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’.

45
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.

![PORE Process Model](image)

**Fig 2: PORE Process Model**

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


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 cases.

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

![Methodology for reuse in Analysis and Design Stages](image)

**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.

![STACE Framework Page Layout](image)

**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.

WinWin follows the following 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.
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
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:

- Automate the component selection process.
- Optimize the component selection process.
- Be understood and implemented easily.
- Employ exact methods for component selection.
- Address the build versus buy issue.
- 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