



**Fraunhofer** Institut  
Experimentelles  
Software Engineering

# **A Method for Efficient Measurement-based COTS Assessment and Selection – Method Description and Evaluation Results**

**Authors:**

Michael Ochs  
Dietmar Pfahl  
Gunther Chrobok-Diening\*  
Beate Nothhelfer-Kolb\*

Accepted for publication in  
proceedings of the METRICS 2001  
Conference, London, UK

\* Siemens AG, Dept. ZT SE  
Otto-Hahn-Ring 6  
D-81730 München

IESE-Report No. 055.00/E  
Version 1.0  
December 2000

---

A publication by Fraunhofer IESE



Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.

The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competitive market position.

Fraunhofer IESE is directed by  
Prof. Dr. Dieter Rombach  
Sauerwiesen 6  
D-67661 Kaiserslautern



## Abstract

The use of Commercial-off-the-Shelf (COTS) software has become more and more important in state-of-the-art and state-of-the-practice software and system development. The use of COTS software promises faster time-to-market, reduced development cost, increased productivity, and the possibility for companies to focus on their own core competencies. At the same time COTS software raises risks such as economic instability of the COTS software vendor, unknown quality properties of the COTS software in use, and side effects of the COTS software on the final product. Typically, COTS-based development – in parallel to the traditional development cycle, e.g. waterfall, spiral – consists of four phases. The first phase – COTS assessment and selection – is the most crucial phase in the COTS-based cycle since long-term decisions on which COTS software will be used in a software system are made here. A late recognition that the “wrong” COTS software was used can become extremely costly for a software organization. This paper presents a repeatable, cost-efficient, and systematic method for performing measurement-based COTS assessment and selection. Moreover, cost/benefit analysis results of two industrial pilot projects, in which the method was successfully applied, are presented

**Keywords:** COTS Acquisition, Software Measurement, Decision Making, Case Study



## Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Definition of the CAP Method</b>	<b>4</b>
2.1	Outline of the CAP Model	4
2.2	Heuristics Underlying CAP	6
2.3	The CAP Evaluation Taxonomy for COTS	8
2.4	Variations of the Evaluation Depth and Process Enactment Patterns	10
<b>3</b>	<b>Design of the CAP Case Studies</b>	<b>11</b>
3.1	Case Study Metrics and Variables	11
3.2	Case Study Context Information	13
3.3	Techniques Applied for Data Collection and Analysis	13
<b>4</b>	<b>Case Study Results</b>	<b>15</b>
4.1	Enactment of the Pilot Projects	15
4.2	Evaluation Results	15
4.2.1	Results for Goal 1	15
4.2.2	Results for Goal 2	17
4.2.3	Results for Goal 3	18
4.2.4	Results for Goal 4	20
4.3	Case Study Variation Factors	21
4.4	Threats to Validity	22
4.4.1	Construct Validity	22
4.4.2	Internal Validity	22
4.4.3	External Validity	23
4.4.4	Reliability	23
<b>5</b>	<b>Conclusions and Future Work</b>	<b>24</b>
<b>6</b>	<b>References</b>	<b>25</b>
<b>7</b>	<b>Appendix A. Monte Carlo Simulation Results</b>	<b>27</b>





# 1 Introduction

The use of Commercial-off-the-Shelf (COTS) software has become more and more important in state-of-the-art and state-of-the-practice software and system development. Using COTS software promises faster time-to-market [22], which can yield substantial advantages over competitors with regards to earlier placement of a new product on a market. Option pricing models can be used to monetarily quantify this advantage [6]. In addition, COTS-usage reduces development cost [22], and increases productivity [22]. Moreover, the company can redirect effort, which is not allocated (due to COTS-based development), to traditional development activities in the areas of its own core competencies. At the same time COTS software introduces risks [21] [22] such as the economic instability of the COTS vendor, which can – in the worst case – stop maintenance support of the COTS software in use. In addition, unknown quality properties of the COTS software in use can inject harmful side effects into the final product. Typically, COTS-based development – in parallel to traditional development process models, e.g. waterfall, spiral – consists of four phases, comprising (i) COTS assessment and selection, (ii) COTS tailoring, (iii) COTS integration, (iv) maintenance of COTS and non-COTS parts of the system. The phases are depicted in Figure 1, which is adopted from [1] and tailored to the context of this paper.

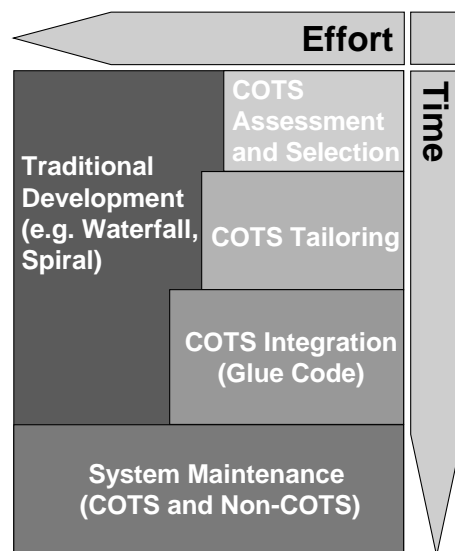


Figure 1

Phases of COTS-based and traditional software development.

The first phase – COTS assessment and selection – is the most crucial phase in the COTS-based cycle. Here long-term decisions on which COTS will be used in

a software system are made. A late recognition – either in COTS tailoring, COTS integration, or even maintenance – that a non-optimal COTS software was used in the development of the system can become extremely costly for a software organization. Therefore, repeatable and systematic methods to assess and select COTS software are an important issue in COTS-based software engineering.

Currently, there exist several solutions for COTS selection. Table 1 gives a brief summary of existing methods to COTS assessment and selection.

Table 1 Brief summary of existing approaches to COTS selection.

