Commercial-Off-The-Shelf Software Development Framework

Basem Suleiman¹,²,(*)

¹ School of Computer Science and Engineering, University of New South Wales, Australia
² Managing Complexity – Sydney ATP Group, (National ICT Australia)∗, Australia
b.suleiman@acsmail.net.au

Abstract

Budget and schedule savings are the driving factors for the adoption of Commercial-Off-The-Shelf (COTS) software components by software development organizations. The reliance on COTS components has lead to Component-Based Development (CBD) software systems and introduced changes to the software development process and hence software project management responsibilities and roles. This paper introduces a general framework discussion of essential management aspects for CBD, focusing on COTS. Particularly, stakeholders, requirements, component selection and architecture management issues are discussed from different angles. Some CBD management guidelines and best practices for these aspects are outlined in the conclusion. In addition, CBD management challenges are drawn along with some suggestions in the conclusion section.

1. Introduction and motivation

Building software projects from scratch is a very complex process that requires high multi-dimensional management skills [10]. Competition in software market is increasing rapidly along with the emergence of new technologies and requirements. Furthermore, timely development or acquisition quality software has become crucial for maximizing business value of many corporations. As a response to these evolving challenges, adoption of Commercial-Off-The-Shelf (COTS) components have been growing rapidly as an emerging paradigm in software development. By COTS components, we mean commercial software packages with common purposes that are ready to be used in software development and application integration. The driving benefit of COTS-based software development is the potential for massive savings in project’s cost and time [7, 8].

Component-Based Development (CBD) is the process of building software application by relaying on COTS software components. It is a systematic approach that would replace the traditional Software Development Life Cycle (SDLC). Note that apart from COTS, other components, such as open-source components and libraries, can be used during CBD. The emergence of CBD has required essential adaptations and changes in development processes and best practices of traditional SDLC. Activities, tasks, roles and deliverables are not the same in CBD. For example, selecting the most appropriate components according to organization’s requirements could be a new process in CBD. Similarly, activities and best practices of requirement analysis phase should be changed to analysing candidate components and their suitability to project requirements. Such changes require an adaptation of management techniques, strategies and methodologies. Project managers need to consider new management challenges that could arise during and after CBD. In the first instance, it might seem that management responsibilities could become easier as software development will become an assembling process rather than creating everything from scratch. However, the reality is completely contradicting such thoughts, as new issues, elements and considerations should be taken into account when managing CBD. These new elements may include (but are not limited to) system context or environment, component product providers, software marketplace and

---

*This work was done during my Master studies at the University of Western Sydney (UWS). I am now student at NICTA and at the University of New South Wales (UNSW).

* NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.
technology trends, COTS components conformity to system infrastructure and requirements, availability of mature expertise and many additional factors.

We argue that CBD demands new management activities, processes and best practices to be considered and applied for achieving quality within planned time and cost. In this paper, a comprehensive discussion and guidelines are introduced as set of best practices for managing different aspects in CBD, focusing on COTS. Apart from COTS, other components, such as open-source components and libraries, can be used during CBD. A few issues (e.g., those related to licensing) for these components could be somewhat different than for COTS. However, the majority of issues are significantly similar. Since the body of literature for COTS is much bigger and more mature, we decided to focus this paper on COTS.

The paper addresses the most essential management aspects (as we perceive them according to the literature) in the CBD process. Since risk management concerns crosscut several of these management aspects, it has been discussed across these management aspects. Further, these aspects are discussed in terms of project roles, activities and tasks, as well as desired outcomes and deliverables. The guidelines summary indicates the suggested best practices, roles, and outcomes/deliverables. While these guidelines and best practices provide a general framework for managing component-based software, they need to be empirically validated and verified.

Some related work in the literature discussed similar aspects of COTS-based systems, which are introduced in this paper, such as: maintainability issues [2, 9, 11], requirement engineering and COTS selection [4, 5, 16], components architecture and interoperability [13] and component software development challenges [7]. Compared with this related work, the original contribution of this paper is a general framework for managing such aspects in CBD, focusing on COTS. Specifically, a set of management guidelines and best practices for COTS component software development (as we perceive them) are discussed in terms of system stakeholders, requirements, components selection and architecture. This work should be further extended to introduce more specialized methodologies for managing COTS software development because of the increase use of COTS components in software development industry.

