

SoCoEMo-COTS: A Software Economic Model for Commercial Off-The-Shelf (COTS) Based Software Development

Sana Ben Abdallah Ben Lamine, Lamia Labeled Jilani, Henda Hajjami Ben Ghezala
Laboratoire Riadi-Gdl, Ecole Nationale des Sciences de l'Informatique
Campus Universitaire la Manouba, La Manouba, 2010, Tunisia

Abstract - With component-based reuse, software development is achieved through the planned integration of pre-existing software components. Commercial-off-the-shelf (COTS) Based Development (CBD) is one of the systematic reuse approaches promising gains in cost, operational quality, functionality, time to market and maintenance overheads. This is an increasingly popular paradigm for software development, but one that is not without risks and associated costs. In fact, an increasing number and variety of COTS components become available but it is important to understand the costs, benefits, and risks entailed in using these components. This paper presents an economic model for any organization willing to adopt a CBD approach for its systems development. Such model which quantifies the predicted benefits and return on investment of CBD can help managers make the good decision to adopt or not such a reuse approach. We denote our model SoCoEMo-COTS for "Software Cost Estimation Model for COTS".

Keywords: COTS, cost estimation model, investment cycles.

1. Introduction

COTS based software development is the process of building software applications from commercially available software components. In this work, we adopt the definition of a COTS component which is given in [1]. A COTS product is an executable software product that has the following characteristics:

- It is sold, leased, or licensed to the general public.
- Buyers, lessees, and licensees have no access to the source code; hence can only use the product as a black box.

- It is offered by a vendor who has created it and is typically responsible for its maintenance and its upgrades.

- It is available in multiple identical copies (within the same version) on the market.

There are many advantages in using component-based software reuse or simply componentware [2]. The use of COTS products shows particular advantages such as gains in cost, operational quality, functionality, time to market and maintenance overheads.

However, there are risks as well; most stemming from the fact that there is much that is out of the control of the COTS acquirer, integrator, and user. It is the vendor and not the stakeholders who control the component's functionality, its performance, and its evolution. Hence, "COTS is high risk because we are dependent on someone else". In this paper, we propose an economic model which quantifies the predicted benefits and return on investment of CBD that can help managers make the good decision to adopt or not such a reuse approach. We denote our model SoCoEMo-COTS for "Software Cost Estimation Model for COTS". SoCoEMo-COTS model is principally based on the use of the integrated (generic) cost estimation model [3]. Other economic cost models exist for CBD development [4, 5, 6, 7, 8, 9, 10] but a lot of issues are still existing and each model concentrate on certain aspects and neglect others by having its own viewpoints, cost factors and economic functions. Some works concern the maintenance phase costs and risks while others evaluate the development effort and maintenance costs.

Our economical model, SoCoEMo-COTS, considers that the system to be developed may be of two kinds:

1) COTS-Solution Systems: one substantial product (suite) tailored to provide significant system functionality, 100% COTS (common for administrative systems),

2) COTS-Intensive Systems: multiple products from multiple suppliers integrated to collectively provide

system functionality, a mix of COTS and custom code (common for operational, safety-critical systems). This kind of systems have these features : 1) generic solutions; tightly coupled to end user/business processes, 2) vendor maintained, 3) tailoring, parameterization focus, 4) probably more flexible in supporting end user/business processes, 5) project maintained, 6) integration, engineering focus and 7) products/parts are "black boxes". In our model, the COTS Based Development approach is considered as an instance of a CBSE reuse approach where all the reused components are of type COTS. These COTS components can be reused in several applications of a given corporation and so, stored in a reuse library. Initial and prototype systems constructed with COTS elements can often be developed and fielded more quickly than systems built entirely from scratch, but final system testing and post-delivery maintenance and long term sustainment can be highly challenging. We are faced to the problem of COTS components "Volatility" which means the frequency with which vendors release new versions of their products and the significance of the changes in those new versions (i.e., minor upgrades vs. major new releases). When we first deploy our system, we of course have selected a suite of components that will provide the functionality we require while at the same time work in concert with each other. Over time, however, those products will likely evolve in different directions in response to the market place, in some cases even disappearing from the market all together. As a consequence, the ability of these diverging products to continue functioning adequately together if and when we install the newer versions will likely also become more problematic; the more COTS components you have, the more severe the consequences of these increasing incompatibilities will become. In [8], Chris Abts proposes a CBS design rule:

The CBS Functional Density Rule of Thumb [7]: Maximize the amount of functionality in your system provided by COTS components but using as few COTS components as possible.