Approach	Brief summary
PORE [11] (Procurement-Oriented Requirements Engineering)	Elicitation of features of existing COTS software and requirements engineering are conducted in parallel. Eventually, a COTS software is selected that almost exactly fits the requirements.
OTSO [9][10] (Off-The-Shelf-Option)	Starting from a set of requirements specifying the system component, a decision taxonomy using AHP [17] and a set of measures is defined to select the most suitable COTS component in a given requirements context. The phases are screening on the full set of measures, ranking, detailed evaluation, cost and value estimation, and then the buy decision for a specific COTS software.
CEP [15][16] (Component Evaluation Process)	CEP is an advancement of the OTSO method. The activities in CEP are on a higher operational level. Activities facilitating planning of component evaluation and selection as well as evaluation scenario development and execution have been included.
Method by Taweel and Brereton [20]	Starting from a set of requirements several iterations of screening and requirements reduction are conducted, i.e. requirements that are not met by any COTS software candidate are systematically dropped. Finally the maximally covered set of requirements fits the COTS software selected.

One condition for developing CAP was that the full system requirement specification was existing prior to the decision to search for COTS software. Therefore, the PORE [11] method, which focuses on requirements engineering and COTS procurement at the same time, thus be out of scope in this context.

The OTSO [9][10] method starts from the same input as CAP but appeared too much brute-force and thus effort consuming to be applied in industrial environments. Furthermore, OTSO does not allow for a smooth and dedicated change of COTS evaluation depths.

CEP [15][16], as stated in Table 1, is an advancement of OTSO. There are several enhancements of OTSO, e.g. a planning facility for evaluating the available

component alternatives, an activity for developing, executing, analyzing evaluation scenarios, and the use of an additional mathematical method for evaluation data analysis. Overall, as the OTSO method, CEP does account for dedicated changes of evaluation depths.

The method described in [20] did not appear applicable in industrial context: No stakeholder issues are included in the method, the prioritization of the requirements is two-class (i.e. optional and mandatory), and changing of requirement priorities is not done synchronized with the requirements engineering. All methods but PORE [11] (which is out of scope in this context) form a more or less waterfall-like [2] process for reusable component evaluation and selection.

This paper presents the CAP – COTS Acquisition Process – method, a repeatable, and systematic method for performing measurement-based COTS assessment and selection. CAP is also based on OTSO, but strictly Measurement-oriented, allowing for optimization of the evaluation towards cost-efficiency, systematic changes of evaluation depth, and spiral model-like [3] enactment. Moreover, cost/benefit results of two industrial pilot projects, in which the method was successfully applied, are presented.

The paper is structured as follows: Section 2 outlines CAP, underlying heuristics, the evaluation taxonomy, evaluation depths, and enactment patterns. Section 3 explains the goals and framework for cost-benefit analysis of the industrial case studies. The cost/benefit results are presented in Section 4. Finally, conclusions on the results and pointers to future work directions are stated.

## 2 Definition of the CAP Method

### 2.1 Outline of the CAP Model

The CAP Method consists of three components: The CAP Initialization Component (CAP-IC), the CAP Execution Component (CAP-EC), and the CAP Reuse Component (CAP-RC). CAP-IC comprises all activities concerned with setting up a decision model for COTS selection and a measurement plan for assessing the COTS software alternatives. CAP-EC provides guidance for performing the assessment of the COTS software alternatives and the decision of either buying a specific COTS software product or traditionally developing the respective parts of the software system. CAP-RC enables the storing of knowledge gained about COTS software for reuse in future COTS assessment projects.

The CAP model has interfaces to the existing processes in a software organization, namely Requirements Engineering (RE)/System Design (SD), and Supply (SP). CAP receives input from RE/SD in form of requirement specification documents for the respective system component. CAP processes these documents and performs the assessment and selection of COTS software on that basis. SP is triggered to acquire the selected COTS software. After acquisition by SP, the COTS software is passed on to system integration.

The first step in CAP-IC is the identification of criteria against which candidate COTS software alternatives must be evaluated (CAP activity "Tailor & Weight Taxonomy"). In this activity the requirements are translated into a taxonomy of evaluation criteria and prioritized (or weighted) according to the Analytic Hierarchy Process (AHP) [17] under incorporation of multiple stakeholder interests [14].

The second step is to estimate how much effort will probably be needed to actually apply all evaluation criteria to all COTS software candidates (CAP activity "Estimate Measurement Effort"). This step is needed to estimate the potential cost of applying CAP, and in order to align the CAP activities with the budget and resource constraints that may apply. The estimation of measurement effort is either experience-based (from historical data) [14] or by eliciting expert knowledge [12][24].

The third step is to set up the measurement plan according to which all evaluation activities will be conducted (CAP activity "Elaborate Measurement Plan"). The measurement plan is either designed straightforward from the taxonomy of evaluation criteria – in the case the measurement effort estimates for measurement satisfy the budget and resource constraints. Alternatively, the meas-

urement plan is constructed by employing optimization algorithms with the objective to maximize priority coverage in the measurement plan while keeping the overall set of selected measures in the measurement plan within budget.

Finally, a review step certifies that all CAP-IC activities have been conducted correctly (CAP activity "IC-Review"). If the review is not successful, appropriate rework will be triggered.

CAP-EC basically consists of two elementary CAP activities for exploration of COTS software (CAP activity "Exploration") and for the review of all CAP-EC activities (CAP activity "EC-Review"), and one CAP activity cluster for COTS software assessment and selection comprising all data collection, evaluation and decision-making steps. The CAP activities contained in this cluster, i.e. "Collect Measures (1)", "Screening", "Collect Measures (2)", "Ranking", "Collect Measures (3)", and "Make-or-Buy-Decision", have been designed to make CAP an efficient and effective procedure (cf. Section 2.2 for the "Underlying Heuristics in CAP").

