On Some Critical Issues in Component Selection in Component based Software Development

Jeetendra Pande
Dept of CS & IT
Uttarakhand Open University,
Haldwani-263139, India

ABSTRACT
Most software companies have increased their inclination towards Component Based Software Development (CBSD) due to the benefits it provides, like reduced development cost and less time-to-market. Moreover, quality of the product also increases. A component is primarily selected based on the functionality it provides, along with other important factors such as the value of quality attributes like functionality, security, maintainability, cost etc. There are many potential candidate components that provide the same functionality as desired by the target application for which the software is to be developed. The most crucial task for the developers/ integrators is to select the best matching component from COTS-library which satisfies all the functional requirements, without compromising on the quality of the overall product and at minimum cost. The current work aims to highlight the research gap in the component selection process, after conducting a detailed survey of the literature of the current component selection techniques available and provide recommendation(s) for a new component selection framework.

General Terms
Software Engineering: Component Based Software Development

Keywords
Component Based Software Development (CBSD), component selection framework, optimization, commercial-off-the-shelf (COTS), cost.

1. INTRODUCTION
Software being at the core of computer's – and hence Information Technology's – efficacy, software technology has been evolving at a frenetic pace. With this unprecedented growth in the extent and variety of software has emerged the need for ensuring cost effectiveness of software. Component Based approach comes as a saviour in which, rather than developing the code from the scratch, already developed and tested code modules (components) are integrated to develop the software product.

A software component is defined as a unit of composition with contractually specified interfaces and explicit context dependencies[1]. Component Based Software Development (CBSD) is the process of assembling existing software components in an application such that they interact to satisfy a predefined functionality[2]. In CBSD, software systems are built as an assembly of components already developed and prepared for integration.

The many advantages of CBSD approach include effective management of complexity, reduced time-to-market, increased productivity, greater degree of consistency and wider usability. It is generally assumed that reuse of existing software will enhance the reliability of a new software application. This concept is almost universally accepted because of the obvious fact that a product will work properly if it has already worked before. Some reusability guidelines include ease of understanding, functional completeness, reliability, good error & exception handling, information hiding, high cohesion and low coupling, portability and modularity[2].

The main goal of component based software engineering is to reuse already developed software code to develop cost effective software, which requires less development time and reduces time-to-market, without compromising on the quality of the software. Reuse of the time tested software component is likely to lead to better productivity and quality. Components for development of software products can be selected from in-house developed component repository or could be purchased from commercial-off-the-shelf (COTS) vendors.

Component selection is one of the most crucial steps in component based software development and success of the final system largely depends on the component selection process. In the current paper an attempt has been made to perform a rigorous review of the component selection literature, find research gaps in the prevalent component selection practices and provide recommendations for an automated, optimal component selection framework.

2. LITERATURE SURVEY
Performing a good COTS selection plays a critical role in the success of the final system[3]. Researchers have proposed various techniques for the component selection problem. Kontio[4] proposed Off-The-Shelf Option(OTSO) approach for component selection followed by modified OTSO[5] wherein the process of defining the evaluation criteria if addresses in depth.

The major phases of OTSO are:

a. Searching- The selection of the COTS components is based on requirement specification, design specification, project plan and organizational characteristics. A component which satisfies most of the functional requirements and system constraints is selected among the available candidate components.

b. Screening- This is an elimination phase where the components which satisfies less requirement are eliminated and the components which satisfies the most of the requirements are passed on the evaluation phase.

c. Evaluation- The component is judged against the functional requirements, correctness, software architecture and business concerns. The benefit derived by using the component is compared against the cost of the component.

d. Analysis- A COTS component is analyzed using Analytical Hierarchy Process and finally a component is selected.

e. Deployment- the selected component is integrated to the system.

f. Assessment- Once the component is deployed, the success of the selection process is assessed as a feedback for future uses.

