Science Department
Fairfax, VA. 22030
Phone: (703) 993-1530
Fax: (703) 993-1710
{Damiani, drine}@cs.gmu.edu

Abstract

This paper focuses on a CBS module of Knowledge-Based Tutoring System for Software Reuse Practices. The paper presents a public CBS using a validated Software Reuse Reference Model (SRRM). A CBS-SRRM allows the software engineer to improve reuse practice by being tutored with selected course material based on the user profile. This material is combined with actual practice-based knowledge derived from different case studies from software development organizations' reuse practices. A CBS-SRRM provides software engineers with a way to be tutored using positive lessons learned by other organizations. Our research focuses on achieving more effective means for software development organizations to find alternative educational (training) solutions to problems in software reuse practices.

Keywords: Knowledge-Base, Case-Based Reasoning Systems, Intelligent Tutoring, Distance Learning, Learning Environments, Web-Based Training Systems.

1. Intelligent Tutoring Systems

Traditional intelligent tutoring systems are based on the assumptions that a user's thinking process can be modeled, traced, and corrected. Based on the principles of Computer Assisted Instruction (CAI), intelligent tutoring systems would allow for a generic model that can be used for any individual.

There are four main components of an intelligent tutoring system. The user module (1) consist of the incorrect and incomplete knowledge that a user begins with. The expert module (2) contains the correct, expert-like knowledge that is to be transferred and learned. This transfer of learning occurs as a two-way communication process, made possible through (3) the graphical user interface (GUI). The pedagogical element (4) is the basis of the instruction, and it determines what instruction will be given at which point. Some intelligent tutoring systems go further, and incorporate full simulation as part of the instruction.

The term "intelligent" refers to the system's ability to know what to teach, when to teach it, and how to teach it. It must have the capacity to understand, learn, reason, and problem solve. It must be capable of identifying a user's strengths and weaknesses and establish a training plan that is consistent with these results. It can pick up relevant learning information from the user (such as learning style), and apply the best means of instruction for that particular individual. Throughout the instruction, the system makes judgments about what the user knows and how well she/he is

processing the information. The instruction can then be tailored to the user's needs. [Schnackenberg, 99; Hall, 90; Laurillard, 89; Newman, 89]

1.1. Software Reuse Reference Model (SRRM)

In recent years, reusability has become an important factor in the process of software development. In fact, the availability of reusable assets in development phases provides valuable support to design and implementation with software architectures by improving productivity, quality, and time-to-market [Nada, 97]. Industry has demonstrated that reuse of software assets will provide a basis for dramatic improvements in quality and reliability, speed of delivery, and in long-term decreases in costs for software development and maintenance. Some researchers estimate that even with a less than 50% reuse rate, component-based software development leads to reliability improvement as much as ten times that of development that is not component-based [Lim, 94].

Opportunistic software asset reuse will not always succeed if it is not based upon a supporting reference model for developing software [Sommerville, 96]. Hence, a Software Reuse Reference Model (SRRM) may be considered as a key starting element to implement, realize, and quantify such savings. The SRRM needs to include both technical and organizational activities required to implement reuse successfully.

1.2. Case-Based Systems (CBS)

Decision theory provides an intuitively appealing axiomatic framework for reasoning in the face of uncertainty. *Decision analysis*, on the other hand, provides the methodology to apply this framework to solve practical problems.

Since its inception around the 1960s, decision analysis has developed a number of powerful ideas and techniques to help people and organizations analyze and solve decision problems.

Case-Based Systems (CBSs) offer a knowledge architecture system for managing, sharing and accessing knowledge. A CBS unifies many previous forms of knowledge management into a single intuitive mechanism, providing access to a broad spectrum of on-line knowledge through a single access method. CBSs support such diverse knowledge types as structured data, free-text documents, activity patterns, and expert system knowledge bases. CBSs unify access methods such as query-by-example, free-text retrieval, decision trees, and case-based reasoning (CBR).