During the activities "Collect Measures (1-3)" data according to the measurement plan is collected on the set of COTS software alternatives. The data gained during "Collect Measures (1)" is used in "Screening" to eliminate those COTS alternatives that are unacceptable for use. The screening is performed by applying acceptance thresholds to the measures on the COTS software alternatives. In "Collect Measures (2)" data on the remaining COTS software alternatives is collected according to the measurement plan and used in "Ranking" to establish a data-driven and priority-based ranking (best to worse) of the COTS software alternatives. For the ranking the Analytic Hierarchy Process (AHP) [17] is used. As input for ranking the measures on the remaining COTS software alternatives collected in "Collect Measures (1) and (2)" are used (using data from Collect Measures (1) in ranking does not cause any additional cost). From the set of ranked COTS software alternatives the top-ranked alternative is taken and exposed to measurement of a set of final make-or-buy decision criteria in "Collect Measures (3)". In "Make-or-Buy Decision" the top-ranked COTS software alternative is checked for satisfying the criteria for the make-or-buy decision. If this is the case the process continues with "EC-Review", otherwise the second best COTS software alternative is undergoing measurement in "Collect Measures (3)", and so on. If none of the COTS alternatives passes "Make-or-Buy Decision" with a positive, i.e. "Buy", result, feedback on the non-satisfied requirements is given in EC-Review and passed to RE. In RE then it can be decided whether to reformulate or even to drop specific requirements and then iterate CAP. In addition, "EC-Review" triggers appropriate rework of CAP-EC activities, if necessary.

CAP-RC contains an activity that packages and stores all useful information generated during the enactment of a CAP project (CAP activity "Packaging for Reuse"). The purpose of this activity is to facilitate as much reuse of measurement data and information on related CAP artifacts as possible in order to de-

crease the cost of future COTS software acquisition projects. The repository in which all CAP-related information is stored has been named "CAP\_Repository".

Figure 2 gives a comprehensive control-flow based overview of the CAP model.

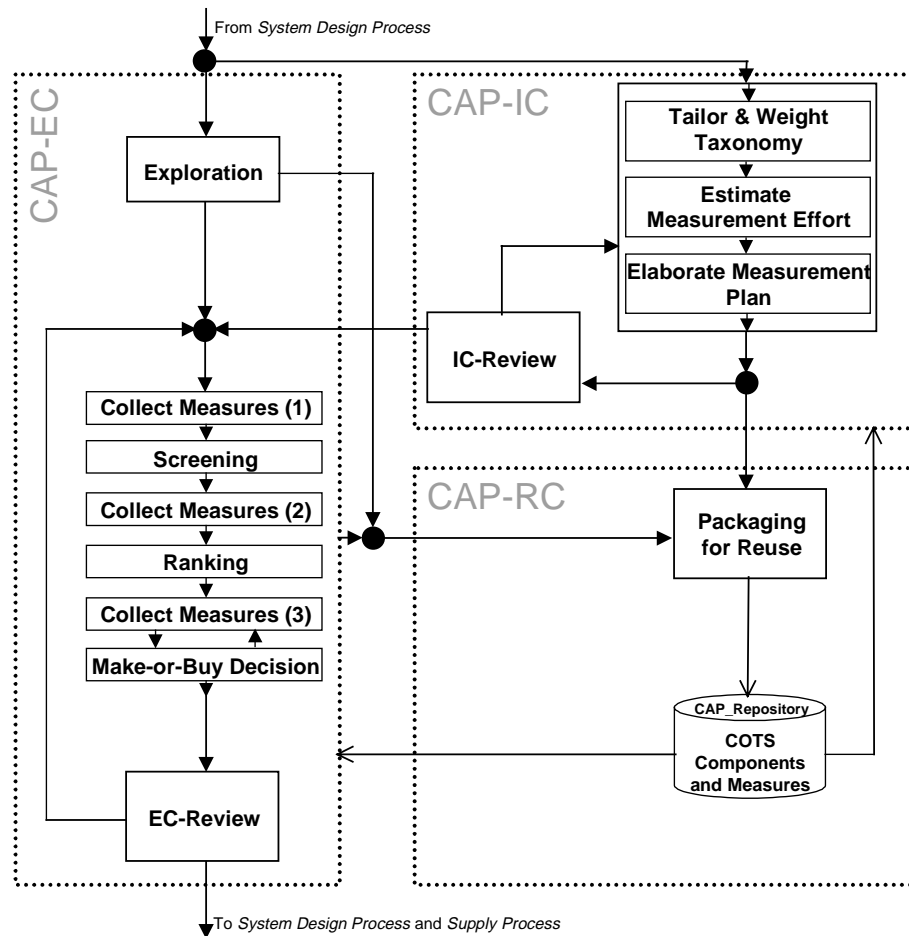


Figure 2 CAP model: Control-flow overview.

## 2.2 Heuristics Underlying CAP

CAP defines a procedure for evaluating and selecting COTS software. The starting point for this decision procedure is an initial set of COTS software alternatives and a measurement plan that defines which criteria have to be measured and evaluated in order to identify the most suitable COTS software. This could be done in a quite straightforward manner by simply measuring all applicable criteria on all COTS software alternatives. However, this brute-force approach can be quite expensive since (i) many COTS software alternatives might be available, (ii) the set of evaluation criteria could be quite large, and (iii)

some of the criteria might be very difficult or expensive to measure, e.g. reliability.

Let  $n$  be the number of COTS software alternatives identified during exploration and  $k$  the number of metrics in the measurement plan. Then  $n \times k$  is the overall number of measures that must be collected for making a straightforward decision. In order to reduce this effort and thus make the process more efficient the CAP heuristic was designed. Figure 3 depicts how the decision-making and measurements activities of CAP interact via the set of COTS software alternatives.