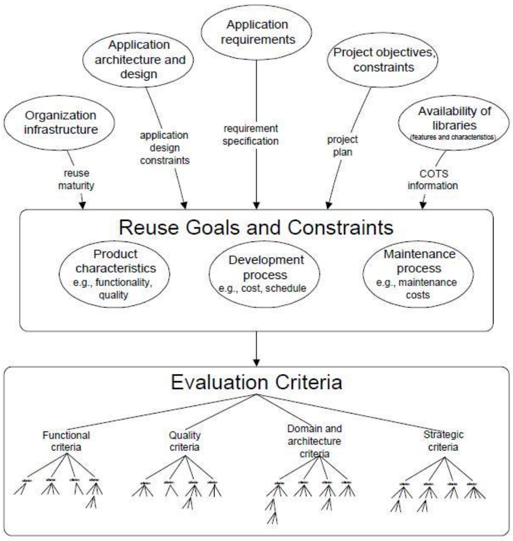

Fig 1: Factors, reuse goals and evaluation criteria

OTSO is one of the first approach for component selection and many of the models are derived from this model.

Morisio & Tsoukis[6] proposed to address the quality requirement during the evaluation process to formalize the component selection process. They proposed IusWare(IUStitia softWARis) approach, which is based on Multi-criteria Decision Aid (MCDA). The major phases of IusWare are:

a. Design of evaluation model- In this phase the actor relevant to the evaluation, objective, role along with the available resources are identified. The components are evaluated either by ranked them in order from highest to lowest or partitioning them into two sets, best and the remaining one; or by formal description of the products. Then corresponding to the quality attributes, hierarchy of the evaluation attributes is defined along with the measure for the evaluation attributes. Analytical Hierarchy Process(AHP) or Weighted Sum Model(WSM) is chosen for aggregation.

b. The evaluation model defined in (a) is applied for component selection.

Fox, Lantner & Marcom Proposed IIDA(The Infrastructure Incremental Development Approach)[7] . This process is based on the combination of two models; waterfall model and spiral model. It has two phases:

a. Analysis prototype- The COTS products are categorized as a family of components and the components which provide the similar functionality are placed in same family. In this phase, the possible candidate COTS component is identified and selected for the family.

b. Design Prototype- After evaluating a COTS component based on the functionality and the performance of a COTS component, the best COTS component is selected for integration.

This approach address many of the challenges associated with building systems that contain large amounts of commercial off-the-shelf (COTS) software.

Lichota, Vesprini & Swanson proposed a generic component architecture better known as PRISM (Portable, Reusable, Integrated, Software Modules) approach which can be used for COTS evaluation[8]. It consists of two parts:

i. Generic component architecture

ii. Product evaluation Process

The important phases of PRISM are:

a. Identification: an initial criteria is set and based on the this criteria, the component that fit-into the criteria are identified.

b. Screening: The components obtained in the first phase are shortlisted based on the criteria of best fitness for further examination.

c. Stand-alone test: shortlisted components obtained from (b) are evaluated against the quality characteristic like reusability, reliability and system requirements.

d. Integration test: in this phase, an estimate of how readily the product can be integrated to the overall architecture is made.

e. Field test: in this phase, the component is re-evaluated after deployment.

This model also provide guidelines for buy versus build

decision.

Tran & Liu[9] proposed a model known as CISD (COTS-based Integrated Systems Development) which is used to select multiple homogeneous COTS products. The three phases of CISD model are:

a. Identification-it consists of two sub-phases

   i. Product classification- information on potential COTS candidates is collected based on the requirement of each(application) service domain.

   ii. Product prioritization- COTS products are screened and prioritize on the basis of interoperability and an ability to fulfil multiple requirements of the service domain.

b. Evaluation- Evaluates the product sets to find the optimal component set, which satisfies most/ all the functionalities required by the system, for integration. The criteria for evaluation are ‘individual functionality’, ‘interoperability’, and ‘performance’.

c. Integration- in this phase necessary product adaptors are build into the selected product set.

The model entails an integration of development –centric and procurement-centric models used to support the development of integrated software systems.