Corollary to the CBS Functional Density Rule: The absolute number of COTS components in your system should not exceed the maintenance equilibrium value.

The maintenance equilibrium value n as shown in detail in the COTS-LIMO (COTS-Life span Model) [4] gives the maximum number of used COTS components in an application. As long as the number of COTS components in the system is less than n , the increase in experience gained by your system maintainers over time and thus the inherent improvements in their productivity will outpace the increased effort required to maintain the system as the COTS products it contains age and evolve in divergent directions.

However, at some number of installed COTS components n , the breakeven point is surpassed and no matter how skilled and experienced your maintainers become, the increases in their efficiency at maintaining the system can no longer keep pace with the impact of the increasing incompatibilities arising in the evolving COTS components. At this point you have reached the zone in which the useful life of your system has been shortened considerably and a decision to retire the system will soon have to be made.

So, in our model, we suppose that the maintenance equilibrium value is taken into account and more specifically the CBS Functional Density Rule of Thumb applied.

Finally, our model is based on the use of the integrated model which we present in section 3. In previous work, we have used this latter model to build also SoCoEMo-PLE and SoCoEMo-PLC using COTS components as economical models for PLE reuse approach [11, 12]. In fact, we want to use the same concepts to provide an economic model for each of the large scale reuse approaches (PLE, CBSE and CBD) but specializing each of them according to the development life cycle and the specificities of each reuse approach. We also use some formulas coming from COCOTS model [5] and the works of Abts, specifically for the maintenance phase [6].

The paper is organized as follows. In section 2, we present the integrated cost model which is the basic used model that we extend for taking into account CBD. Section 3 details our model. A discussion is presented in section 4 and a conclusion and future work are presented in section 5.

2. The Integrated Cost Estimation Model for Reuse

The integrated cost estimation model for software reuse, presented in [3] is characterized by:

Variety of Investment Cycles. The model defines four distinct investment cycles and identifies how they feed into each other: The corporate engineering cycle, the domain engineering cycle, the component engineering cycle and the application engineering cycle. In fact, there are four parties in the software reuse process: the corporate management, which has a stake in seeing the reuse program reap benefits for the corporation; the domain engineering team, which has a stake in seeing its domain engineering products reused; the application engineering teams, which has a stake in producing applications with low cost, high quality, and short time-to-market; and the component developers, who have a stake in seeing their components reused widely.

Variety of cost factors. The cost factors used to define the various economic functions are quantified for each investment cycle and they include:

- Investment Cycle (Y), in years.
- Start Date of the investment (SD).
- Discount Rate (d), which is an abstract quantity that reflects the time value of money.
- Investment Cost (IC), in person months (PM).
- Periodic Benefits (B(y)), at year y, for $SD+1 \leq y \leq SD+Y$, in PM.
- Periodic Costs (C(y)), at year y, for $SD+1 \leq y \leq SD+Y$, in PM.

Variety of Economic Functions. The economic functions used to assess the worthiness of the investment after the estimation of cost factors are: Net Present Value (NPV), Return on Investment (ROI), Profitability Index (PI), Average Rate of Return (ARR), Average Return on Book Value (ARBV), Internal Rate of Return (IRR), and Payback Value (PB).

Variety of Viewpoints. The model analyzes the cost factors for each stakeholder (corporate managers, domain engineering teams, application engineering teams, and producers of reusable assets).

3. SoCoEMo-COTS

SoCoEMo-COTS is a Software Cost Estimation Model for COTS Based Development. It is based on the strong features of the integrated cost estimation model for reuse and tries to palliate its insufficiency regarding to the COTS Based Development. In fact, the integrated cost estimation model for reuse considers reuse in general. It doesn't consider specifically the COTS development life cycle.