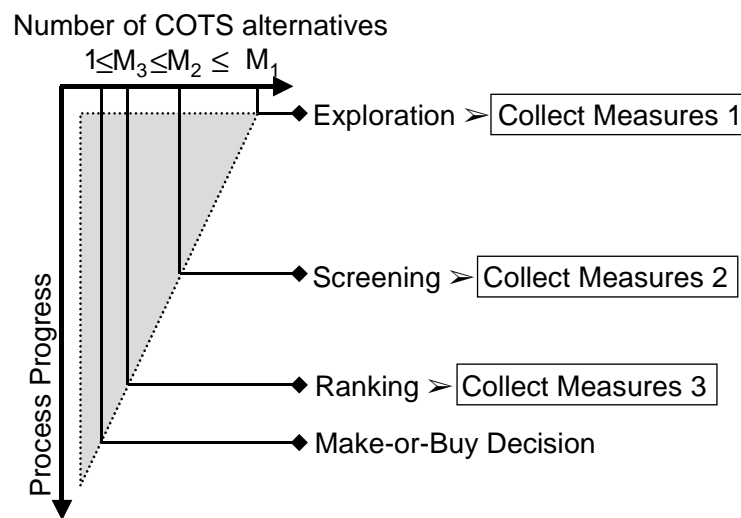


Figure 3

Effect of the heuristic along the enactment of the process.

The measurement plan consists of three parts ( $i = 1, 2, 3$ ), with  $k_i$  metrics in each part. The overall number of metrics in the measurement plan is  $k$  with  $k = k_1 + k_2 + k_3$ .

The initial number of alternatives resulting from CAP activity "Exploration" is  $M_1$ , which is reduced to  $M_2 \leq M_1$  in CAP activity "Screening" employing the  $k_1$  measures of measurement plan, part 1, collected in "Collect Measures (1)".

Then the  $M_2$  COTS software alternatives are examined using the  $k_2$  measures of measurement plan, part 2, collected during "Collect Measures (2)". Afterwards, the alternatives are being ranked in CAP activity "Ranking" based on the information coming from the  $k_2$  measures on the  $M_2$  alternatives. Finally, the  $M_3 < M_2$  top-ranked alternatives are measured using the  $k_3$  measures from part 3 of the measurement plan.

This yields an overall effort for data collection of

$$effort_{CAP} = \sum_{i=1}^3 \sum_{j=1}^{k_i} effort(m_{ij}) \cdot M_i,$$

while the effort without the CAP heuristic would be

$$\begin{aligned} effort &= \sum_{i=1}^k effort(m_{1j}) \cdot M_1 \\ &= M_1 \cdot \sum_{i=1}^k effort(m_{1j}) \end{aligned}$$

where  $m_{ij}$  specifies metric  $j$  of part 1 of the measurement plan and  $M_i$  specifies the number of COTS software alternatives assessed in measurement activity  $i$  ("Collect Measures (i), where  $i=1,2,3$ ) of CAP-EC. Comparing the two effort equations it is clear that  $effort_{CAP} < effort$ , which justifies the application of the CAP heuristic. Note, that in the second formula  $k = k_1$ , because in a brute force decision-making procedure there is no partitioning of the measurement plan and all metrics have to be always applied to all COTS software alternatives identified in CAP activity "Exploration". Overall, CAP-EC executes the measurement plan developed in CAP-IC by trying to achieve highest possible efficiency. The modular structure of the measurement plan in CAP is achieved by the Kano-Model [5], which distinguishes different classes of product features, i.e. dissatisfiers, satisfiers, and delighters, that are associated with different types of utility functions and thus lead to different states of customer satisfaction depending on the features a product possesses.

The effectiveness of CAP strongly depends on the expressiveness of the criteria selected for evaluation. But it cannot be expected that all selected criteria can always be measured when the project plan constraints on cost, resources, and time have to be satisfied. So, in the case the initial set of evaluation criteria is not feasible in the scope of the project plan, a trade-off between the effectiveness of the evaluation criteria and the cost, time, and resource allocation of the criteria while measuring must be reached. This is achieved by maximizing the coverage of important criteria/measures in the measurement plan while not exceeding the budget and resource constraints. Methods for tackling such integer linear programming problems can be found in [13].

### 2.3 The CAP Evaluation Taxonomy for COTS

The core part of the selection performed using CAP is the CAP Evaluation Taxonomy. A high level description of the taxonomy is depicted in Figure 4.

The evaluation taxonomy is tailored and weighted in the CAP-EC activity "Tailor & Weight Taxonomy". The underlying technique for weighting the taxonomy is the Eigenvector Method of the Analytic Hierarchy Process (AHP) [17]. AHP is



also used for ranking the COTS software alternatives and performing a sensitivity analysis of the ranking in the CAP-EC activity "Ranking". Note that the activity "Screening" is only performed on a subset of all measures (evaluation criteria), which is defined by the CAP heuristic for efficiency. The categories and criteria that will be in use are devised by stepwise subset construction from the initial set of requirements delivered as input to CAP. The assignment of requirements to categories is based on the definitions of the categories and criteria. The categories and criteria are based on the definitions stated in ISO 9126 Standard on Product Quality [8], additional criteria definitions from literature [9] [10] [11], and results from applied research.

