Filtering COTS Components Through an Improvement-Based Process*

Alejandra Cechich[1] and Mario Piattini[2]

[1] Departamento de Ciencias de la Computación,
Universidad Nacional del Comahue, Buenos Aires 1400,
Neuquén, Argentina
acechich@uncoma.edu.ar
[2] Grupo Alarcos, Escuela Superior de Informática,
Universidad de Castilla-La Mancha, Paseo de la Universidad 4,
Ciudad Real, España
Mario.Piattini@uclm.es

Abstract. Typically, COTS evaluations embody a first stage intended to determine rapidly which products are suitable in a target context. This stage – called "filtering" or "screening" – chooses a set of alternatives to be considered for more detailed evaluation. For successful filtering processes, composers increasingly focus on closing the gap between required and offered functionality, hence reducing ambiguity of information for comparison. In this paper, we introduce a filtering process, which is based on early measurement of functional suitability of COTS candidates. Measures are immersed in a Six Sigma-based process aiming at improving the filtering process itself as well as its deliverables.

1 Introduction

The adoption of COTS-based development brings with it many challenges about the identification and finding of candidate components for reuse. The search is generally driven by evaluation criteria defined at different levels or as part of an iterative process, in which the preliminary analysis of the current system is an important source for criteria definition [10].

However, the first part in the identification of suitable COTS candidates is currently carried out dealing with unstructured information on the Web, which makes the evaluation process highly costing when applying complex evaluation criteria. Currently, empirical studies indicate that the necessity of formal processes for evaluation depends on the context, but the results also confirm the necessity of accelerating the identification and filtering of candidates [14,20].

Identification of COTS candidates is a complex activity itself. It implies not only dealing with an impressive number of possible candidates but also with unstructured information that requires a careful analysis. In this context, the proposal in [12] sug-

* This work is partially supported by the CyTED project VII-J-RITOS2, by the UNComa project 04/E059, and by the MAS project (TIC 2003-02737-C02-02).

gests extending the identification stage with a learning phase, which provides support to the COTS component discovery process. As a different and possibly complementary approach, other proposals use description logics to develop an ontology for matching requested and provided components [4,18]. Some other approaches try to measure the semantic distance between required and offered functionality [1,13] but these measures usually need detailed information as input to the calculations.

In addition to learning and classification issues, a filtering process is concerned with the pre-selection of candidates. It actually takes place by matching several properties of COTS components, including some inexact matching. Moreover, there are some cases where goals cannot be entirely satisfied without considerable product adaptation and other cases where these goals must be resigned to match product features [2,11].

As a possible improvement, in [19] the Six Sigma approach has been suggested selecting packaged software; however the evaluation mainly relies on the information provided by demos and additional documentation of the software. Then, the lack of measures makes this process perfectible.

Our Six-Sigma based approach focuses on fact-based decisions and teamwork, which might drive the identification and filtering process by using specific measures [6]. Particularly, we consider functional suitability as the main aspect to be measured; however, measures should be expressed in such a way that calculation is possible at early stage. Additionally, our process might be extended by classifying and standardizing information for analysis, building upon some recent works on this field.

In section 2 of the paper, we introduce our process for filtering, which is described in terms of its main activities. Specific techniques and measures are referred in that context. Then, section 3 discusses some insights of the process. Finally, section 4 addresses conclusions and topics for further research.

2 An Improvement-Based Process for Filtering

Six Sigma is typically divided into five phases, creating what is referred to as DMAIC, which is an acronym for the following phases [19]:

1. *Define* the problem and identify what is important: Identify the problem and the customers; define and prioritise the customer's requirements; and define the current process.
2. *Measure* the current process: Confirm and quantify the problem; measure the various steps in the current process; revise and clarify the problem statement, if necessary; and define desired outcome.
3. *Analyse* what is wrong and potential solutions: Determine the root cause of the problem; and propose solutions.
4. *Improve* the process by implementing solutions: Prioritise solutions; and develop and implement highest benefit solutions.
5. *Control* the improved process by ensuring that the changes are sustained: Measure the improvements; communicate and celebrate successes; and ensure that process improvements are sustained.

Filtering COTS components needs to ensure – as in any Six Sigma project – that decisions are based on facts and that customer's requirements have been considered. However, in the continuing attempt to introduce COTS-based development, organisations have problems identifying the content, location, and use of diverse components. Six Sigma might help to put all these pieces together and define a measurement-based procedure for filtering COTS components.