As presented in the introduction, the SoCoEMo-COTS model has a main assumption that all the reusable components used in the development are COTS components. The corporation will not need to develop reusable components internally in the corporation. Thus the component engineering cycle will avoid costs of developing and maintaining reusable components. Here, we can omit the component engineering cycle, since no component engineering activities are conducted.

In this section we detail the SoCoEMo-COTS model's equations for three engineering cycles: the domain engineering cycle, the application engineering cycle and the corporate engineering cycle. For each cycle, estimations are done for three cost factors: IC, C(y), and B(y), since Y, d, and SD are uniform within a corporation. Notational conventions for SoCoEMo-COTS are δ , α , and ρ which denote respectively domain, application and corporate engineering factors.

Cost factors used are Y, d, SD, IC, C(y), and B(y). Economic functions used are NPV, ROI, PI, ARBV, and PB.

3.1. Domain Engineering Cycle

3.1.1. Investment Cost. The investment cost of the domain engineering cycle is estimated by:

$$IC_{\delta} = DA + \sum_{i=1}^N (CCOTS_i + LI_i + IdSel_i) \quad (1)$$

DA is the Domain Analysis Cost which is determined by expert judgment.

$CCOTS_i$ is the Cost of buying a COTS components i in year SD. We suppose that the cost of a COTS component is less then the cost of the same component if it was developed by the component engineering cycle internally in the corporation. In fact, to [1], using COTS components allows gain in cost, because the product is produced once and used multiple times. Then, it can be sold for an arbitrarily small fraction of its development cost.

N_{SD} is the total number of COTS components bought in year SD.

LI_i is the certification and Library Insertion cost of a COTS component i . It is determined by expert judgment.

$IdSel_i$ is the identification and selection (assessment) cost of a COTS component i . It is determined by expert judgment.

3.1.2. Periodic Cost. The periodic cost for the domain engineering cycle is estimated by:

$$C_{\delta}(y) = \sum_{i=1}^{N_y} (CCOTS_i + LI_i + IdSel_i) + \sum_{i=1}^{N_{cots}} (NRC_i + Re ass_i) + OC(y) * payl \quad (2)$$

N_y is the total number of COTS components bought in year y, perhaps new COTS components to be put in the reuse library or components to replace others when the vendors interrupt their activities (disappearing from the market). Note that N_y can be equal to zero if no new COTS are bought in year y.

$CCOTS_i$, LI_i and $IdSel_i$ are costs relative to buying a new COTS component i in year y. The same gain in cost thanks to the use of COTS components (see Investment Cost).

N_{cots} is the total number of COTS components

bought in years before y .

NRC_i is the new release cost of a COTS component in year y . Since the vendor of a COTS component is typically responsible for its maintenance and its upgrades, then the new release cost of a COTS component is given by the vendor. The maintenance cost of a COTS component is less than the maintenance cost of a reusable component developed internally because the maintenance charge is divided on the multiple users of the COTS component.

$Reass_i$ is the cost of reassessment (selection) of COTS component. This cost is determined by expert judgment or by those proposed in the COCOTS model [5].

$OC(y)$ is the operating cost of the library of COTS components in year y . It is determined by expert judgment.

$Payl$ is the average monthly salary of the librarian.

3.1.3. Periodic Benefit. The periodic benefit of the domain engineering cycle is estimated in year y by:

$$B_{\delta}(y) = \sum_{j=1}^{N_{sell}} CCOTS_j \quad (3)$$

$CCOTS_j$ is the Cost of selling a COTS component j to the application engineering cycle in year y . We suppose that the domain engineering cycle sells COTS to the application engineering cycle at the same price it bought it.

N_{sell} is the total number of COTS components sold in year y to application engineers. Note also that $B_{\delta}(SD) = \sum_{i=1}^{N_{SD}} CCOTS_i$ because these N_{SD} COTS components will be sold to application engineering team.