The CBS approach uses the technique of comparing a current situation (e.g. company profile) to a library of known solutions (e.g. successful professional practices). CBS has been applied to a range of classification and construction tasks. It is particularly useful in tasks where a formal set of rules, patterns, or algorithms for generating solutions is difficult to obtain, but where examples of correct solutions are readily available. These "previous solutions" are stored as "cases" in a case base. The case base can be used for multiple purposes, including training and human and automated decision-making. Because of this, a CBS can keep pace with a changing environment by adding and improving cases, eliminating the need for repeated software upgrades performed by knowledge engineers. Because of the simple knowledge representation, using case study templates and patterns, little expertise is required to maintain the CBS. The CB manager does not need to be a programmer [Aamodt, 94; Kolodner, 93; Riesbeck, 89; Schank, 94] There are two primary benefits to the use of CBSs.

1.3. CBS and SRRM Correlation

It is necessary for software developers to have systematic procedures supported by a CBS and a validated SRRM to provide a real starting point for good software assets reuse and adoption

decisions, utilization decisions, and management activities. In addition to a SRRM, an organization interested in moving into a reuse-oriented software development methodology also needs more detailed knowledge about how to implement the SRRM in the organization. Hence, access to a CBS with this more detailed knowledge would be very useful.

It is important for software reuse practitioners and new enterprises that are interested in adopting software reuse to access lessons learned, access more detailed knowledge about how to implement the SRRM in the organization, and access reuse experience of successful enterprises based upon a validated SRRM for the software reuse process. Accessing these three kinds of knowledge is but a first step in an iterative software improvement environment. Usually, it is important to know what lessons and experiences lead to improved software development. But it is equally important to be able to implement and practice the skills behind these lessons and experiences so that, by doing and not just knowing, measured improvements will occur. Hence, a second step is building an educational environment, based upon individual tutoring, where the knowledge accessed in the CBS can be incorporated into individualized learning based on implementation and practice of those skills that will, in turn, lead to measured improvement. Measured improvement can, in turn, lead to increased software assets quality and increased process productivity. Section 2.3 describes such a total CBS-SRRM educational environment where learning based upon individualized tutoring can take place.

The existence of a publicly accessed Reuse CB (National Reuse CB), via our CBS-RRM will help software industry and academia capture best practice-based knowledge derived from different software development organizations' reuse programs and activities. This reusable set of best practices available by use of our proposed CBS-SRRM could provide software industry and academia with a systematic way to capture and access the lessons learned by other organizations. This will promote recurrence of good reuse practices and improve current reuse processes by increased software quality and decreased effort and time to market.

Having a set of case studies that can be used to derive solutions to reuse problems from prior lessons learned will help to carry out the following: (1) Describe current problems and identify ways to avoid them in the future. (2) Predict opportunities and possible successes in applying reuse. (3) Derive new knowledge from ongoing research projects. (4) Better leverage best reuse practices. (5) Avoid unnecessary risks. (6) Better justify technical and business reuse decisions.

Our assertion is that the case studies and lessons learned would be reused more often if organizations that have successfully adopted, utilized, and managed reuse could indirectly help organizations with similar environments, problems, or situations, and are interested in adopting or researching software assets reuse, locate the information about best software assets reuse practices and decisions about whether or not to adopt, utilize and/or manage software development based upon reuse.

2. What Is Missing

Referring to our previous research in the area of software assets reuse [Nada 97, Rine and Nada 98], the practicing and researching software engineering communities are still in need of the following professional practice resources:

- A publicly accessed CBS for the software engineering community that can be easily used to identify lessons learned and reuse experiences from successful enterprises based upon a validated software reuse reference model for the software reuse sub-process within the general software development processes.

- Use of an applicable, conceptualized, effective, and validated software assets Reuse Reference Model that considers and incorporates all technical and non-technical aspects of the software reuse process.
- On-line Software Reuse Self-Assessment system.
- On-line Software Reuse Individualized Distance Learning system.
- Identification of effective software assets reuse processes and products metrics.
- Identification of standardized reuse practices, i.e. systematic software reuse methodology.

3. CBS-SRRM Knowledge Based Tutoring System

Based on the principles of Computer Assisted Instruction (CAI), CBS-SRRM tutoring systems would allow for a generic model that can be used for any individual who is involved in software development and engineering [Schnackenberg, 99.]