To clarify our approach, the next section describes the first stages of the process – Define, Measure, and Analysis – in terms of their main steps and activities.

2.1 Describing the Process

In the following diagrams, a box is a rectangle representing a function, and each box on a diagram has a number in the bottom right corner to identify it within the diagram. The layout defines information/control/mechanism flows between activities as stated by the SADT technique [15].

Figure 1 defines the external interfaces for the "Filtering" process. The *Quality thresholds/Constraints* control consists of attributes that influence or constraint system's requirements and the filtering process itself. Typically, the constraint scope will include aspects such as schedule, cost, context and domain – we consider domain constraints those in which the application domain has been the cause of changes on the system's architecture, in contrast from context constraints, which have been caused by execution environment conditions. Quality thresholds represent the acceptance thresholds associated to quality attributes of the system.

The *Scenarios* input consists of different sequences of behaviour depending on the particular requests made and conditions surrounding the requests; the *COTS candidates* input consists of a number of COTS components available from marketplace; and the *Software architecture* input consists of the architectural basic units, components, and relationships among them. At this level, a basic unit for architecture is undetermined allowing multiple instantiations – such as compound units, corresponding processes, etc.

The *Stakeholders* mechanism consists of people who are involved in the filtering process. *Component sources* represent the external resources that are explored in the search of COTS candidates to be considered for evaluation.

The *Impact on stability* output consists of an identification of functional dissatisfactions according to the stability state defined during the filtering process, which embodies quality requirements and architectural aspects among others [5]. This output might include suggestions for new requirements or requirement's updates discovered during the search for COTS candidates as well as suggestions for architectural changes.

Finally, the *Filtered components* output consists of the component or components chosen for more evaluation as a result of the filtering process.

There are three primary activities in the Filtering Process: a commitment process, a pre-filtering process, and a final filtering process, as shown in Figure 2. These processes consist of activities related to the three first phases of our Six-Sigma based process for filtering – define, measure, and analysis – implicitly referring control and improvement through reporting feedback [6].

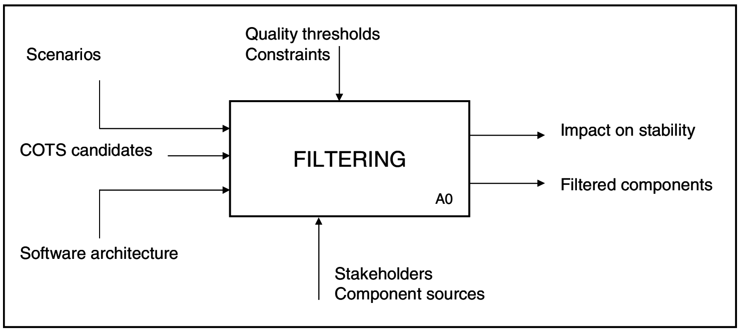

Fig. 1. Process Context

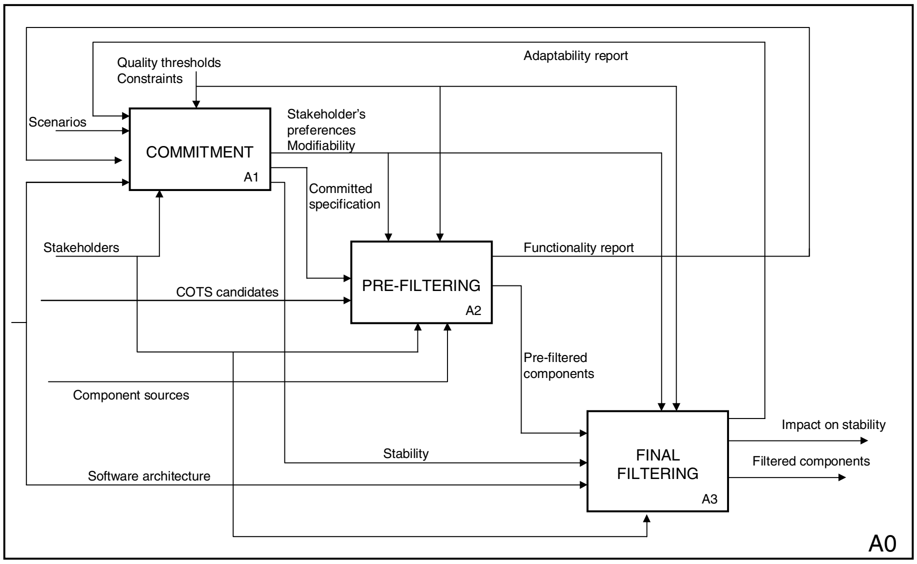

Fig. 2. Diagram 0 – Process steps

The "Commitment" process in the decomposition contains the following activities (as shown in Figure 3):