3.2. Application engineering cycle

3.2.1. Investment Cost. The investment cost of the application engineering cycle is estimated by:

$$IC_{\alpha} = C_{\alpha}(SD) = \sum_{i=1}^{N_{\alpha}} (CCOTS_i + Tail_i) + INCOTS + VVC + \sum_{i=1}^n (Eu_i * Payd) \quad (4)$$

N_{α} is the total number of COTS components used in application α .

$CCOTS_i$ is the price of the COTS component i used

in application α .

$Tail_i$ is the tailoring cost of the COTS component i used in application α . This cost is determined by expert judgment or by those proposed in the COCOTS model [5].

$INCOTS$ is the cost of integration of the COTS components used in application α (glue code, wrap code). These costs are determined by expert judgment.

VVC is the verification and validation cost. It is determined by expert judgment.

Eu_i is the custom components (n components) developed in the case of COTS intensive systems. This cost is determined by COCOMO in organic mode. This cost is zero in case of COTS solution systems, $INCOTS$ is also equal to zero for such systems and $NC=1$ because we have just one COTS product.

$Payd$ is the average monthly salary of the developer.

3.2.2. Periodic Cost. It is estimated by:

$$C_{\alpha}(y) = \sum_{i=1}^N (Retail_i) + GlueEvol + VolC + MaintUnique \quad (5)$$

$Retail_i$ is the cost of retailing an eventual new release of a COTS component i . $Retail_i$ is equal to zero if there is not a new release for COTS component i .

$GlueEvol$ is the cost of glue code evolution. This cost is given by the COCOMO maintenance cost.

$VolC$ is the Extra Application Effort due to COTS volatility during development.

All the three above costs are estimated by expert judgment or by COCOTS model.

Note that EXTRAE and COTSR are reduced (perhaps equal to zero) in our model because we adopt the assumption that the maintenance equilibrium value is taken into account and more specifically the CBS Functional Density Rule of Thumb applied (as explained in the introduction).

$MaintUnique$ is the maintenance effort for the unique custom components developed in the application in case of software intensive systems. This cost is given by the COCOMO maintenance cost.

3.2.3. Periodic Benefit. The periodic benefit of an application in year SD is estimated using RCA to quantify cost economies for an application α in year SD .

$$B_{\alpha}(SD) = \sum_{cots \in \alpha} RCA_{cots} \quad (6)$$

RCA_{cots} [13] is reuse cost avoided by the use of a

COTS component in the application α . We suppose that the reuse cost avoided by the use of a COTS component (RCA_{cots}) is greater than the reuse cost avoided by the same component if it was developed internally in the corporation.

The use of COTS components permits gain in cost (the multiple users of a COTS product share the cost of developing the product) and gain in operational quality because the product is widely used by a broad segment of users, then, it is typically thoroughly tested and debugged, hence it has much better quality. In addition, the use of COTS components permits gain in maintenance overhead, because the multiple users of a COTS product not only share the cost of developing the product, but they also share the cost of its long term operation and maintenance. The vendor is typically responsible for its corrective, perfective, and adaptive maintenance.

For years y after SD , we consider that benefits of an application α using the COTS based development come from cost economies achieved by the use of high quality reuse components, pretended to need less maintenance:

$$B\alpha(y) = \sum_{\text{cots} \in \alpha} SCA_{\text{cots}} \quad (7)$$

SCA_{cots} [13] is service cost avoided by the use of a COTS component cots in the application α .

3.3. Corporate engineering cycle

3.3.1. Investment Cost. The investment cost of the corporate engineering cycle is estimated by:

$$IC_{\rho} = INF + \sum_{\delta \in \rho} C_{\delta}(SD) \quad (8)$$

INF is the reuse infrastructure cost (training to the development with COTS components, market watch, administrating COTS licences, etc). It is determined by expert judgment.

$C_{\delta}(SD)$ is the investment cost of the domain engineering cycle δ .

3.3.2. Periodic Cost. The periodic cost in the corporate engineering cycle is estimated by:

$$C_{\rho}(y) = \sum_{\delta \in \rho} C_{\delta}(y) + \sum_{\alpha \in \rho} C_{\alpha}(y) \quad (9)$$

$C_{\delta}(y)$ is the periodic cost of the domain engineering cycle δ in year y .

