Impact of Software Components Characteristics above Decision-making Factors

Asunción Gómez-Perez
Laboratorio de Inteligencia Artificial
Facultad de Informática
Universidad Politécnica de Madrid. SPAIN
+34 91 3367439
asun@delicias.dia.fi.upm.es

Adolfo Lozano
Área de Lenguajes y Sistemas Informáticos
Escuela Politécnica de Cáceres
Universidad de Extremadura. SPAIN
+34 924 257226
alozano@unex.es

ABSTRACT

During the software project planning phase, software engineers face the dilemma of deciding which of several software components to reuse, all of which meet appropriate requirements. The first thing they have to do in order to decide which is the best suited is to prioritize requirements. If the component is to undergo a process of adaptation for integration into the project, an evaluation has to be made. Software engineers must assess several characteristics and factors to select a component examining its relevant features, which will affect to their decision with different ratio, depending on company make-up or project peculiarities. This paper provides a guide as an aid for making these decisions, describing the relevant characteristics and factors of software components which will influence each decision-making issue.

Keywords
Software Components, Factors, Characteristics, Reuse.

1 INTRODUCTION

When software project managers plan product development they should consider the possibility of reusing components that have already been built in order to reduce development costs and times. In the long term, the only rational means of cutting software costs and raising quality is by means of reuse [9].

The problem is that not all software is developed with a view to reuse even within the same company, which means that the benefits of reuse, that is, not having to develop a product from scratch, can turn out to be disadvantages if the software is not prepared for this purpose. Moreover, even if software components (SC) meet optimum requirements for reuse, it is difficult to evaluate whether they are suited for a particular project.

Software project development companies have different demands for creating their product by means of reusing SCs. During the planning phase, they have to weigh up different dimensions, specific to economic, quality, planning or product risk demands, depending on company make-up or project peculiarities. As an aid for selecting one SC from several candidates, this paper presents a set of SC characteristics which have a direct impact on these decision-making issues.

We assume that the SCs that are to be analyzed have been designed for reuse, which means that the additional effort of extracting the component from the original system is not taken into account (for a consideration of this problem, see [3]). We also assume that a single SC is to be analyzed for a particular software project; the classification of SCs in repositories or libraries and the methods of accessing these components are beyond the scope of this paper.

2 THE USE OF COMPONENTS DEVELOPED FOR REUSE

There are a host of models that establish factors for measuring how much SC reuse there is in a project: amount of reused software (measuring code lines, function points, etc.), how reuse improves product quality or economic models that calculate reusable component investment costs/benefits. These models are set out and analyzed in a series of publications [6, 9, 15, 16]. However, there are no guides to assist project managers in selecting one reusable SC from several similar ones [17], all of which meet the same specifications. This choice will be made in the planning period before the development and, therefore, the decision made at this point will be critical and will have a considerable impact on project success or failure.

Reusing built components amounts to a big saving in terms of production and leads to high reliability, provided the SCs have passed the respective quality controls [16,18]. However, the demand for SC reuse differs widely and will depend on both the characteristics of the application to be

DIMENSIONS	Production time			Costs rating			Product quality		Development risk	
FACTORS CHARACTERISTICS	Training time	Adaptation time	Development time	Licenses price	Adaptation expenses	Development expenses	Effectiveness	Reliability	Feasibility	Capability
Quality of documentation	X	X	X							X
Methodology	X	X						X		X
Training help	X				X					X
Adaptation help		X			X					X
Development help			X			X				X
Modularity		X								
Complexity	X	X								
Size	X	X								
Requirement covered		X								
Interoperativity		X							X	
Tools quality		X								
Ease of handling	X	X	X							
Acquisition license				X						
Kind of maintenance				X				X		
Update cost				X						
Human, Hw & Sw resources					X	X	X			
Execution speed							X			
Output accuracy							X			
Applications								X	X	
Maturity								X		
Credibility								X		

Table 1. Table 1 shows the impact of software components characteristics above decision-making factors and dimensions.

developed and the organization that is to undertake the project.

According to the estimate of fitness for the project, reusable components can be classed as follows:

- **suitable**: can be reused immediately by the application development team and require no additional learning or adaptation effort for use. This type of “ideal” software usually consists of internal components built by the organization itself and which can be reused for a new project.
- **adaptable**: when software developed by others or even personally is reused, not all of the functional needs of the new project are generally met. This will involve having to adapt and develop software to meet all the demands of the project, with the resulting expenditure in terms of effort and time.