The paper structure is as the following. Stakeholders, requirements, components selection and architecture management aspects of COTS software development are discussed in section 2, 3, 4, and 5 respectively. Conclusions and future work are introduced in section 6.

2. Stakeholders management

Managing expectation of systems’ end users plays a central role in their acceptance of the new system [16]. COTS software development is less neutral than custom ones and it is usually hard to meet all end users’ needs and even to tailor COTS components’ feature to their exact requirements. Therefore, it is better to tune end users’ expectations to the reality of CBD process without losing the identity of their system requirements and features. Convincing end users with such issues is essential practice for managing COTS software development. Therefore, business processes and operations should be reengineered to take the advantage of time-to-market technology. The end users will be more satisfied with the valuable cost and time saving of adopting these COTS components along with time-to-market quality achievement which will put their organization on the edge of technology. Once end users’ expectations are managed and adapted with COTS software development, they keen to effectively participate in building quality system through critical input to the software requirement and feedback. This will also return with good gain in maintaining the software [9, 11]. End user acceptance for COTS technology is important to reduce their resistance to change as they understand the need for this technology and they do not feel losing their privileges because of using them. This in turn, will enhance their feedback productivity and therefore software development quality.

COTS software providers are another important stakeholder in COTS development. Specifically, their level of maturity, technical and customer support, quality of technology tools and standards they use for developing their software components and many other criteria. Such criteria need to be planned well and agreed with COTS providers to verify compatibility of COTS providers’ standards with your project standards. COTS providers play a key role in software development because they participate in developing, through their software components, basic parts of your software. Aspects of managing COTS providers are further explained in section 4.

3. Requirements management

System requirements are the most essential artifacts in software development because they
determine and derive the majority of development activities and tasks. They always change and grow during the entire SDLC [6]. In custom-based systems, managing requirements change is one of the best practices that project manager should deal with. In COTS software development, requirement management should be planned as early as possible in SDLC [5] to gain real benefits of COTS components. Requirement change management is a complex task that requires multi dimension skills during the COTS software development since selected components would have some restrictions on the scope and the growth of requirements. However, well-understood and established requirements will help considerably managing the evaluation and selection of COTS products. Once basic systems' requirements are clearly established and analyzed, it is good practice to keep them as one of the driving factors for selecting COTS components. This also will require discussing systems’ requirements with end users in terms of selected COTS because this will help in estimating their satisfaction and early acceptance of system capabilities and reducing requirements change evolution. For example, if you find any type of non-satisfaction, it would be a good practice to negotiate with them and explain the deriving reasons for adopting COTS and their positive and negative influences. For instance, it will be more realistic to clarify that although COTS components have common requirements which do not fit to exact custom requirements but, they are built for specific domain of business processes and they can bring timely IT services and upgrades to the organization. Another reason for the need to manage requirement change in early stages of COTS software development is the high cost, in terms of time and budget, which will arise after selecting and integrating specific COTS components into system. Therefore, it will be a good practice to specify systems’ requirements in a flexible way and accommodate them to the most appropriate COTS components. A trade-off between system requirements and available COTS will help planning and managing requirements change and evolution.

Although requirement change is normally planned and managed early in CBD but, this, sometimes, would not guarantee fixed requirements during CBD. Therefore, plan to specify requirements in a loosely manner [5] so that changes that might be requested during the development process can be accommodated to the selected COTS and to the system infrastructure. In addition, when evaluating and selecting COTS components consider some criteria for COTS coupling with other components and system’s environment and infrastructure. On the other hand, it will be good practice to convince end users in the impact of any change in requirement after the selection of COTS components in terms of quality, cost and time. This will enforce end users to clearly specify their system requirements to avoid any extra cost. They also will focus on basic requirements that the system “must have” and avoid some other requirements that the system “nice to have”. Further, they will understand the fact that although COTS-based systems are unable to absorb many requirements changes like custom-based ones but, they will allow time-to-market quality IT services.

4. Component selection management

Choosing the most appropriate quality components will be a contributing factor in building the quality software. Current trends in choosing software COTS may include, but not limited to, component features demos, Web search and ranking scheme, trade journals and magazines, third party test labs and verification and validation of quality and functionality of COTS products [4, 5]. The evaluation and selection process should be rigorously planned and controlled because it is an early-significant activity in CBD that if it is inaccurately planned will result in high risk in many aspects such as budget, schedule and requirement satisfaction. Further, it should be built on a well planned platform that is based on real and measurable facts. It may also depend on the software system or service being developed. In addition, it should at least consider basic criteria such as: component’s quality and functionality, quality of components’ providers, components conformity to software requirements, compatibility of COTS components with software environment, availability of development technology of COTS components, long –term and short-term vendor support and/or many other factors.