Maiden & Ncube[3] proposed PORE(Procurement-Oriented Requirements Engineering) approach for component selection which suggest that the requirements should be elicited and analyzed at the same time when the COTS products are evaluated.

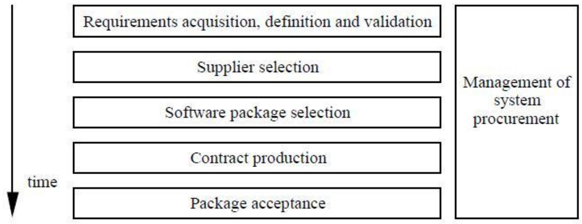

Fig 2: PORE Process Model

It defines three models to achieve a compromise between customer requirements and product features.

a. Requirement model- defines guidelines for acquiring customer requirements using given documents and information.

b. Product model- defines guidelines for acquiring customer requirements from vendor-led demonstration by using test cases.

Compliance model- defines guidelines for acquiring customer requirements from customer-led product exploration by using trial versions.

Gerald et.al.[10] suggested a reverse engineering, which is a process of examining system components and component interrelationships in order to derive a description of a system at a higher level of abstraction, based approach for supporting automated component selection.

Young et.al.[11] proposed a method for component search on the basis of execution of the software components by providing inputs generated systematically and based in this the selection decision of the software component is taken

Vijayan et. al.[12] have proposed a software component identification technique based on domain model and object libraries . The reuse methodology developed by Vijayan have following phases:

a. Derive action and objects from Domain Model Knowledge base

b. Retrieve the corresponding implemented objects from the Object Repository

c. Select the appropriate attributes and methods for the objects based on the new system requirements and perform consistency checking

d. Customize and reconcile the objects for the new system design

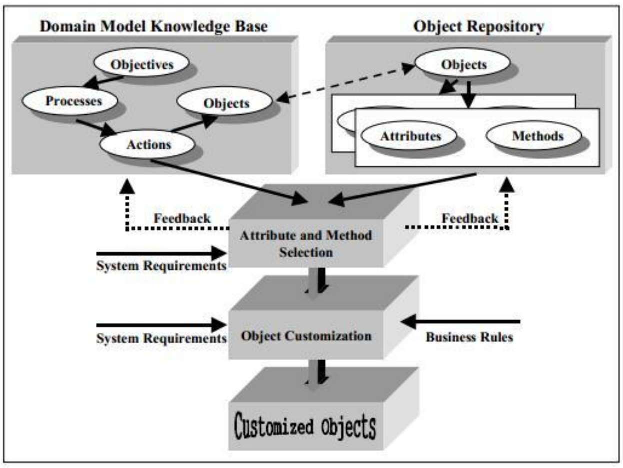

Fig 3: Methodology for reuse in Analysis and Design Stages

The methodology begins with identification of process and higher-level object components, to meet a set of business requirements and maps these to reusable data attribute/ method component to arrive at a set of specific design components.

Kunda & Brooks[13] proposed the Social-Technical Approach to COTS evaluation, better known as STACE approach, in which non-technical issues like social issues, human issues and organizational issues are also given due importance in the component evaluation process.

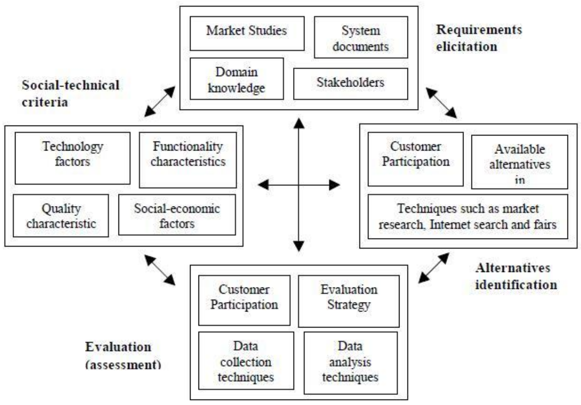

Fig 4: STACE Framework Page Layout

The major steps of STACE are:

a. Requirement elicitation- As the name suggests, this phase is concerned with eliciting high level requirements from various stakeholders of the system, market studies, documents and domain knowledge.

b. Social-technical criteria definition- The requirement obtained from step one is decomposed into measurable attributes.

c. Alternative identification- This phase consists of searching and screening for COTS products/ technology that will be assessed in the evaluation stage.

d. Evaluation- The candidate COTS component are evaluated and ranked according to the social-technical criteria.