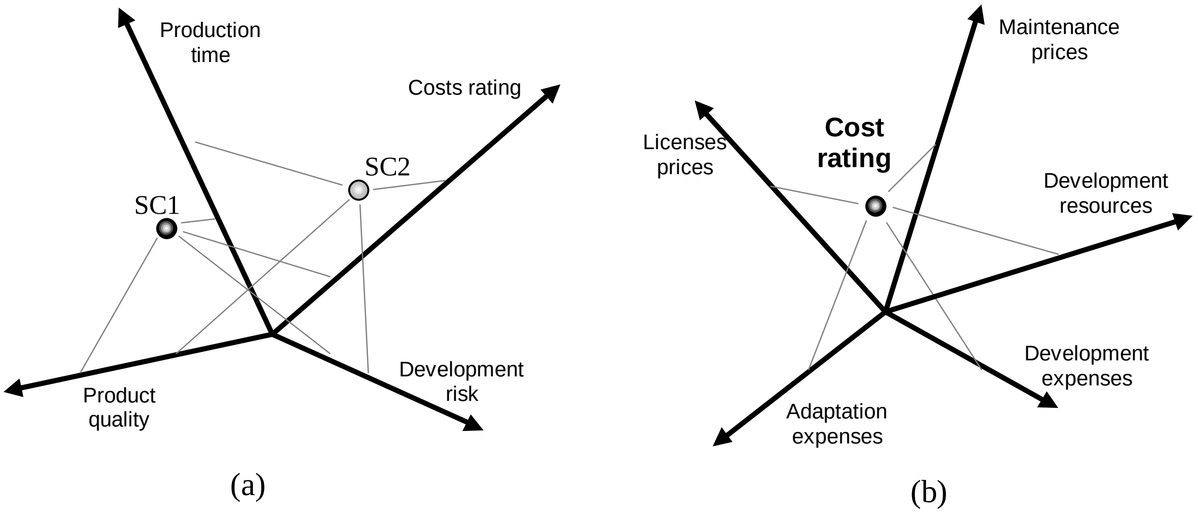

Figure 1. Figure 1a shows the estimates for two software components for reuse in a software project, having different values for the four decision-making dimensions. Figure 1b shows the decision-making factors and their impact on the costs rating dimension of a particular component.

1. Apart from the production effort involved in adapting and adding to the software that is to be reused, extra work will have to be done on studying its specifications and the conceptual and implementation compatibility of the new modules with those that have been selected for reuse.
2. Project managers should carry out the following tasks before adapting SCs to the needs of the new system:
3. Gather the candidate software components from libraries or repositories.
4. Study the characteristics of each SC against the same frame of reference.
5. Establish threshold for each factor involved in the decision on whether or not to reuse an adaptable SC.
6. Select the candidate that is best suited to the needs of the final product.
7. Decide who is to adapt (in-house development team, supplier or another company).

- **rejectable**: there are SCs which either do not belong to the project domain in question or cannot be reused even if adapted. These SC should be discarded during a preliminary selection phase.

3 BASIC ISSUES TO BE CONSIDERED IN SOFTWARE COMPONENT REUSE

Jones [10] considers that the software reuse process extends to different reusable "artifacts". Apart from source code or data, project plans, cost estimates, architectures, specifications and requirements models, designs, user documentation, and techniques, human interfaces and test cases can also be reused. However, although the requirements specification for two projects differ, the same factors should be taken into account, to which different importance will be attached in each case. The first task for the people who are to decide which components to reuse will be to establish the minimum requirements for each issue to be considered for their particular project and to be met by the reusable components.

In our experience, we have found how difficult it is to select and decide which SC to reuse both with regard to the reuse of functional procedures of traditional software

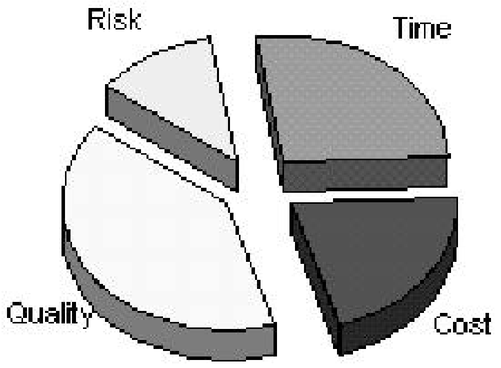

Figure 2. The four dimensions have different importance for different projects.

development at companies and the development of applications based on the reuse of knowledge-based software [7]. The proposed factors provide an overview for undertaking the study of any SC and is designed to offer a conceptual framework with which to analyze the fitness process of the above components.