Quality and functionality of COTS components may vary from software to another due to many reasons. For example, when considering quality you may need to pay attention to user satisfaction, conformance to standards or set of predefined quality criteria standards such as; security, performance and reliability [6, 15]. Furthermore, considering special quality measurements and metrics or third-party test laboratories will enhance the selection process. Metrics and measurements are considered one of the most powerful techniques [15] used by project managers for decision making in CBD. More importantly, evaluation of COTS components quality should be performed in the context of system environment because some components show high
quality when they tested separately but, when they are to be integrated within desired software infrastructure they do not. Therefore, prototyping may be useful technique to evaluate and select quality components. This will help in selecting quality components from end user perspective (customer satisfaction).

When evaluating and selecting COTS components it will practical to involve system integrators, interface developers and/or any other technical development members in this process because they will predict other quality criteria from their technical perspective [1, 13]. For example, they may ensure the selection of particular components could lead to a poor performance as they require intensive interface code and communication messages while end users find that these components satisfy their needs. Experience and familiarity of development team with used technologies of selected components may play a critical role as well. If system developers are not so mature in the technology field of selected components then you may expect poor productivity in the integration process and/or time and budget consumption. In such situations, it is a good practice to make trade-off among such factors.

Functionality aspects, on the other hand, play a key role in the evaluation and selection process. Functionality is the basic business operations and functions that comprise a system. Its focus is to meet system requirements [4]. Therefore, you may need to focus on end user perspective by involving them in finding the closest functional components to system requirements because they can verify and validate desired functionality. To meet system requirements, selected COTS components should have most of the required system functionality. This will reduce maintenance during and after CBD. This will also avoid reliance on documentation of COTS components to investigate their functionalities. Instead, it will be better to test these functionalities closely with the system end users using prototyping. Furthermore, set out a list of required system functionalities in the selection and evaluation plan before considering documentation of COTS.

The third criterion for evaluating and selecting COTS components is the maturity COTS providers [14]. Maturity of COTS vendors is a key factor for component’s quality. For example, if components’ vendors normally follow a disciplined and quality process in building their components, then this will ensure quality of building components and in turn they will contribute to the overall quality of software system that use them. This also includes maturity of development process being used to develop COTS components. Other possible vendors’ characteristics that will ensure their maturity when planning for selecting components are: the size and level of establishment of components providers, their experience age in developing selected components, their level of support, especially for old versions of the product, their level of improvement and enhancement for their products along with emerging standards and technologies, their cooperation and interaction with your organization during developing a software system.

Another key criterion for evaluating and selecting COTS components is their conformity to system environment [8]. Conformity means consistency between system requirements, hardware, software, application systems and other elements and COTS components infrastructure. Although it is hard to find exact conformance between system environment and components’ environment but, considering effective criteria will help finding the most appropriate COTS components. These criteria may include elements such as: hardware infrastructure and software application. In addition, include systems’ end users and systems’ integrators in the selection of final components after balancing their needs. End users will confirm the quality of software, and therefore enhance organization’s ROI which will increase the satisfaction of their management. Meanwhile, system integrators will provide you with technical issues that may cause any risk during software development.

System budget and schedule [7, 10] are key elements for evaluating and selecting COTS components. Components that are developed using mature technology and standards do not require expensive development, including integration and maintenance, cost and time because there will be enough expertise, professionals and/or tools in the development market. Meanwhile, COTS components developed with non-mature technology and standards will consume project budget and time because of lack of expertise and technology standards and tools in such domain. In addition, technology expertise and standards, if they are found, might not be mature enough so that software quality will suffer. Therefore, end user’s selection of some components with immature technology needs to be controlled during the evaluation and selection process.

Managing support should have consideration in COTS evaluation and selection process. COTS components that have effective technical and end user support play a key factor in developing and maintaining quality software [2, 9]. For instance, quality COTS components provide effective technical support for integrating and deploying the components in system environment. This will reduce technical
problems and will save project schedule and budget. Meanwhile, end user support for COTS components will help increasing customer satisfaction as different support services are available most of the time and widely spread. These criteria and many others should be considered in the selection process.