STACE supports negotiation between requirement elicitation and COTS evaluation through the involvement of the stakeholders.

Chung et. al.[14] included Non-Functional Requirements while making judgment on COTS components with similar functionality. In the NFR framework, the design decisions and their relation are captured in a goal graph where the nodes are either NFRs or design decisions. The impact of every design decision can be well understood by studying the relationship between NFRs and the intended explicit decisions. The advantage of NFR framework is that it can be reused by other models to address NFR issues.

Ochs et al.[15] emphasized experts' knowledge and suggested COTS Acquisition Process(CAP), which is based on utilizing experts' knowledge for customisable evaluation process. CAP has three phases:

a. Initialization- This phase deals with planning for COTS' acquisition and estimating the cost of acquisition.

b. Execution- This phase is concerned with providing guidance for assessment of COTS products, for which the Analytical Hierarchy Process(AHP) is used.

c. Reuse- In this phase, useful information pertaining to COTS products is saved for future use. This reduces the cost involved in COTS' acquisition process for future systems.

One of the important features of CAP is that it addresses the build versus buy issue of software component.

Comella et.al.[16] proposed PECA(Plan, Establish, Collect and Analyze), which provides the guidelines for customisable COTS selection process. The four main steps of PECA are:

a. Planning the evaluation.

b. Establishing the criteria.

c. Collecting the data.

d. Analyzing the data.

The customisability of PECA makes it applicable in many contexts, for making carefully reasoned and sound product decisions.

Gregor, Hutson & Oresky[17] proposed the Storyboard Approach, which helps the customer to understand their requirements better. Use of use-case and screen-capture during the requirement engineering phase improves the understanding of requirements by the customer. This facilitates customer in selecting more appropriate COTS products, as the requirements are well understood.

Burgues et.al.[18] proposed combined selection approach for COTS components based on distinction at two levels, viz. local level and global level. At the local level, all the individual selection processes of different business areas are located, under the supervision of the process at the global level and at global level the combined selection process takes place, which results in selection of the best combination of the COTS.

Bohem, Port & Yang[19] proposed tree step Win-Win Spiral Model, which uses risk-driven approach to first identify risk, then to analyze risk and finally resolve the risk in an iterative evaluation process.

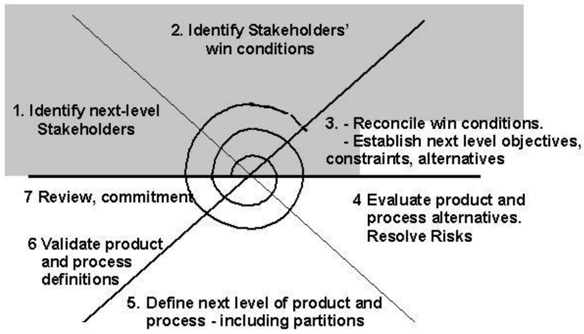

Fig 5: Win-Win Spiral Model

WinWin follows thelowing iterative steps:

a. Identify the stakeholders and their win conditions.
b. Reconcile win conditions and establish next level objectives, constraints and alternatives.
c. Evaluate product and process alternatives and resolve the risk involved.
d. Define next level of product and process, including partitions.
e. Validate
f. Review

Both the customer and the developers are benefited and are in win-win situation, in the sense that the customer wins by getting a product that fulfils the majority of his requirements and the developer wins by working to realistic and achievable budgets and deadlines.