We have identified four main dimensions (see figure 2) in the decision-making stage when using any type of reusable SC:

- **Production time**: project development time, after having selected a particular SC.
- **Costs rating**: capital investment to be made by the company in resources to be able to carry out the project with the aforesaid SC.
- **Final product quality**: assessment of the final product outputted. The characteristics of the component chosen will have an impact on final product quality.
- **Development risk**: probability of successfully outputting the final product by selecting a reusable component.

As shown in table 1, attached to each dimension, we have identified a set of factors that state the decision fundamental elements for SC selection, and for each factor we have identified a set of significant SC characteristics to consider before the SC selection process. Note that some characteristics of the reusable components can have an impact on several of the above factors and dimensions; a summary of the interrelation between characteristics, factors and dimensions is given in Table 1. It is also important to mention that the SC characteristics proposed will have different impact on each selection factor, and also each factor will have different influence on the dimensions. All project managers should weigh up each dimension, factor and characteristic to decide how important they are, relating them to the peculiarities and demands of their particular project. This n-dimensional space analysis can be illustrated graphically as shown in figure 1. Figure 1a shows two SCs which have been examined as possible candidates for reuse in a software project and how important each dimension is in their selection. Figure 1b shows how several factors have a different percentage influence on the development costs rating dimension for a particular SC.

Next sections describe in detail dimensions, decision-making factors and characteristics. The exposed features can be difficult to measure and sometimes can be inappropriate for a given project. However the engineer must take into account to obtain a best appreciation of the component suitability to his/her project. Some relevant characteristics of software component are represented in Figure 3.

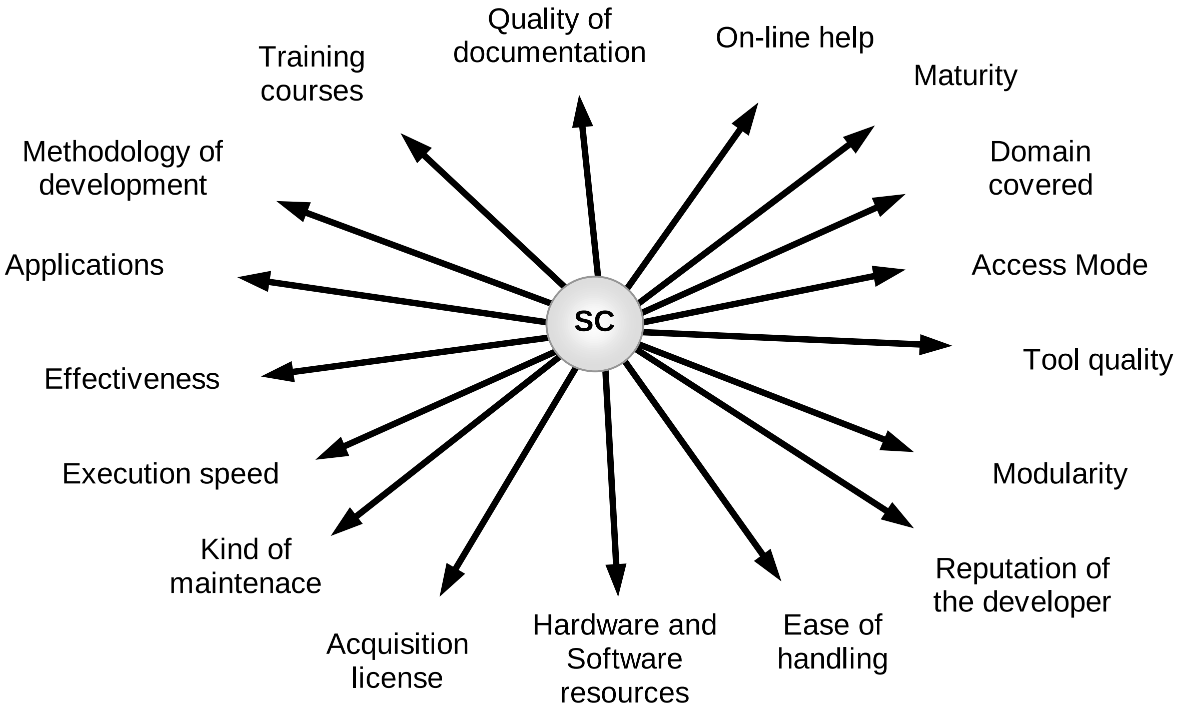

Figure 3. Some extractable software components characteristics for estimate its suitability.