3.1. CBS-SRRM Overview

Our current project, funded by the NSF, investigates effective public Case-Based System (CBS) tool-kits using a validated Software Reuse Reference Model (SRRM). CBS-SRRM allows the software engineer to improve reuse practice by capitalizing on effective practice-based knowledge derived from different software development organizations' reuse practices. CBS-SRRM provides software engineers with a way to utilize lessons learned by other organizations. The system also promotes recurrence of good reuse practices.

The research focuses on a more effective means for software development organizations to find alternative solutions to problems in successful practice of reuse. We demonstrate that developing a CBS-SRRM that will allow software developers to learn how organizations similar to theirs have successfully adopted, utilized and managed this technology can support improved reuse practices. The plan is to research, develop, and make publicly available what we and our affiliates have learned through our evolving set of case studies, surveys, interviews, and experimental results. This plan is carried out by researching and developing a publicly accessible reuse practices CBS for the software engineering community, using lessons learned and reuse experiences from successful enterprises based upon a validated SRRM that incorporates important technical, organizational, and cultural factors needed in adopting, utilizing, and managing reuse technology.

3.2. CBS-SRRM Objectives

The main objective of this research is to develop a tutoring system including a knowledge-based web-based distance assessment module that is technically supported by Case-Based Reasoning (CBR) technology.

The objective is to motivate software developers to access a web-based tutoring system including an assessment module that will help them improve their software development process using reuse practices. The practical implication is to provide trainees with a demonstration of a more efficient, more effective, and publicly accessed assessment and teaching package that will enhance their learning outcomes, increase their productivity, and improve their products' quality in shorter time.

We have collected, and continue to collect, data from industry on actual processes used and experiences with software reuse. This data is collected and then presented on the Web in a standard form based on a validated model. The CBS-RRM also provides interface to allow users to describe their own environment and objectives and to receive the data corresponding to the recorded projects that best match their profile. Work such as this can be of great value for developers who are under increasing economic pressure to avoid building each new system from the ground up. It is also of value to the research community as an empirical basis for the validation of claims and methods related to software reuse.

3.3. A CBS-SRRM Tutoring System

There are four main components of the tutoring system. (1) The user profiling module that will qualify the user for a certain software engineering domain, and identify the user's or trainee's (user's) organization size. (2) The assessment module that will examine and assess the user's previous software reuse experience and his/her organization reuse potential, capability, RRM level, and the depth of users's knowledge and experience in reuse. This step will be followed by a pre-test to evaluate the user/trainee background knowledge on reuse; our prototype can identify 3 levels: initial, middle, or advanced. Based on the outcome of the previous two modules and the results of the user's pre-test, the user will be assigned to a certain level of training material. (3) The CBS module will use the profiling information to match the user with several case studies, and present the best software reuse practices that have been used by similar organizations. This module contains the correct, expert-like knowledge that is to be transferred and learned. (4) The fourth module contains the course material that fulfills the users' needs and matches their profile. So, this module is a pedagogical element that provides the basis of the instructional material, determines the user's gains after taking the course (through a pre-test), and declares if the user is in need of further learning material.

The current CBS-SRRM tutoring system allows software developers to learn how organizations similar to theirs have successfully adopted, utilized, and managed improved reuse practices enterprises based upon a validated SRRM that incorporates important technical, organizational, and cultural factors needed in adopting, utilizing, and managing reuse technology. We are researching, developing, and making publicly available what we and our affiliates have learned through our evolving set of case studies, surveys, and interviews, thereby making it available to the whole software engineering community.

3.4. CBS-SRRM Architecture

Using a web-based Distance Assessment and Tutoring system combined with the CBR system will provide tools to allow users and supervisors to have a good educational system to improve the individual's skills and knowledge in software reuse. The CBS-SRRM Architecture is depicted in Fig. 1. The remaining part of this paper will focus only on the CBS module.

Figure 1. CBS-RRM Architecture

Figure 2. CBS Architecture

3.4.1. Searching Requirements of the Best Practices CBS

Believing that analogues may provide a way to predict results based upon what has been true in the past, the CBS's searching mechanism will be developed along the lines of searching systems. It will maintain a CB of cases that represent the performance of best-practiced software reuse. When the partially known profile of a new organization is presented to the CBS, the search engine will search the CB, find the case(s) of organization(s) and its/their profile(s) that is/are most similar with the profile of the new organization, and finally predict the level of practice of the organization in the CB that will be the level of practice assigned to the new organization. We adopted the following CBS Architecture (Fig. 2).