Erol.et.al.[20] proposed a method based on fuzzy set theory, which assists decision-makers in selecting from a finite number of alternatives when there are more than one objectives and both qualitative and quantitative factors must be considered.

Yao et.al.[21] proposed an approach for classification and retrieval of software components, which uses natural language to communicate with the reuse repository. Retrieval of software component is based on semantic matching, components' semantic description and user query semantic representation against the Domain Ontology.

Chung et.al.[22] proposed CARE (COTS-Aware Requirements Engineering), which supports iterative matching, ranking and selection of COTS components. It uses the functional, Non-functional and the architecture of the component.

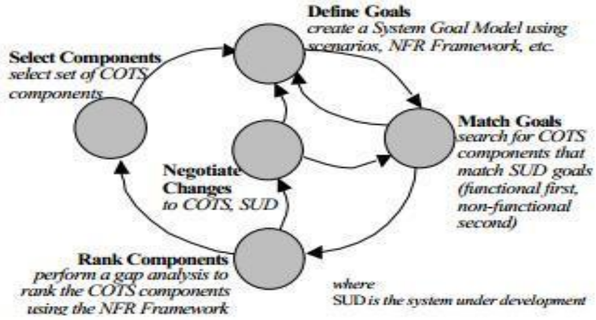

Fig 6: CARE/SA Process

Following are the main steps of CARE:

a. Define Goals- System goals are defined in terms of functional and non-functional requirements.
b. Match Goals- Search for components that match the goal.
c. Rank Components- Components are ranked on the basis of gap analysis.
d. Negotiate Changes- Negotiate changes to COTS components and system goals, in case of mismatch.
e. Select Components- select a set of COTS components that match the goals.

Grau et.al[23] proposed DesCOTS system (Description, evaluation and selection of COTS components), based on use of quality models, to help clients in selection of COTS components. This system was designed to take care of functional as well as non-functional requirements. The architecture of DesCOTS-SD (Strategic Dependency) model describes the elements of a socio-technical system as actors and clarifies the relationships among them (fig.7). DesCOTS describes a detailed method of evaluation criteria and this can be adapted by many domains.

Maxym et.al.[24] suggested using components' description to select and classify the reusable components. A semi-automatic generic method is suggested for component identification and classification, based on generic domain taxonomy and the generated semantic input.

SemaCS consists of four modules:

a. WWW crawler- Scans the web and locates a component, to extract component description.
b. Generic taxonomy- Used to classify components.
c. Query interpreter- Used to interpret queries and extract semantic user input.
d. Cache- Used to store recently evaluated components and user taxonomies.

This approach combines manually generated taxonomy system and query reformulation to allow the system to cut costs and increase accuracy.