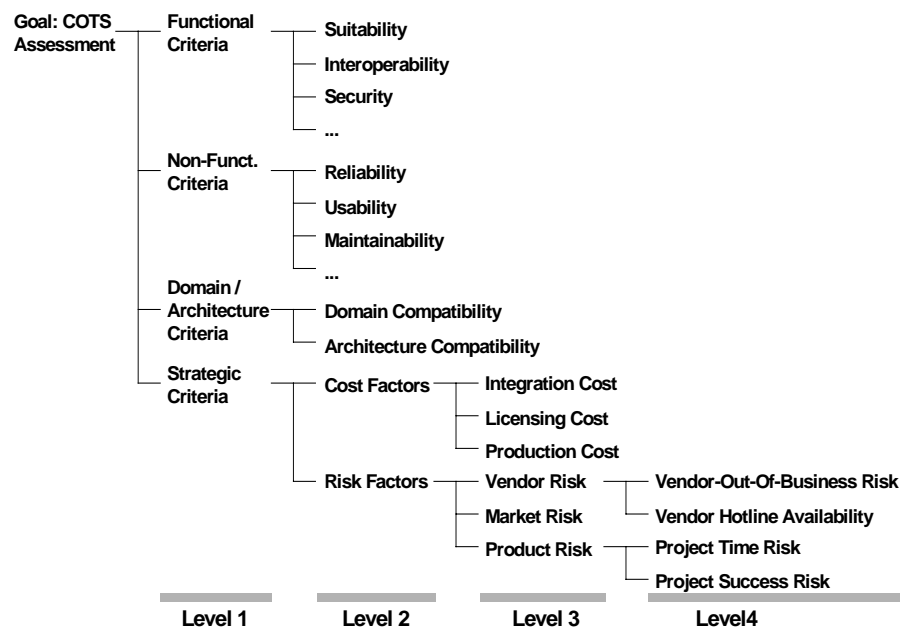


Figure 4

CAP Evaluation Taxonomy.

The taxonomy consists of four levels of criteria. At the lowest level (level 4), the taxonomy is operational by means of metrics, e.g. Project Success Risk, for collecting data on the COTS software alternatives. The range of criteria is, as can be seen from Figure 4, considerably wide, i.e. criteria on product quality [8] as well as criteria on strategic aspects like cost, risk, and vendor issues. For more detailed information see [14], in which the focus is more on the method description and thus also on the structure of the taxonomy. In a whole, the taxonomy provides a set of more than 120 pre-defined metrics from which the user can choose or even add new ones.

## 2.4 Variations of the Evaluation Depth and Process Enactment Patterns

Another issue while employing CAP is the evaluation depth in use when conducting the assessment and selection steps of CAP-EC. The term evaluation depth describes the way, in which the measures have been collected in the respective measurement activities. Possible classes (on an ordinal scale [5]) of evaluation depth (ED) are (i) documentation/interview/clarification, (ii) expert knowledge, (iii) non/functional testing, (iv) scenario-based testing, and (v) prototyping. Obviously, from ED class (i) through ED class (v) the effort for collecting data is increasing. Therefore, CAP starts with the lowest ED possible, i.e. (i) or (ii). When a reduction in the number of alternatives has been gained the evaluation depth can be increased. This strategy can save a considerable amount of effort. In addition, increasing the ED constantly along the process increases the confidence in and reliability of the COTS selection decision at the end of the process. From this, a large variation of possible enactment patterns for CAP projects is possible. Almost every enactment pattern that obeys the rules induced by the control-flow of the model can be conducted. The variety of enactment patterns ranges from a single-ED class based sequential enactment pattern to a concurrent, highly iterative, multi-ED class enactment pattern.

Overall, there is a set of potential triggers for iteration of certain CAP activities. Triggers could be a change of the evaluation depth (as described above, iteration of CAP-EC), the identification of new COTS software alternatives to be assessed (iteration of CAP-EC), change of the requirements specification (iteration of CAP-IC and CAP-EC), and review findings (iteration of CAP-IC and/or CAP-EC). An example of an enactment pattern including iteration of CAP-IC and CAP-EC is depicted in Figure 5.

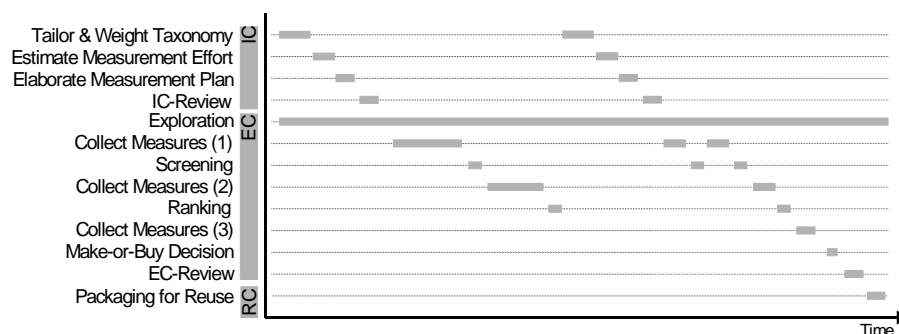


Figure 5 Concurrent multi-ED class iterative enactment pattern.

Possible reasons for the iterations in Figure 5 could be – as stated above – a change of the requirement specification or review findings (for iteration of CAP-IC), and a change of the evaluation depth, the identification of new COTS software alternatives, or review findings (for iteration of CAP-EC).

### 3 Design of the CAP Case Studies

The CAP case studies were performed in a Siemens Business Unit (SBU) operating in the Information System domain. Both case studies were performed simultaneously during a 7- respectively 8-months period in different locations. CAP was introduced to the SBU for the purpose of improving the current practice of COTS assessment and selection, which was an ad-hoc practice.

The evaluation goals of the case studies were 4-fold: (i) verification of the effect of the CAP heuristic, (ii) evaluation of the additional cost caused by CAP, (iii) evaluation of the process improvement induced by CAP, and (iv) evaluation of the COTS-based vs. the traditional development approach.

#### 3.1 Case Study Metrics and Variables

The variables and metrics used in the case study were devised using the Goal/Question/Metric Paradigm [4] [7] [18]. The GQM Goal definitions are described in Table 2 using the GQM Goal Template.

Table 2

Case Study GQM Goal definitions.

GQM Template	GQM Goal 1	GQM Goal 2	GQM Goal 3	GQM Goal 4
<b>Object</b>	CAP			COTS-based development approach
<b>Purpose</b>	Evaluating			
<b>Quality Focus</b>	effect of heuristic	CAP-related add-on cost	process improvement	cost and benefit
<b>Viewpoint</b>	SBU Management and Developers of CAP			
<b>Context</b>	SBU Pilot Projects			

The definitions of the case study variables and metrics are listed in Table 3.