- "Derive Goals" determines the stability status of the system and provides a component specification to be committed. This activity uses information from scenario and software architecture specifications. Stakeholders use scenario authoring and goal discovering to elicit requirements at different levels of detail, and an abstract component specification is provided as input to the "Compute Preferences" process. *Desirability* is used to iteratively calibrate the abstract component specification until a *Committed specification* is produced as output. The *Goals* output consists of goals to be refined and weighted during the "Compute Preferences" process. Information about functionality and adaptability is used as refinement constraints, i.e. they drive the activities helping decide on further searching and evaluation of candidates.

- "Compute Preferences" calculates preference indicators such as desirability, modifiability and stakeholder's preferences on refined goals [7].

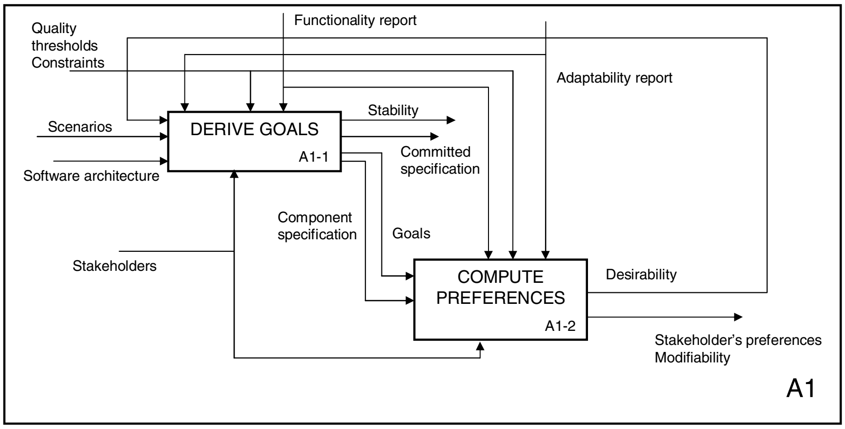

Fig. 3. Diagram 1 – Commitment steps

The "Pre-Filtering" process in the decomposition contains the following activities (as shown in Figure 4):

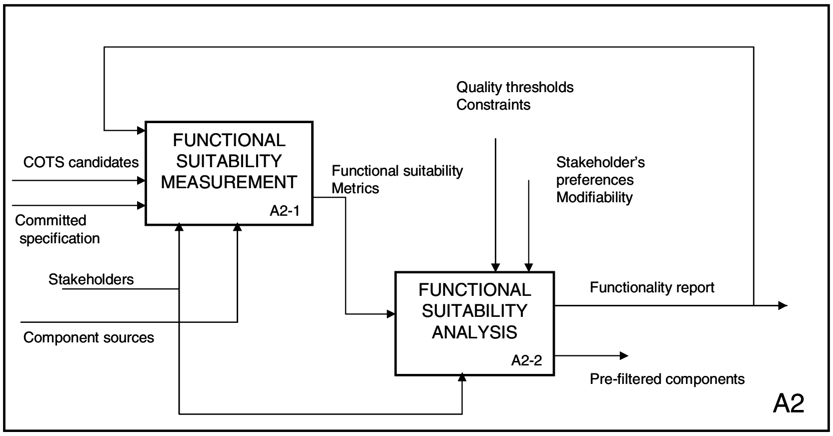
Quality thresholds Constraints
FUNCTIONAL SUITABILITY MEASUREMENT
A2-1
COTS candidates
Committed specification
Functional suitability Metrics
Stakeholder's preferences Modifiability
Stakeholders
Component sources
FUNCTIONAL SUITABILITY ANALYSIS
A2-2
Functionality report
Pre-filtered components
A2

Fig. 4. Diagram 2 – Pre-filtering steps

- "Functional Suitability Measurement" computes metrics on functional suitability of COTS candidates. *Component sources* are used as a mechanism to search candidates. Then, *COTS candidates* from a marketplace are chosen and *Functional* *suitability metrics* [8] are produced as input to the "Functional Suitability Analysis" process. The *Committed specification* acts as a guideline to *Stakeholders*, who drive the search procedure. Information from functionality is used as refinement constraints similarly to other activities in the process.