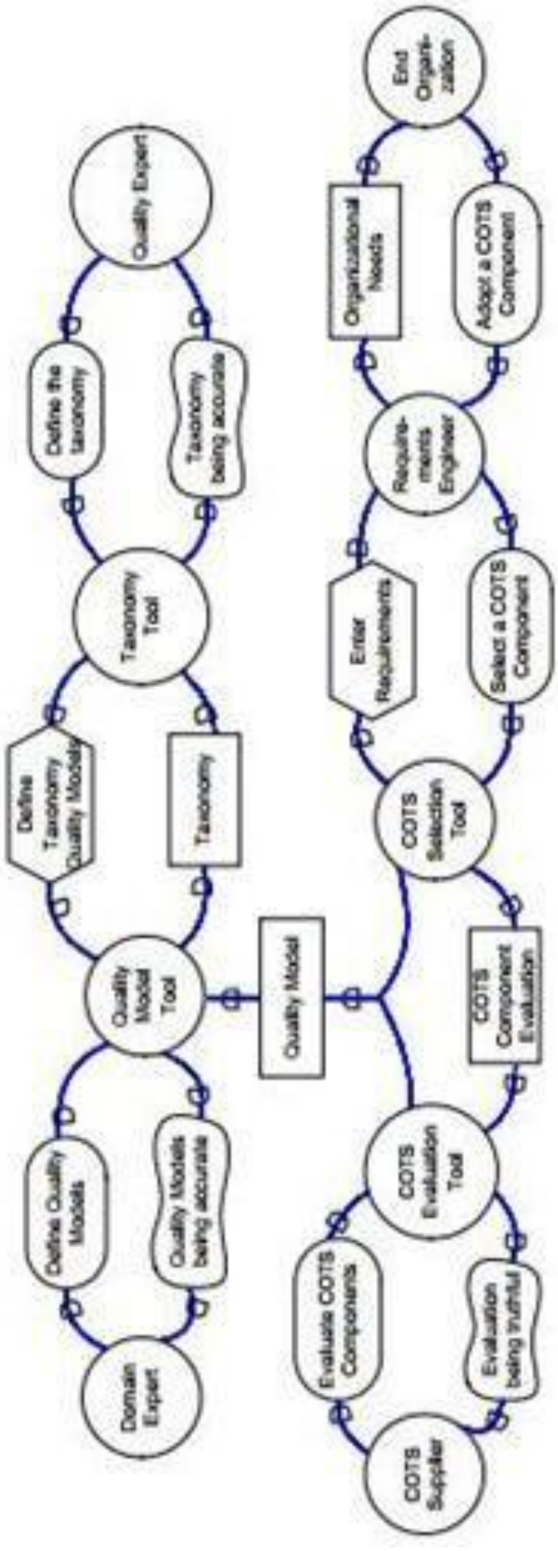

Fig 7: DesCOTS System

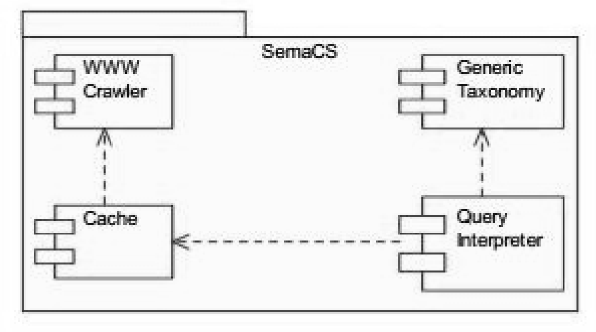

Fig 8: SemaCS

Chor[25] proposed a metric-based approach to assess the conformance of components. Conformance to the requirement is set as a criterion for selection of a COTS component among the other components that provide similar functionality. The features of components related to conformance, along with their sub-characteristics, are identified. Based on this, a practical quality model is proposed to evaluate conformance in components. However, this model lacks theoretical and practical validation.

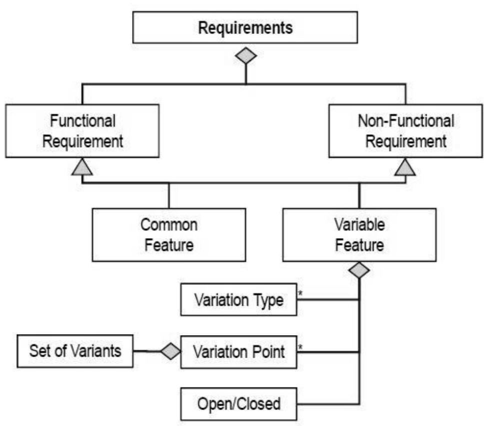

Fig 9: Meta model of family requirements

Mohamed, Ruhe & Eberlein[26] proposed MiHOS (Mismatch Handling for COTS Selection), which addresses the COTS mismatch between system requirements and COTS products, during and after the selection process. This method introduced a method that aims to provide decision support both to selection process and mismatch handling.

The three important phases of MiHOS are:

a. Modelling- This phase is used to identify the mismatch of selected COTS products. Then, these mismatches are assigned relative importance, after which possible techniques for mismatch handling are identified along with the cost of handing the mismatch.

b. Exploration- Optimization techniques are applied to find the optimal or near-optimal solution.

c. Consolidation- Result obtained in phase b is analyzed. The result is accepted if it satisfies the selection criteria. If not, problem parameters are adjusted and the process is restarted. This continues till a satisfactory result is obtained.