Table 3 Case study metrics: Metric, Description, Definition, Scale, and Range.

Metric	Description	Definition	Scale	Range
CDE (selection)	Effort spent on CAP-based assessment and selection of the COTS component	$CDE_{selection} = \text{Effort of performing CAP (person hours or person years)}$	absolute	$0 < CDE_{selection} < \infty$
CDD (selection)	Duration of CAP-based assessment and selection of the COTS component	$CDD_{selection} = \text{Duration of performing CAP (years)}$	absolute	$0 \leq CDD_{selection} < \infty$
CDE (integration)	Expert estimate on the effort needed for integrating of the COTS component with the software system	$CDE_{integration} = \text{Effort of COTS integration and tailoring (person hours or person years)}$	absolute	$0 \leq CDE_{integration} < \infty$ Format: $smin - smlv - smax$
CDD (integration)	Expert estimate on the duration of integrating (includes tailoring) the COTS component with the software system	$CDD_{integration} = \text{Duration of COTS integration and tailoring (years)}$	absolute	$0 \leq CDD_{integration} < \infty$ Format: $smin - smlv - smax$
TDE	Expert estimate of the traditional from-scratch development effort of the system component	$TDE = \text{Traditional development effort (person hours or person years)}$	absolute	$0 \leq TDE < \infty$ Format: $smin - smlv - smax$
TDD	Expert estimate of the traditional from-scratch development duration of the system component	$TDD = \text{Traditional development duration (years)}$	absolute	$0 \leq TDD < \infty$ Format: $smin - smlv - smax$
NMP (screening)	Number of measurements performed for screening	$NMP_{screening} =  \{\text{measurements during "Collect measures (1)"}\} $	absolute	$0 \leq NMP_{screening} < \infty$
NMP (ranking)	Number of measurements performed for ranking	$NMP_{ranking} =  \{\text{measurements during "Collect measures (2)"}\} $	absolute	$0 \leq NMP_{ranking} < \infty$
NCA (screening)	Number of COTS alternatives to be measured for screening	$NCA_{screening} =  \{\text{COTS alternatives to be measured during "Collect measures (1)"}\} $	absolute	$0 \leq NCA_{screening} < \infty$
NCA (ranking)	Number of COTS alternatives to be measured for ranking	$NCA_{ranking} =  \{\text{COTS alternatives to be measured during "Collect measures (2)"}\} $	absolute	$0 \leq NCA_{ranking} < \infty$
NCR	Number of COTS software requirements	$NCR =  \{\text{Requirements for COTS software component}\} $	absolute	$0 < NCR < \infty$
CTYPE	COTS type under assessment in case study	$CTYPE = \text{COTS software type under assessment in case study}$	nominal	$CTYPE \in \{\text{Database System, Speech Recognition, ...}\}$
NEI	Number of experts involved	$NEI =  \{\text{Experts involved in the assessment}\} $	absolute	$0 \leq NEI < \infty$
YCE	Years of professional experience with COTS type	$YCE = \text{Number of years of experience expert has with COTS type [years]}$	absolute	$0 \leq YCE < \infty$
TCE	Type of professional experience with COTS type	$TCE = \text{Type of professional experience with COTS type}$	nominal	$TCE \in \{\text{none, user, consultant, developer, development manager}\}$

For GQM Goal 1, the effect of the heuristic is measured through the number of measurement actions until a make-or-buy decision regarding COTS software is made. The measurement effort (i.e. cost) is growing with the number of number of measurement actions. The cost of COTS assessments is dependent on the number of evaluation criteria (i.e. number of different metrics). Even if the exact relationship between evaluation criteria and COTS assessment cost is not yet precisely known, one assumption on the relationship can be seen as evident and stable: the COTS assessment cost is a monotonous increasing function of the form  $\text{Cost/Effort} = F(\text{evaluation criteria, COTS software alternatives, evalua-}$

tion depth). Thus using the number of measurement actions, i.e. the number of evaluation criteria per assessment step (c.f. Sections 2.1 and 2.2), for evaluating CAP in GQM Goal 1 is adequate. GQM Goal 2 evaluates the additional cost of CAP, being a systematic and formal approach to COTS assessment and selection, compared to ad-hoc COTS selection. In GQM Goal 3 the qualitative beneficial aspects of CAP gained by the additional cost (c.f. GQM Goal 2) are evaluated. The results of GQM Goal 4, which evaluates the cost/benefit ratio of COTS-based vs. traditional software development, provide some quantitative insight into the profitability of COTS-based development. Several authors state that COTS-based development is an economical choice compared to traditional development, e.g. [22], due to cost and time saving but do not provide any quantitative evidence. The results of GQM Goal 4 provide quantitative evidence on this issue by calculating Return on Investment (ROI), Cost Effectiveness (CE), and Time Saving Ratio (TSR) of COTS-based development defined as functions of the development approach.

### 3.2 Case Study Context Information

Two case studies were performed in parallel at a Siemens Business Unit operating in the Information System Domain. Both case studies were conducted at different locations and by different people. A more complete set of variation factors describing the context of the case studies is given in Section 4.3.

### 3.3 Techniques Applied for Data Collection and Analysis

The methods and techniques applied for collecting the data in the case studies are (i) GQM [4] [7] [18], (ii) expert knowledge elicitation [12] [24], and (iii) subjective measurement [19].

GQM is a systematic approach to measurement based on an explicit measurement goal definition. In the goal definition, the object (e.g. process) and the purpose of measurement (e.g. monitoring), the quality focus of the object (i.e. the attribute of interest, e.g., effort), the viewpoint (i.e. who is interested in the data), and the context (i.e. environmental setting) are specified. For each goal, questions, variation factors, hypotheses, and variation hypotheses are devised and refined into metrics. For more detailed information on GQM see, e.g., [4] [7] [18].