Due to its importance, COTS evaluation and selection process should be carefully planned and documented [14]. Documentation artifacts can be used effectively for managing development and maintenance tasks. They should have a clear description about activities and tasks of the process, people involved, evaluation and selection criteria, budget and schedule impacts, set of candidate components and trade-off among them, decision made and rationale for selecting specific components and many others. These artifacts can be used to resolve any problem that could arise during development and maintenance. They also can be reused for other similar project to help evaluating and selecting COTS components. Although COTS components have their own documentation but, they just provide general artifacts instead of specific one to software environment. The more accurate documentation is made about decisions and why such decisions were adopted, the easier is to resolve risks that might emerge during and after development.

Careful planning for COTS selection [4, 5] should depend on particular criteria which are driven from end user requirements and software environment. Although you may follow the same activities and tasks in selecting components for two distinct systems but, selection and evaluation criteria along with their documentation could be completely different.

5. Architecture management

In CBD, the use of COTS component could have an influence on the overall software architecture. COTS components and their own architectural concepts could limit some aspects of software architecture and at the same time the software architecture may constraint the type of COTS products to be used [3, 13]. Plan to explicitly include such types of constraints in the system architecture as well as in its documentation. This will help managing changes that would be predicted during development or when some new versions of used COTS are released. Moreover, by documenting the decisions made for choosing a particular architecture along with its constraints and features, it will be practical to determine the possibility of using new version of COTS components or the addition of new interfaces.

This will be a long term management point of view. Further, this documentation may be used for COTS reuse in building similar software architectures.

Besides the mutual impact of software architecture and COTS components architecture, plan to have a flexible architecture that can absorb changes and upgrades [13]. This should include a well planning, scheduling and executing software structuring tasks to produce a flexible architecture that allow not only the inclusion and exclusion of new product or software features but also the accommodation to unanticipated changes. Such unpredictable problems could be because of the tension between system requirements and COTS products because they are from distinct providers that use different technologies and developments standards. The more flexible the system architecture is built, the easier the system can be evolved with the component market.

6. Conclusions and future work

An increasing number of software applications are being built using COTS products to overcome two main reasons of software project failures: budget and schedule. However, as it has been warned by some COTS researchers and practitioners and also experienced by many COTS-based projects [1, 7, 8] that, it is unrealistic to expect the true advantages of cost and time saving without adoption of a systematic approach towards the use of COTS. Such adoption should facilitate the management process of CBD, which is complex because of diversity of existing COTS products and the way they were developed.

The comprehensive discussion in this paper introduced a high level framework for some essential CBD aspects from management point of view. The original contribution of this paper is a general framework for managing different aspects in COTS software development, focusing on COTS. Specifically, a set of management guidelines and best practices for COTS software development (as we perceive them) are discussed in terms of system stakeholders, requirements, components selection and architecture. Software project management is a challenging responsibility that could lead to the success or failure of software project. Diversity of considerations that have to be taken into account before, during and after software development requires mature experience and skills by project managers. For COTS-based systems, there is variety of challenges which need to be faced to gain the real benefit of COTS components in software development. Such challenges depend, but not limited to, on the following factors (and some others):
- System requirements and available components.
- Software system environment and infrastructure.
- Common software marketplace and component technology trend competition.
- Availability of components, tools and qualified/experienced developers.
- Conformity of components to software engineering standards.

These challenges could be resolved through the adoption of some policies and strategies such as:

- Advancing COTS software development as an engineering discipline approach by software engineering community.
- Adopting a set of quality and development standards for building COTS components by COTS providers to ensure unified component development.
- Introducing new courses in the universities to provide qualification in component-based development and management aspects.
- Adopting a set of practices and disciplined processes for development and management of COTS components based on market needs.

The suggested guidelines and best practices require empirical validation through a set of proved projects which can be gained from components’ providers, development organizations and software consumers. This should be done in future work. Another issue for future work is to research which additions and modifications to these guidelines and best practices should be added for components that are not COTS (e.g., open-source components).

Acknowledgement

I would like to acknowledge Bhuvan Unhelkar for the valuable information and knowledge he presented in IT Project Management unit at the University of Western Sydney. Vladimir Tosic from National ICT Australia also provided valuable feedback and suggestions for some sections.

References