MiHOS is very effective in addressing mismatch issues during and after the selection process but does not address the quality requirements of the system.

Kwang et.al.[27] introduced a model for component selection based on high cohesion and loose coupling modules.

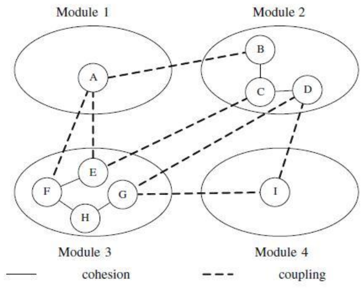

Fig 10: Cohesion and Coupling of modules

Genetic Algorithm (GA) technique is used to solve optimization problem and, based of this, an optimization model for selection of components for Component Based System Development is proposed.

3. CONCLUSION AND FUTURE SCOPE

Most of the above methods are theoretical and are rarely used since, by and large they are very difficult to comprehend by the component integrator. Moreover, most of the above component selection methods are manual and labour intensive; hence it is very impractical to use them for larger systems since doing so increases the component selection cost and the development time. Many of the selection techniques given above make use of approximation techniques, like Analytical Hierarchy process (AHP), which yield approximate results. Therefore, some other method needs to be employed to achieve more accurate results.

Often, during the component selection process we come across a situation when none of the components among the available alternatives satisfies a particular requirement or it is very expensive to implement any requirement using the available components. Also, at times the cost of selection exceeds the cost of the component itself. For these situations, we need some guidelines that help in deciding when to stop searching and instead develop the required component with the said functionality within the organization, using internal resources. Very few of the above selection methods address this issue.

Component selection for a target system should be made keeping in mind quality expectations of the stakeholders. Quality of a system depends on the functional and non-functional characteristics of the system, which tend to vary from one system to other. Therefore, it is imperative to develop a component selection framework that can:

a. Automate the component selection process.
b. Optimize the component selection process.
c. Be scaled up for large systems.
d. Be understood and implemented easily.
e. Employ exact methods for component selection.
f. Address the build versus buy issue.
g. Select components keeping in mind quality expectations of the stakeholders.

It is very important to remember that the main motive behind reusing an already existing software component is that it decreases cost of development without compromising on product quality. The main issue that needs to be resolved is that it is not easy to determine the desired quality attributes of the final software system, such as its performance, scalability or reliability, since the available quality projection tools are not mature enough. To fill this critical gap, this work attempts to identify and delineate the aspects that must be worked upon to provide a truly optimal component selection framework.

4. REFERENCES

[1] \ C. Szyperski & C. Pfister, “Component Oriented Programming” WCOP’96 workshop report. Special issue in object-oriented programming, page 127-130,1996

[2] Naseb S. Gill,” Importance of Software Component Characterization For Better Software Reusability” , ACM SIGSOFT, Software Engineering Notes, Vol. 31, Number 1, Jan 2006

[3] N.A.Maiden & C.Ncube, "Acquiring COTS Software Selection Requirements ,"IEEE Software, vol.15(2).pp. 46-56,1998

[4] J. Kontio, "OTSO: A Systematic Process for Reusable Software Component Selection," University of Maryland, Maryland, CSTR- 3478, December 1995

[5] J. Kontio, G. Caldiera, and V. R. Basili, "Defining factors, goals and criteria for reusable component evaluation," in CASCON'96, Toronto, Ontario, Canada: IBM Press, 1996

[6] M. Morisio and A. Tsoukis, "IusWare: a methodology for the evaluation and selection of software products," IEE Software Engineering vol. 144 (3), June 1997

[7] Greg Fox, Karen Lantner, and Steven Marcom.” A Software Development Process for COTS-based Information System Infrastructure, ” In *Proceedings of the 5th International Symposium on Assessment of Software Tools (SAST '97)*, IEEE Computer Society, pp.133,1997