$C_{\alpha}(y)$ is the periodic cost of the application engineering cycle α in year y .

3.3.3. Periodic Benefit. The periodic benefit of the corporation is given by:

$$B_{\rho}(y) = \sum_{\alpha \in \rho} B_{\alpha}(y) \quad (10)$$

$B\alpha(y)$ is the periodic benefit of application α in year y .

4. Discussion

SoCoEMo-COTS estimates costs and benefits of software development with a COTS-based development approach for three investment cycles (see figure 1). All of them cooperate to ensure their own interest and also the collective one of the corporation that adopts a COTS-based development approach to develop software. Some costs are determined by expert judgment [14].

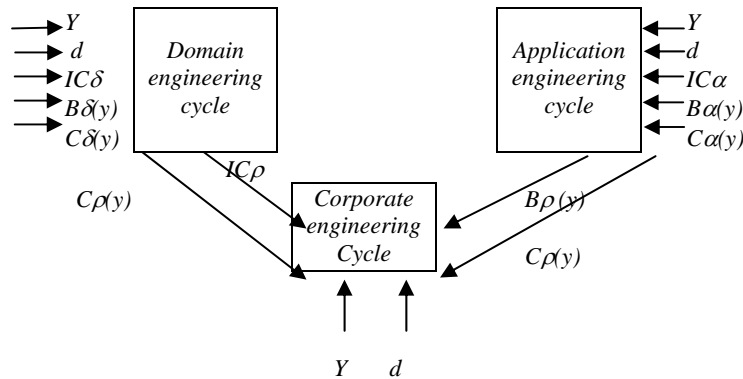


Figure 1. Cost cascade for SoCoEMo-COTS. The different cycles feed costs and benefits to each other

The preliminary experiments of the model (we did not present the example here for constraints of space) have shown the potential benefits of the use of COTS components development approach since these components permit gains in cost, quality and then maintenance costs. The use of maintenance equilibrium value and more specifically the CBS Functional Density Rule of Thumb is fundamental for maintenance concerns with COTS components.

The cost balances of the SoCoEMo-COTS model for the different engineering cycles are given in tables 1 to 3.

Table 1. Cost balance for the domain engineering cycle

<i>Year</i>	<i>Cost $C_{\delta}(y)$</i>	<i>Benefit $B_{\delta}(y)$</i>
$y = SD$	Cost of the domain analysis + Cost of COTS components identification and selection (assessment) + Cost of buying COTS components in year SD + Cost of library insertion	
$y > SD$	Cost of buying COTS components in year y + impact of the COTS vendor's maintenance cost (new release and reassessment of COTS components) + Operational cost of the library of COTS.	Sell of COTS components to application engineering team

Table 2. Cost balance for the application engineering cycle

<i>Year</i>	<i>Cost $C_{\alpha}(y)$</i>	<i>Benefit $B_{\alpha}(y)$</i>
$y = SD$	Cost of buying and tailoring COTS components + Cost of COTS integration (for COTS intensive systems) + cost of verification	Economies on development and maintenance costs through the use of COTS components

	and validation + cost of developing custom components (for COTS intensive systems)	
$y > SD$	Cost of re tailoring COTS components + cost of glue code evolution + cost of Extra Application Effort due to COTS volatility during development. + cost of replacing some COTS components by others if this scenario happen, it is equal to zero otherwise. + cost of maintaining unique components.	Quality gains (maintenance cost economies) through the use of COTS components.

Table 3. Cost balance for the corporate engineering cycle

<i>Year</i>	<i>Cost $C_{\gamma}(y)$</i>	<i>Benefit $B_{\gamma}(y)$</i>
$y = SD$	Infrastructure cost + Domain cost in year SD.	
$y > SD$	Domain periodic costs + Application periodic costs	Application periodic benefits.