Expert knowledge elicitation is a subjective approach to measurement of attributes of objects for which no objective data is available. The basic concept is to ask experts in the respective field for their estimate on the interrogated attribute to be measured. The expert provides a subjective most likely value (smlv). In addition, in order to capture issues of uncertainty, the expert also provides a subjective minimum (smin) and maximum (smax) value forming a range for the respective attribute. These subjective values describe a triangular probability dis-

tribution covering a range, in which the real value of the attribute is likely to be located from the expert's view. From triangular distributions the mean (expected) value can be calculated or they can be used as input for Monte Carlo simulations. For more detailed information on expert knowledge elicitation see [12] , for Monte Carlo simulations see, e.g., [24] .

Subjective measurement can be used in situations when it is impossible or too expensive and time-consuming to objectively measure specified attributes of objects. Therefore instruments for subjectively measuring the attributes are used. These instruments consist of one or more questions with answering scales attached to the questions. There are two types of answering scales, i.e. Likert Scales and Semantic Differential Scales [19] . Likert Scales are usually used to enable the respondents to provide their answer by choosing a category (nominal or ordinal value). Semantic Differential Scales are used to enable the respondents to express their perception on the interrogated attribute of the object under measurement by providing the answer ordinal scaled. Important is that Semantic Differential Scales have semantic opposites of an attribute value on the ends of the scale, such as "good" on the one and "bad" on the other end of the scale. For more detailed information on subjective measurement see, e.g., [19] .

## 4 Case Study Results

### 4.1 Enactment of the Pilot Projects

The results of performing the COTS assessment and selection case studies are briefly described in Table 4. This includes a brief description of important measures on the enactment of CAP such as number of COTS software alternatives before and after each evaluation step, number of measurement actions per evaluation step.

Table 4

Case study overview.

Case Study Information	Values	
	Case Study 1	Case Study 2
Initial number of alternatives	4	5
Number of screening criteria	13	19
Alternatives after screening	2	4
Number of ranking criteria	18	31
Number of Make-or-Buy Decision Criteria	0	0

The number of criteria in “Make-or-Buy Decision” is 0 since there was no option to develop the software by the traditional development approach. Instead there was a “rule” to buy the top-ranked COTS software in both case studies.

Finally, in both case studies a piece of COTS software was selected for tailoring and integration with the software system under development. More detailed information on the set of criteria, the COTS software alternatives, and the COTS selection decision in case study 1 can be found in [14].

### 4.2 Evaluation Results

#### 4.2.1 Results for Goal 1

GQM Goal 1 evaluates the effect of the heuristic embedded in CAP by comparing the number of measurement actions caused by CAP to the number of measurement actions a brute-force approach would have needed until a make-or-buy decision is made. Table 5 summarizes the measurement results.

In Table 5, NMA is defined as

$$NMA(Approach) = NMP_{screening} \times NCA_{screening} + NMP_{ranking} \times NCA_{ranking},$$

where *Approach* is either CAP or Brute-force.

The blackened cells in Table 5 are not applicable in the brute-force case. This is due to the fact that in a brute-force approach *all* measures are collected before screening. Thus, the overall number of measurement actions is  $31 \times 4 = 124$  and  $50 \times 5 = 250$ , respectively, which maps to an associated effort for data collection.

From Table 5 the reduction of measurement actions calculates as 29.03% for case study 1 and 12.4% for case study 2. This means that there is a considerable amount of measurement actions – and thus measurement effort – which could be saved by the heuristic embedded in CAP. Regarding expensive measures of COTS software it is clear that the decrease in NMA can map to considerable amount of effort, and respective cost saving.

Table 5

Measurement results – goal 1.

Measure	Case Study 1		Case Study 2	
	CAP	Brute-force	CAP	Brute-force
<b>NMP</b> (screening)	13	31	19	50
<b>NCA</b> (screening)	4	4	5	5
<b>NMP</b> (ranking)	18	not applicable for brute-force approach	31	not applicable for brute-force approach
<b>NCA</b> (ranking)	2		4	
<b>NMA</b> <b>(overall)</b>	<b>88</b>	<b>124</b>	<b>219</b>	<b>250</b>



#### 4.2.2 Results for Goal 2

GQM Goal 2 evaluates the additional effort caused by CAP compared to the effort an ad-hoc COTS software selection would have cost. In this context *ad-hoc* means that all requirements to the component would have been checked the same way they were checked with CAP but there would have been no systematic decision modeling.

Table 6 Measurement results – goal 2.

Measure	Effort [ph]			
	Case Study 1		Case Study 2	
	CAP	ad-hoc	CAP	ad-hoc
<b>CDE</b> (selection)	91.0	61.5	2342.5	2203.0
<b>CDE<sup>1</sup></b> (integration)	smin: 1600 <sup>2</sup> smlv: 2400 smax: 3200		smin: 3200 smlv: 4000 smax: 8000	
<b>Overall<sup>3</sup></b> <b>(expected)</b>	<b>2491</b>	<b>2461.5</b>	<b>7409.2</b>	<b>7269.7</b>

Furthermore, there also would have been no explicit incorporation of multiple stakeholders' interests and no sensitivity analysis of the make-or-buy decision. In the sensitivity analysis changes of the decision when changing the weighting of the criteria in the Evaluation Taxonomy for COTS are analyzed [17] (c.f. Section 2.3). Table 6 lists the effort for selecting a COTS component, the (effort based on expert opinion) for integrating (which also includes COTS tailoring), and the overall effort for the full COTS-based cycle. Possible maintenance efforts are excluded in the study due to the fact that the system is still under development.

From Table 6 the additional cost of COTS-based development using CAP compared to ad-hoc COTS selection can be calculated. COTS-based development using CAP had approximately 1.2% higher expected cost in case study 1 and approximately 1.9% higher expected cost in case study 2. The process improvement that is bought into the COTS-based development cycle regarding the COTS selection decision through the additional cost is discussed in the context of GQM Goal 3 (c.f. Section 4.2.3).