4 ESTIMATING PRODUCTION TIME

As a general rule, software development companies operate with very rigid production times, on which their competitiveness depends. This will sometimes be a critical dimension when deciding whether to use a particular SC. The software engineers must establish (if it does not already exist) a maximum project termination time by means of which to reject reuse of a given SC, if the estimated production time is exceeded. Times must be forecast considering the SC adaptation options by supplier companies and by the in-house development team in search of the one that requires least effort [2]. The project scheduling process (see [19]) involves a series of tasks or milestones; only the SC characteristics to be considered as having an impact on application production time are discussed here:

Training time factor

It will take the development team some time to learn to use and modify the SCs that are to be reused, and, therefore, the manager will have to take several characteristics into account:

- construction methodology: We should analyze the entire engineering process used to create the SC in order to ascertain what knowledge the development team has of the process; if developers are unacquainted with the process, the time required to learn the methodology will have to be assessed. Also, it is important to take into account the complexity of the above methodology.
- quality of the documentation supplied about the SC: good documentation will reduce the time employed in learning how to use the component. In this paper, the documentation characteristic is considered as all the information expressed in natural language designed to assist in SC handling. This information can appear in external printed manuals, internal code explanations, execution help, etc.
- training help: component use training courses and on-line help support (via telephone or Internet) solves doubts about use immediately.
- ease of handling: a complicated and non-instinctive mode of accessing the component, with a complex nomenclature, will delay learning to use the component.
- Volume of the component: the size of a component could affect, between other factors, to the time of learning. The concept of volume will be fixed following an established criterion, determined for the asset type that is attempted reuse: lines of code or function points for modules of code, number of data for test data, number of files or records for data bases, number of phases for a design method, etc.
- Complexity: (referred any type of reusable component software), the complexity could be seen like the type and number of existent relationships between the component parts. This property should be evaluated of subjective way for the engineer, defining of any form and what a scale understands for complexity, and how it affect to the understandability. This external and subjective vision of complexity will attempt to define the group of attributes that they allow to value the complexity.

Adaptation time factor

Once the SC has been selected and once the development team knows how the SC operates, it can start to adapt it to the new system to be built. This adaptation will require less effort if there is a correspondence between the project needs and the adaptable component characteristics. Therefore, in order to decide whether it is to be adapted by the in-house development team or an external organization, the following component characteristics should be taken into account:

- requirement covered [1]: it is important to determine the percentage of project demands covered by the SC in order to calculate the work that will be needed to meet operating requirements.
- interoperativity [3, 12]: it is important to consider whether the SC access mode coincides with project requirements. It may be necessary to develop a wide range of adaptation modules, such as translation to other access modes, adaptation to execution environments, development of communications protocols, conversion to different storage structures, etc., to get the information or procedures supplied.
- complexity and volume: a component that has complex relationships between modules, and that it have an inadequate size, it will carry more time adapt it so that it complete the desired requirements. "If a component is too small, the costs combined of extraction, recuperation, and integration exceeds their intrinsic value. If it is too big, the component is prone to the error and has minor quality" [3].
- use of an appropriate construction methodology: components developed according to design and implementation standards are much easier to add modules to or reengineer.
- re-editing tool quality: some suppliers provide applications to visualize, modify and test their product; if no such tools exist, they may have to be designed by engineers in order to be able to reuse the component.
- adaptation help: the supplier may have appropriate curses and personnel specialized in queries about any design or implementation component of the product; even if the project development team has received

training, they may come up against conceptual or technical difficulties.

- quality of the documentation supplied: although this point is associated with the construction methodology, a clear and accurate documentation will reduce the time employed to adapt the software.
- modularity [12]: a component that has separable and integrated parts is better suited for adaptation. It is important to study whether it has an optimum number of modules, as while a larger number of modules is an aid to modification, it is an obstacle to integration to the integration process [13].
- ease of handling: unknown or difficult use characteristics will delay adaptation time.