5. Conclusion

In this paper we presented a software cost estimation model for COTS Based Development when an organisation develops COTS solution systems and/or COTS intensive systems. In our model, the COTS Based Development approach is considered as an instance of a CBSE reuse approach where all the reused components are of type COTS. These COTS components can be reused in several applications of a given corporation and so, stored in a reuse library. Another assumption considered by our model is that we suppose that the maintenance equilibrium value is taken into account and more specifically the CBS Functional Density Rule of Thumb applied. This model is based on the calculus of costs and benefits for three investment cycles in a corporation: the domain

engineering cycle, the application engineering cycle, and the corporate engineering cycle. It is based on the use of the integrated model that we extend to take into account the development life cycle of CBD and where the component engineering cycle is not necessary because all the reused components (COTS) are bought from a third party. However the model SoCoEMo-COTS has proved potential benefits both theoretically and by the application of the model on an example (we didn't present it here because of lack of space), our future work focuses on detailing much more some cost estimations, especially those determined by expert judgments and applying COCOTS formulas for concrete case studies.

5. References

- [1] Mili, H., Mili, A., Yacoub, S., Edward A.: *Reuse Based Software Engineering: Techniques, Organizations, and Measurement*. John Wiley & Sons, Inc., ISBN: 0-471-39819-5 (2001) 672 p
- [2] Brown A.: *Large-Scale Component-Based Development*, Prentice-Hall, Inc, 2001.
- [3] Mili A., Chmiel S.F., Gottumukkala R and Zhang L.: "Managing Software Reuse Economics: An Integrated ROI-based Model". *CSEE Department West Virginia University*, Morgantown WV 26506-6109 USA (2000)
- [4] Abts, C.: "A Perspective on the Economic Life Span of COTS-based Software Systems: the COTS-LIMO Model", *Proceedings of the COTS Software Systems Workshop held in conjunction with ICSE 2000*, Limerick, Ireland, May 2000.
- [5] Abts, C., Boehm, B. and Bailey Clark, B.: "COCOTS: a COTS software integration cost model", *Proceedings ESCOM-SCOPE 2000 Conference*, Munich, Germany, 2000.
- [6] Abts, C.: "CBS Maintenance Phase Modeling", *USC-SEI-CeBASE Workshop on COTS-Based Systems*, Feb. 7-9, 2001
- [7] Abts, C.: "COTS-Based Systems (CBS) Functional Density—A Heuristic for Better CBS Design", *Proceedings of the First International Conference on COTS-Based Software Systems*, Orlando, FL, February 2002.
- [8] Abts, C.: "COTS-based Systems and Make vs. Buy Decisions: the Emerging Picture", *Position Paper for the International Workshop on Reuse Economics*, Austin, Texas 4.16.2002, April 2002.
- [9] Brooks L.: "Costing COTS Integration", *Third International Conference on COTS Based Software Systems*.
- [10] Staley M., Oberndorf P. and Sledge C.: "Using EVMS with COTS-Based Systems", *CMU/SEI-2002-TR-022*, ESC-TR-2002-022
- [11] Ben Abdallah Ben Lamine S., Labeled Jilani L., Hajjami Ben Ghezala H.: "A Software Cost Estimation Model for Product Line Engineering: SoCoEMo-PLE", *Proceedings, The 2005 International MultiConference in Computer Science and Computer Engineering, Software Engineering Research and Practice conference SERP 2005*. Las Vegas Nevada USA (2005)
- [12] Ben Abdallah Ben Lamine S., Labeled Jilani L., Hajjami Ben Ghezala H.: "Cost Estimation for Product Line Engineering Using COTS Components". *Lecture Notes in Computer Science*, Springer-Verlag GmbH, Volume 3714 / 2005, (2005) 113-123
- [13] Poulin, J. and Caruso, J.: "A Reuse Metrics and Return on Investment Model", *Advances in Software Reuse, Proceedings, the Second international Workshop on Software Reusability*. Lucca, Italy (1993), pp. 152-166.
- [14] Ben abdallah S., Labeled Jilani L. and Ben Ghezala H.: "Importance of Knowledge Engineering in Cost Estimation Models for Softawre Reuse : case of Software estimation Model for PLE", *The Seventeenth International Conference on Software Engineering and Knowledge Engineering, SEKE05*, July 14 to 16, 2005, Taipei, Taiwan, Republic of China