3.4.2. Reuse Practice Cases: Development of CB Study Subjects

The participant subjects are software development organizations who (1) have already been case study participants and who are initially in our CB of best practices, and (2) are considering adopting, utilizing and managing software reuse. Our team worked on the identification and evaluation of new CB subjects. Initially, each organization will constitute a case that contains the profile of certain user attributes. Cases that include all of this information will comprise the space of CB cases. Cases that are lacking the final software reuse practice level assigned, but contain at least a subset of the remaining information, will be considered as test (input) cases. The choice of organizations that will comprise the CB cases and the organizations that will comprise the test cases will be 'pseudo-random'.

The CBS's task, researched and developed by our team, will be to find an appropriate value for the level of reuse practice attribute of an input case; therefore, this attribute is considered the solution data for a particular case in this domain.

3.5. Matching Requirements of the Best Practices CBS

We are now ready to describe the approximate user classification engine used by our training system. Users are identified by structures composed of six features. The first two of these features are used to identify the organization type to which the user's organization belongs. The remaining four features are:

- The (partially known) organization's reuse practice levels RL_1 , RL_2 , RL_3 in the first, second, and third stages of reuse adoption, utilization, and management
- The organization's practice level PL at the end of the evaluation period.

Given this partially known organization type's reuse practice level, i.e., given a test case, the CBS's task is to provide an approximate assessment of the organization's practice level within the class of the given organization type.

Approximate techniques were selected based on the fact that while decision theory provides an appealing framework for representing rich information structures, it often turns out to be computationally intensive for highly interactive systems. For our training application, this overhead alone precluded the use of fully formal decision-theoretic models of selection. Instead, each new user is classified interactively and incrementally, using the closest existing user structure as a default. In order to do so, the CBS search engine exploits a *matching method* [Goguen, 96; Herman, 97; Luqi, 92; Nguyen, 95; Steigerwald, 91] to find the case or cases that are closest to the test case. The procedure works as follows: when encountering a new user,

called A, our system first elicits A's information structure $\{RL_1, RL_2, RL_3, PL\}$, and then determines which user in the existing population presents the CB structure that is closest to A's. The CB structure of the closest user is then used to determine an initial default representation (or working model) of A's training needs. We are fully aware that adopting this approximate, *similarity-based* user modeling, we heavily rely on the *clustering hypothesis*, i.e. the assumption that people tend to form clusters according to their preferences or needs. This hypothesis has however been confirmed by many studies in many areas of market segmentation [Frank 72].

3.5.1. Distance measures in partial-knowledge situations

Any notion of closeness requires a *measure* of distance among users; indeed, various *distance measures* over fully and partially specified structures could be used for this application. For instance, in a full-knowledge situation, our set of features could be interpreted as a *vector*, and Euclidean distance could be straightforwardly used to compute the distance between the newcomer and the rest of the user population. Pre-computed clustering based on Euclidean distance would make this approach fast enough even for an interactive system. Another interesting possibility is simply to *sort* the new user's features according to their values (if the value is the same, use the feature name as a tie-break). After sorting the system can compute a *scoring function* from the newcomer's feature ordering to the orderings of the other users of the system, in order to identify the closest one. As a simple example, suppose that the scoring function provides a score of 1 for each feature occupying the same position in the two orderings, and 0 for each feature occupying a different position, going from a minimum of 0 (reverse order) to 4 (the same order). Note that, while very fast to compute, such a function does not provide the needed fine granularity (only 4 clusters would be present) for our system; however, different weights can be associated to preserved/not preserved feature pairs in the ordering to refine the function granularity.