- "Functional Suitability Analysis" analyses metrics on functional suitability measured for COTS candidates. A *Functionality report* summarises the results from the analysis and serves as an indicator to decide on how to stop the search. *Stakeholder's preferences* and *modifiability* constraint the analysis taking into account the degree in which goals can be modified. The *Pre-filtered components* output consists of the component or components that are functionally suitable, and hence candidates for further evaluation.

Finally, the "Final Filtering" process in the decomposition contains the following activities (as shown in Figure 5):

- "Architectural Adaptability Measurement" computes metrics on architectural adaptability (size and complexity of adaptation, and semantic architectural adaptability) [9] on the given *Software architecture* and considering a set of *Pre-filtered* (and functionally suitable) *components*. Then, *Architectural adaptability metrics* are produced as input to the "Architectural Adaptability Analysis" process. The *Software architecture* acts as a basement to judgments of *Stakeholders*. Information about adaptability is used as refinement constraints similarly to other activities in the process.
- "Architectural Adaptability Analysis" analyses metrics on architectural adaptability. An *Adaptability report* summarises the results from the analysis and serves as an indicator to decide on reviewing stakeholder's judgements and/or continuing the search for candidates. *Stakeholder's preferences* and *modifiability* constraints the analysis taking into account the degree in which goals can be modified – this time depending on adaptability judgments. The *Filtered components* output consists of the component or components that are finally filtered. The *Impact on stability* output reports on the degree in which initial system's stability, in terms of semantic architectural aspects, is affected by the filtered components.

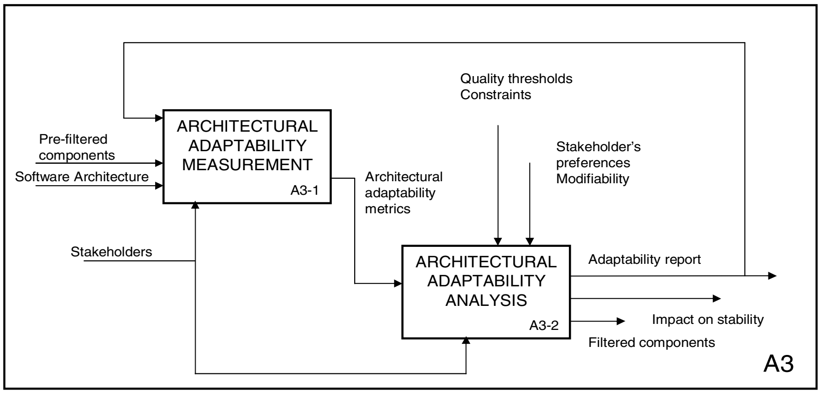

Fig. 5. Diagram 3 – Final filtering steps

3 Insights into the Process

Our proposal aims at providing a basement for improving the filtering process by a two-level improvement cycle as shown in Figure 6. The Figure additionally shows the three cycles that constitute our filtering process: (1) a "commitment" cycle, which produces a committed abstract specification S_C along with a modifiability indicator as inputs to the second cycle; (2) a "pre-filtering" cycle, in which COTS candidates are pre-selected according to their functional suitability; and (3) a "filtering" cycle, in which architectural semantic adaptability produces an indicator of stability that serves as a basis for the final candidate filtering. Note that the three cycles might also include relationships and improvements of several activities that remain internal to the process.

To define the process, we took into account how to identify suitable COTS components providing an early measure for comparison. We also considered that the evaluation of COTS candidates demands some inexact matching. The phases of our proposal were further defined by introducing some techniques and measures, which would help in establishing a basis for applying the approach. Besides, the presence of specific measures allows stakeholders to make fact-based decisions improving the analysis of COTS candidates.

But collecting effective measures is highly dependent on the amount and quality of information provided by third parties. Once requirements are categorised and weighted, a process to obtain and assess product vendor information should be carried out [3]. Closing the gap between the required and provided information also imply dealing with standard information for analysis. In this direction, recent endeavours – such as the eCots initiative [17] – might help define a web-based repository for collecting, classifying, and sharing information on software COTS products and producers.