Development time factor
Once the operation of the component is understood and the component has been adapted to the project, if so required by the specifications, all that remains is to reuse it in the project under development. However, the cost of the use of the component by the system will depend on:

- ease of handling: a complicated and non-instinctive mode of accessing the component, with a complex nomenclature, will delay project production.
- development help: technical development assistance is essential for immediately solving problems which could perhaps be insurmountable due to internal ignorance of the reused product.
- development documentation quality: good user manuals are an aid for proper use and for settling component handling doubts.

5 COSTS RATING

The available budget is a very important variable. Faced with two components of similar characteristics, we will always look to get maximum benefit [5]. The calculation of financial investment will also depend on the estimated production time. The expenses when selecting a particular SC will be characterized by the following summands:

Licenses price factor

- Acquisition license: the amount to be paid to the supplier will be taken into account, considering the possibility of being able to acquire the part of the component of interest in a modular manner.
- Maintenance and updates prices factor: periodic costs for the established maintenance characteristics or demanded by the supplier.

Adaptation expenses factor
If a component is to be adapted, two possibilities will be weighed up: that it be adapted by an external company or by the in-house development team. The amount of the investment for adapting the component, which will be determined by the component training time and the time used to develop the adaptation, will be accounted for; these times will generate expenses for the company which should be studied in each case. The following characteristics of the SC will also be considered:

- price of courses and on-line help for training.
- price of courses and on-line help for adaptation.
- characteristics of the hardware platform required to be able to use the component: processor, memory, storage capacity, communications lines, etc.
- characteristics of the software platform required to be able to use the component: operating system, communications programs, access management tools, etc.
- specialized personal required to adapt the component.

Development resources estimate factor
Investment in purchasing technology or adapting equipment and applications for developing the project using the SC. The following should be considered:

- price of courses and on-line help for development.
- characteristics of the hardware platform required to be able to use the component: processor, memory, storage capacity, communications lines, etc.
- characteristics of the software platform required to be able to use the component: operating system, communications programs, access management tools, etc.
- specialized personal required to use the component.

6 FINAL PRODUCT QUALITY EVALUATION

There are a host of papers that measure software quality from various development and operativity viewpoints [8, 11, 12]. Here, however, we state how the characteristics of the SC have an impact on final product quality. In order to assess customer satisfaction with a product1, software engineers can consider the following factors with regard to SC use:

Effectiveness factor [8]
In a broad sense, the system's capability of outputting good results in relation to the resources employed. The following characteristics of the SC will be involved in this:

- human, hardware and software requirements for operating with the component.
- execution speed: response time for outputs.
- output accuracy, in attaining the expected result.

[1] The customer may be the application developer, an organization that has commissioned the project or the anonymous sales market.

Reliability factor

“Probability of fault-free operation of a computer program in a particular environment in a given time” [14]. Offering the customer a product that can fail due to the component will devalue the article for sale, which means that the following must be taken into account:

- applications using the component: number and kind of the applications that already use the SC.
- methodology used in construction: a good component development methodology will lead to a low probability of system failure.
- maturity: component development date, number of checks conducted, errors detected, number of repairs in the last year, date of the latest modification due to error, etc. [11].
- credibility of the developer organization: a company with a good reputation in component development will give us a bigger guarantee that exhaustive quality controls have been performed.
- characteristic of the clauses of the licenses of maintenance: they will study the agreements of the contracts of maintenance, in order to evaluate the guarantees that they cover in any failure of the component.

7 DEVELOPMENT RISK

Even if engineers or project managers have managed to find the right component that appears to perform all the tasks it should, at the price they expected, with apparently suitable development times and that can offer the customer acceptable quality levels, the danger involved in choosing a component that has not been sufficiently tested, has been developed according to a poor (or no) methodology or does not provide any help for the technical difficulties when developing the product should still be considered [4]. So that one could get a product with these characteristics, they should give two conditions: that it “can" make and that it “know" make. In order to evaluate these two factors they will observe the characteristics of the component, and their relationship with the project and the personnel of development. Generally, the following reusable component characteristics will have an impact on project development risk:

Feasibility

If for the description of the component the engineer deduces that match to the demands of the new project, he/she owe also examine if the processes of adaptation that are required could be carried out, and if it is going to be possible to use the component in the system with success.