In our case, however, pure scoring techniques cannot be applied, as it is a *partial-knowledge* situation: the CB structures of users are partially specified due to complete lack or uncertainty of data. In other words, users' feature sets cannot be totally ordered according to their values as some value are uncertain or missing. To the best of our knowledge, there is no general theory addressing the problem of defining distance measures on partial orders. However, a specific solution to our problem can be devised. Suppose the CB structures of two users are partially specified by two partial orders *A* and *B* of their features (as the values of unknown features are not available, they might occupy any position). Then, we consider an approximation of the *worst-case behavior* of the total orders that extend *A* and *B*. Due to the computational hardness of computing total orders, we simply put the unknown features at a "distant" position from each other, inserting them respectively from the top and from the bottom of the ordered ones, and then compute the weighted scoring function on the resulting total orders.

Note that this distance is NOT a metric on the set of partial orders, since the distance between two identical partial orders that are not complete orders is always positive. This, however, is desirable if the two orders represent the situation of two different users, since the "real" complete orders for the two may actually differ.

4. Conclusion

This paper focused only on the CBS module of Knowledge-Based Tutoring System for Software Reuse Practices. The paper presents a public CBS using a validated Software Reuse Reference Model (SRRM). A CBS-SRRM allows the software engineer to improve reuse practice by being tutored with selected course material based on the user profile. This material is combined with actual practice-based knowledge derived from different case studies from software development

organizations' reuse practices. A CBS-SRRM provides software engineers with a way to be tutored using positive lessons learned by other organizations. Our research focuses on achieving more effective means for software development organizations to find alternative educational (training) solutions to problems in software reuse practices.

Our future work will focus on presenting and integrating a comprehensive CBS knowledge-based tutoring system that supports distance learning and reuse self-assessment in combination with CBR and empirically validating the tutoring model.

5. References

- [1] Aamodt, A., and Plazas, E. "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches." *AI Communications* 7(1) (1994): 39-52.
- [2] Goguen, J.; Nguyen, J.; Meseguer, J.; Luqi, L.; Zhang, D.; Berzins, V. "Software Component Search," *Journal of Systems Integration* (special issue on Computer Aided Prototyping) Vol. 6, No. 2 (1996): 93-134.
- [3] Frank, R.E. Massy, W.F. and Y. Wind. *Market Segmentation*. Prentice-Hall, New Jersey, 1972.
- [4] Hall, P. & Wood, P. "Intelligent Tutoring Systems: A Review for Beginners." *Canadian Journal of Educational Communication* 19(2) 1990: 107-123.
- [5] Jaber., K., Nada, N., and Rine, D. "Towards the Design and Integration of Multi-Use Components." *Proceedings of the International Conference on Information Systems Analysis and Synthesis*, July 1998.
- [6] Kolodner, Janet L. *Case-Based Reasoning*. San Francisco, California: Morgan Kaufmann, 1993.
- [7] Laurillard, D. "The Pedagogical Limitations of Generative User Models." *Instructional Science*, 17 (1989): 235-250.
- [8] Lim, W. "Effects of Reuse on Quality, Productivity, and Economics." *IEEE Software* September, 11(5) (1994): 23-30.
- [9] Lim, W. *Managing Software Reuse*. Englewood Cliffs, NJ: Prentice-Hall, 1998.
- [10] Luqi, Y. Lee. "Towards Automated Retrieval of Reusable Software Components." *Proceedings of the AAAI Workshop on Artificial Intelligence and Automated Program Understanding*, San Jose, CA, July 13, 1992. 85-88.
- [11] McDowell. *A Reusable Component Retrieval System for Prototyping*. MS Thesis, Naval Postgraduate School, September 1991.
- [12] McClure, C. *The Three Rs of Software Engineering: Reengineering - Repository - and Reusability*. Englewood Cliffs, NJ: Prentice-Hall, 1992.
- [13] Morisio, M., Ezran, M., and Tully, C. "Introducing Reuse in Companies: A Survey of European Experiences." *Proceedings of the 1999 Symposium on Software Reuse, ICSE-99, IEEE and ACM*, 1999.
- [14] Nada, N., Rine, D., and Tuwaim, S. "Best Software Reuse Practices Require Reusable Software Architecture in Product Line Development." *Proceedings of the Second Workshop on Software Architectures in Product Line Acquisitions*, June 1998.
- [15] Nada, N., and Rine, D. "Software Reuse Reference Model: Development and Validation." *International Conference on Software Reuse*, Victoria, Canada, June 1998.
- [16] Nada, N., Rine, D., and Tuwaim, S. "Practices in Organizational Structure for Software Reuse." *Proceedings of International Conference on Information Systems Analysis and Synthesis*, July 1998.
- [17] Nada, N., Jaber, K., and Al-Daijy, E. "A Product-Line Model." *Object-Oriented Programming Systems, Languages, and Applications (OOPSLA'98)*, Vancouver, Canada, October 1998.