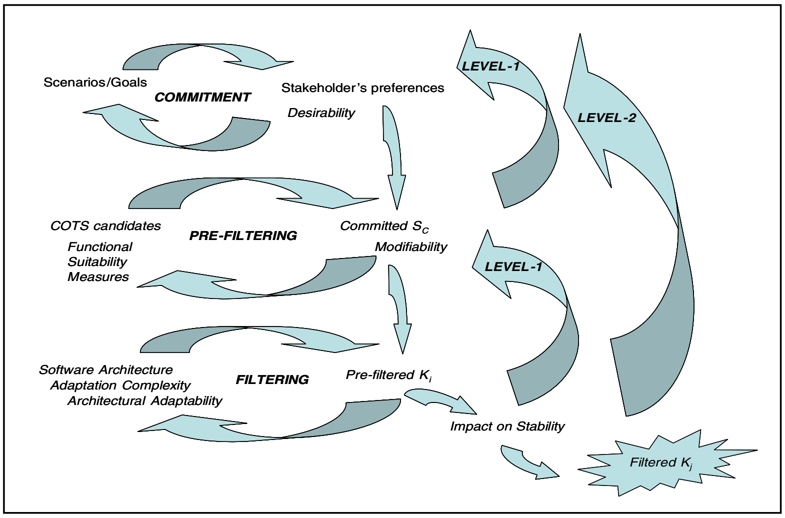

Fig. 6. Improvement in a filtering process

Metrics for COTS based systems are emerging from the academic and industrial field [15]. However, many of these definitions do not provide any guideline or context of use, which makes metric's usability dependable on subjective applications. Measures are not isolated calculations with different meanings; on the contrary, capability of measures is strongly related to the process of calculating and providing indicators based on the measures. Our approach intends to define a filtering process in which measures are included as a way of providing more specific values for comparison. At the same time, the process guides the calculation, so ambiguity is decreased.

Among other relationships, resulting measures are related to the artefact to be measured. In our approach, the artefact is expressed as functionality required by a particular application, and functionality offered by COTS candidates. Generally speaking, both cases are subject to analysing information that is modelled and weighted by people – composers or integrators on one side, and component's suppliers on the other. Different interpretations, perceptions, and judgements are then affected by the expressiveness of information. Nevertheless, our comparisons are abstract level definitions, which allow us to customise the filtering process by instantiating the calculation procedure according to different contexts of use.

As Figure 1 shows, architectural features and software requirements (scenarios) are the main inputs to drive our search of COTS candidates. From the architectural point of view, there are some additional remarks. Firstly, the impact on stability is currently based on qualitative judgements on semantic architectural adaptability, although they are combined with quantitative values of complexity and size of adaptation. We suggest here that quantitative and qualitative metrics together would help reach agreement when a decision on filtering components must be made.

Secondly, basement for decisions includes detecting architectural artefacts affected by the COTS candidate and identifying functional dissatisfactions. Causes of dissatisfaction should drive the improvement process leading to changes on the requirements definition, the host architecture, and even on the filtering process itself.

Finally, decisions on impact of stability as well as decisions made during the process might be weighted by stakeholders, in such a way that different roles and expertise are explicitly incorporated.

4 Conclusions and Future Work

We have presented a Six-Sigma based process for filtering COTS candidates. The process is based on teamwork and measurement, which allow us to provide a value for decision making. Values are calculated within a well-defined process that sets a context of use in the early stages of COTS component selection.

Of course, our process itself is subject to extension and improvement. For example, standardizing information for analysis is still an open issue that may be addressed in several ways – providing classifications and ontologies for COTS components (global and domain-oriented), defining certification issues, and so forth. Additionally, negotiation processes may be further detailed including particular features relevant to COTS development, such as negotiation on modifiability of goals.

Finally, our procedure and its particular measures are currently under validation. Among others, we are analysing the diverse ways of structuring COTS component's information to facilitate the analysis of functional matching.