[8] R. W. Lichota, R. L. Vesprini, and B. Swanson, "PRISM: Product Examination Process for Component Based Development," SAST '97, pp. 61-69,1997

[9] V.Tran and D.Liu,"A Procurement-centric Model for Engineering Component-based Software Systems," SAST '97,pp.70-79.1997

[10] Gerald C. Gannod, Yonghao Chen, and Betty H. C. Cheng , “An Automated Approach for Supporting Software Reuse via Reverse Engineering”, 13th IEEE International Conference on Automated Software Engineering (ASE'98), 1998

[11] Tomas Jell (Ed.), Component-Based Software Engineering (Managing Object Technology Series , (Number 10), SIGS Books/Cambridge Press, June 1998

[12] Vijayan Sugumaran, Mohan Tanniru and Veda C.Storey, “Identifying software components from process requirements using domain model and object libraries”, ICIS '99 Proceedings of the 20th international conference on Information Systems, pp 65-81, 1999

[13] D. Kunda & L.Brooks, "Applying social technical approach for COTS selection" , UKAIS'99, Univ of York, McGraw Hill, 1999

[14] L.Chung, B.A.Nixon, E.Yu, and J.Mylopoulos, Non-Functional Requirements in Software Engineering: Kluwer Academic,1999

[15] M.Ochs, D.Pfahl, G.Chrobok-Diening, and B. Nothhelfer-Kolb, "A COTS Acquisition Process: Definition and Application Experience," ESCOM-SCOPE'00, Munich, Germany, 2000

[16] S. Comella-Dorda, J. C. Dean, E. Morris, and P. Oberndorf, "A Process for COTS Software Product Evaluation," in ICCBSS'02, Orlando, Florida, pp. 86 – 96,2002

[17] S.Gregor, J.Hutson, & C.Oresky, "Storyboard Process to Assist in Requirements Verification and Adaptation to Capabilities Inherent in COTS," in ICCBSS'02, Florida, pp. 132-141,2002

[18] X. Burgues, C. Estay, X. Franch, J. A. Pastor, and C. Quer, " Combined Selection of COTS Components," in ICCBSS'02, Orlando, Florida, pp. 54-64,2002

[19] B.Boehm, D.Port, and Y.Yang,"WinWin Spiral Approach to Developing COTS-Based Applications,"EDSER- 5,Oregon,2003

[20] I. Erol and W. G. Ferrell-Jr., "A methodology for selection problems with multiple, conflicting objectives and both qualitative and quantitative criteria " International Journal of Production Economics vol. 86 (3), pp. 187-199,2003

[21] Haining Yao , Letha Etzkorn, “Towards A Semantic-based Approach for Software Reusable Component Classification and Retrieval”, ACM Southeast Regional Conference, pp 110 – 115, 2004

[22] L. Chung and K. Cooper, "Matching, Ranking, and Selecting COTS Components: A COTS-Aware Requirements Engineering Approach," in MPEC'04, Scotland, UK, 2004

[23] G. Grau, J. P. Carvallo, X. Franch, and C. Quer, "DesCOTS: A Software System for Selecting COTS Components," in EUROMICRO'04, Rennes, France, pp. 118-126,2004

[24] Maxym Sjachyn, Ljerka Beus-Dukic, "Semantic Component Selection - SemaCS," iccbss, pp.83-89, Fifth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems (ICCBSS'06), 2006

[25] Il Woo Chor, “A metric –based Approach to measure conformance in components”, Pacific Science Review, Vol. 9, no. 1, pp.67-71, 2007

[26] A. Mohamed, G. Ruhe, and A. Eberlein, "Decision Support for Handling Mismatches between COTS Products and System Requirements," in ICCBSS'07, Banff, Canada, 2007

[27] C.K. Kwang et.al. “Optimization of Software Component Selection for Component-Based Software System Development,”,Journal of Computers & Industrial Engineering, Elsevier, pp 618-624,2010