- interoperativity: likeness of the functional and technical characteristics of the component with the new system. All the processes of adjustment aim a risk of realization, since it is not safety that they could achieve these adaptations; the engineer should estimate the grade of difficulty that carries implicit.
- applications using the component: it is a guarantee of success that the component had been adapted in projects with similar operative characteristics.

Capability of the development team

If the personnel of development has a scarce knowledge on the operation of the component, the engineer should evaluate the difficulty that will suppose learn to manage it and modify it in order to compose it in the new system. Account should be taken of whether there are good help methods for learning and developing the product with the selected component; account should be taken of:

- methodology of construction of the component: if the team of development does not know the formalisms followed, with methodology not standardized, with wrong specifications, etc., it could carry to that the risk of learning is high.
- documentation quality: satisfactory manuals will settle doubts quickly and provide the best solutions.
- courses and on-line help: targeting learning of the product for adaptation and use of the SC.

8 CONCLUSIONS AND FUTURE WORKS

In view of the want of guidelines for assessing the adaptation of SC for reuse in software projects, we have drawn up a framework of guidelines that can be an aid to project managers for considering the characteristics of the components that will have an impact on the factors that will determine their selection.

The assessment of each characteristic and each factor will be linked to each project and each development company; this assessment will depend on the subjective judgement of the person who examines the SC. However, we suggest that the proposed schema be followed to achieve a good selection. Future research work could target the definition of the particular characteristics of different SCs, assessment of the importance of decision-making factors depending on the characteristics of the SC and their application, design of metrics to quantify the fitness of the reusable components in software projects and the construction of tools for guiding the assessment of the SC as opposed to a particular project.

ACKNOWLEDGEMENTS

We would like to thank Mr Francisco Javier Prieto Rodríguez, MECASER, S.A. project manager, for his valuable contribution to this paper.

REFERENCES

1. Basili V., Briand L. and Thomas W. “Domain Analysis for the Reuse of Software Development Experiences”, Proceedings 19th Annual Software Engineering

Workshop, NASA/GSFC, Greenbelt, MD, December 1994.

2. Boehm B., “Risk Management”, IEEE Computer Society Press, 1989.
3. Caldiera G. and Basili V. "Identifying and Qualifying Reusable Software Components", IEEE Computer, vol.24, n.2, February 1991, pp. 61-70.
4. Charette R. “Software Engineering Risk Analysis and Management” McGraw-Hill 1989.
5. Christensen S. “Software Reuse Initiatives at Lockhead”, Crosstalk, vol.8, n.5, May 1995, pp. 26-31.
6. Fenton N. and Pfleeger S. “Software Metrics: A Rigorous & Practical Approach”, 2nd edition, Intern. Thomson Computer Press, 1996.
7. Gómez-Pérez A. “Knowledge Sharing and Reuse, The Handbook of Applied Expert Systems”, Ed. J. Liebowitz, CRC Press, 1998.
8. Grady R. and Caswell D. “Software Metrics: Establishing a Company-Wide Program”, Prentice-Hall 1987.
9. Hooper J. and Chester R. “Software Reuse: Guidelines and Methods”, Plenum Press, 1991.
10. Jones C. “The Economics of Object-Oriented Software”, American Programmer, vol.7, n.10, October 1994, pp. 28-35.
11. Kan S.H. “Metrics and Models in Software Quality Engineering” Ed. Addison Wesley 1995.
12. McCall J., Richard P, and Walters G. “Factors in Software Quality” NTIS AD-A049-014,015,055 November 1977.
13. Meyer B. “Object-Oriented Software Construction”, Prentice-Hall, 1988.
14. Musa J., Iannini A. and Okumoto K. “Engineering and Managing Software with Reliability Measure”, McGraw-Hill 1987.
15. Poulin J.S. “Measuring Software Reuse: Principles, Practices, and Economic Models” Ed. Addison Wesley 1997.
16. Pressman R.S. “Software Engineering: A Practitioner’s Approach”, Ed. McGraw-Hill, Inc. 1997.
17. Prieto-Díaz R. and Fridman P. “Classifying Software for Reusability”, IEEE Software, January 1987, pp. 6-16.
18. Putnam L. and Myers W. “Measures for Excellence”, Yourdon Press, 1992.
19. Riggs J. “Production Systems Planning, Analysis and Control”, 3rd edition,. Wiley 1981.