- [17] Nada, N., and Rine, D. "Modeling and Designing Global Data and Information Systems Under Software Reuse Emerging Technologies." Fourth International Conference On Computer Science and Informatics (CS&I'98), Special Session on Software Reuse, October 1998.
- [18] Nada, N. and Rine, D. "Component Management Infrastructure: A Component-Based Software Reuse Reference Model." Proceedings of the ICSE98 International Workshop on Component-Based Software Engineering, Japan.
- [19] Nada, N., Rine, D., and Jaber., K. "Towards Components-Based Software Development." Proceedings of European Reuse Workshop (ERW'98), Spain, November 1998.
- [20] Nada, N., Rine, D., and Jaber., K. "Using Adapters to Reduce Interaction Complexity in Reusable Component-Based Software Development." Proceedings of the Symposium on Software Reusability (SSR'98), in conjunction with the International Conference on Software Engineering (ICSE'99), Los Angeles, May, 1999.
- [21] Nada, N. and Jaber, K. "Experimental Model to Validate Software Reuse Technology." Proceedings of the International Symposium on Computer and Information Sciences (ISCIS'99), Izmir, Turkey, October 18-20, 1999.
- [22] Newman, D. "Is a User Model Necessary? Apprenticeship as a Model for Intelligent Tutoring Systems." Proceedings of the Fourth International Conference on Artificial Intelligence and Education, 1989. 177-184.
- [23] Nguyen, D. An Architectural Model for Software Component Search Ph.D. Dissertation, Naval Postgraduate School, December 1995.
- [24] Riesbeck, C. and Schank, R. Inside Case-Based Reasoning. Hillsdale, N.J.: Lawrence Erlbaum, 1989.
- [25] Rine, D. and Sonnemann. "Investments in Reusable Software: A Study of Software Investment Success Factors." Journal of Systems and Software vol. 41 (1998): 17-32.
- [26] Rine, D., Nada, N., and Tuwaim, S. "Practices in Organizational Structure for Software Reuse." Proceedings of the IEEE SCI Conference, vol. 1, 1998.
- [27] Rine, D., and Nada, N. "A Validated Software Reuse Reference Model Supporting Component-Based Management." Proceedings of ICSE98, Japan.
- [28] Rine, D. and Nada, N. "Software Reuse Reference Model: Development and Validation." Journal of Information and Software Technology, in press, (1999).
- [29] Rine, D. and Nada, N. "An Empirical Study of a Software Reuse Reference Model." journal submission under review, (1999).
- [30] Schank, R., Riesbeck, C., and Kass, A. Inside Case-Based Explanation. Hillsdale, N.J.: Lawrence Erlbaum, (1994).
- [31] Schnackenberg, H. Class Notes, Introduction to Educational Computing (ETEC 560/660), Concordia University, Montreal, Canada, (1999).
- [23] Sonnemann, R. Exploratory Study of Software Reuse Success Factors. Ph.D. Dissertation, George Mason University, Fairfax, Virginia, Spring, (1995).
- [33] Sommerville, I. Software Engineering. 5th Edition, New York: Addison-Wesley, 1996.
- [34] Steigerwald, R., Luqi, L., McDowell, J. "A CASE Tool for Reusable Software Component Storage and Retrieval in Rapid Prototyping." Information and Software Technology England, Vol. 38, No. 9, 698-706, Nov. (1991).
- [35] Steigerwald, R. Reusable Software Component Retrieval via Normalized Algebraic Specifications. Ph.D. Dissertation, Naval Postgraduate School, December (1991).
- [36] Herman, J. Improving Syntactic Matching for Multi-Level Filtering. MS Thesis, Naval Postgraduate School, September (1997).