References

1. R. Alexander and M. Blackburn: "Component Assessment Using Specification-Based Analysis and Testing". Technical Report SPC-98095-CMC, Software Productivity Consortium, 1999.
2. C. Alves and A. Finkelstein: "Challenges in COTS Decision-Making: A Goal-Driven Requirements Engineering Perspective". In Proceedings of the Fourteenth International Conference on Software Engineering and Knowledge Engineering, 2002.
3. B. Bertoa, J. Troya, and A. Vallecillo: "A Survey on the Quality Information Provided by Software Component Vendors". In Proceedings of the ECOOP QAOOSE Workshop, 2003.
4. R. Braga, M. Mattoso, and C. Werner: "The use of mediation and ontology technologies for software component information retrieval". In Proceedings of the 2001 symposium on Software reusability: putting software reuse in context, ACM press, pages 19–28, Ontario, Canada, 2001.
5. A. Cechich and M. Piattini: "Defining Stability for Component Integration Assessment". In Proceedings of the Fifth International Conference on Enterprise Information Systems, pages 251–256, Angers, France, April 2003.
6. A. Cechich and M. Piattini: "Managing COTS Components using a Six Sigma-based Process". In Proceedings of the Fifth International Conference on Product Focused Software Improvement, volume 3009 of LNCS, pages 556–567, Nara, Japan, April 2004. Springer-Verlag.
7. A. Cechich and M. Piattini: "Balancing Stakeholder's Preferences on Measuring COTS Component Functional Suitability". In Proceedings of the Sixth International Conference on Enterprise Information Systems, pages 115–122, Porto, Portugal, April 2004.
8. A. Cechich and M. Piattini: "On the Measurement of COTS Functional Suitability". In Proceedings of the Third International Conference on COTS-Based Software Systems, volume 2959 of LNCS, pages 31–40, Los Angeles, USA, February 2004. Springer-Verlag.
9. A. Cechich and M. Piattini: "Quantifying COTS Component Functional Adaptation". In Proceedings of the Eight International Conference on Software Reuse, volume 3107 of LNCS, 195–204, Madrid, Spain, July 2004. Springer-Verlag.
10. A. Cechich, M. Piattini, and A. Vallecillo: "Assessing Component-Based Systems". In Component-Based Software Quality: Methods and Techniques, volume 2693 of LNCS, pages 1–20, 2003. Springer-Verlag.
11. K. Cooper and L. Chung: "A COTS-Aware Requirements Engineering and Architecting Approach: Defining System Level Agents, Goals, Requirements and Architecture", Technical Report UTDCS-20-02, Department of Computer Science, The University of Texas at Dallas, October 2002.
12. L. Jaccheri and M. Torchiano: "A Software Process Model to Support Learning of COTS Products". IDI NTNU Technical Report, November 2002.
13. L. Jilani and J. Desharnais: "Defining and Applying Measures of Distance Between Specifications". *IEEE* Transactions on Software Engineering, 27(8):673–703, 2001.

14. J. Li, F. Bjørnson, R. Conradi, and V. Kampenes: "An Empirical Study on COTS Component Selection Process in Norwegian IT Companies". In Proceedings of the First International Workshop on Methods and Processes for the Evaluation of COTS Components, IEEE Press, pages 27–30, Edinburgh, Scotland, May 2004.
15. D. Marca and C. McGowan: "SADT: Structured Analysis and Design Technique", McGraw-Hill Co., 1988.
16. J. Martín-Albo, M. F. Bertoa, C. Calero, A. Vallecillo, A. Cechich, and M. Piattini: "CQM: A Software Component Metric Classification Model". In Proc. of the 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2003), pages 54-60, Darmstadt, Germany, July 2003.
17. J-C. Mielnik, B. Lang, S. Laurière, J-G Schlosser, and V. Bouthors: "eCots Platform : An Inter-industrial Initiative for COTS-Related Information Sharing". In Proceedings of the Second International Conference on COTS-Based Software Systems, volume 2580 of LNCS, pages 157–167, Ottawa, Canada, February 2003. Springer-Verlag.
18. C. Pahl: "An Ontology for Software Component Matching". In Proceedings of the Sixth International Conference on Fundamental Approaches to Software Engineering, volume 2621 of LNCS, pages 6–21, Warsaw, Poland, 2003. Springer-Verlag.
19. C. Tayntor: "Six Sigma Software Development". Auerbach Publications, 2002.
20. M. Torchiano and M. Morisio: "Overlooked Aspects of COTS-Based Development". IEEE Software 21(2):88–93, 2004.