<sup>1</sup> The values for CDE (integration) are based on expert opinion.

<sup>2</sup> The conversion factor was 1600 person hours per person year.

<sup>3</sup> The overall (expected) values were calculated as the sum of CDE (selection) and the expected (mean) value of the expert opinions on CDE (integration), where the mean was  $(1/3) \times (smin + smlv + smax)$

### 4.2.3 Results for Goal 3

GQM Goal 3 evaluates the perceived process improvement by introducing CAP as a replacement of the former ad-hoc COTS software selection practice by subjective measurement [19]. The subjective measures captured the perception of those people involved in the case studies regarding the (i) objectivity of the COTS software decision, (ii) reliability of the COTS software decision, (iii) traceability of the COTS software assessment and selection, (iv) repeatability of the process, (v) explicitness (documentation) of the COTS software decision, (vi) efficiency (effort) of enacting CAP, (vii) willingness of the management to accept the COTS software decision, and (viii) consideration/incorporation of multiple stakeholders' interests. In each of these subjective measures there was a direct comparison between CAP and ad-hoc COTS selection employing a 5-point semantic differential scale [19] with CAP and ad-hoc on the ends of the scale. The scale was chosen to be 5-point in order to enable the respondents to express their perception that neither CAP nor ad-hoc COTS selection has an advantage over the respective other approach. The measurement results are listed in Table 7. The scale was constructed as follows. A value of 5 expresses that CAP is better with respect to the inquired issue. A value of 3 expresses a perception of equivalence regarding CAP and ad-hoc with respect to the inquired issue. Finally, a value of 1 expresses the perception that ad-hoc is better than CAP with respect to the inquired issue.

Table 7 Measurement results – goal 3.

Measure	Min	Median	Max
Objectivity of COTS decision	4	4	5
Reliability of COTS decision	3	4	5
Traceability of process	4	4	5
Repeatability of process	4	4	5
Explicitness of process	3	4	5
Efficiency (effort) of process	3	3	5
Management acceptance of COTS decision	4	5	5
Incorporation of multiple stakeholder interests	3	3	4

The data listed in Table 7 is visualized using a radar plot with gauges in Figure 6. The radar plot includes the minimum (thin dotted lined gauge), median (thin interrupted lined gauge), and maximum (thin solid lined gauge) values for the ratings on CAP. For the sake of comparability, the gauge of equivalence of CAP and ad-hoc (value of 3 on the 5-point scale, visualized by the thick solid lined gauge) is included in the radar plot. Any data point located in the interior of the equivalence gauge demonstrates a deterioration of the practice by introducing CAP. Any point outside the equivalence gauge demonstrates an improvement of the practice by introducing CAP. For data points located exactly

on the edge of the equivalence gauge, there is neither improvement nor deterioration by introducing CAP, e.g. for the efficiency of the process and multi-stakeholder incorporation.

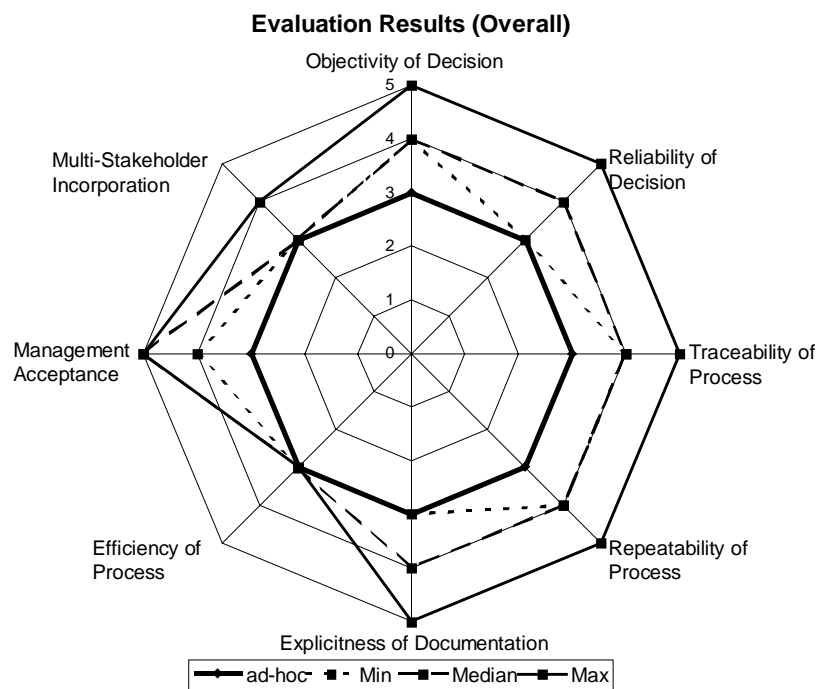


Figure 6

Radar plot visualization of the perceived process improvements.

Overall, there was no deterioration of the COTS software acquisition practice caused by introducing CAP to the organization. Regarding the objectivity of the decision, reliability of the decision, traceability of the process, repeatability of the process, explicitness of documentation, and management acceptance of the COTS selection decision CAP achieves a major improvement over the current COTS software acquisition practice at the SBU. With regards to the efficiency of the process, and multi-stakeholder incorporation there was no improvement and no deterioration of the current COTS software acquisition practice at the SBU.

Considering the improvements perceived by the case study participants, the fact that the efficiency of CAP was comparable to the ad-hoc COTS software acquisition practice clearly qualified CAP as an economic choice. Moreover, this perception is supported by the fact that the additional effort of CAP was below 2% in the case studies (c.f. Section 4.2.2).

Regarding the incorporation of multiple stakeholder views on the COTS software selection problem there was no improvement achieved by introducing CAP within the SBU. This is plausible since the SBU had carried out exhaustive requirements engineering before starting CAP, incorporating multiple possible

stakeholders. The requirements were then engineered using the Kano Model [5] .

Overall, the perceptions of the personnel involved in the CAP pilot projects demonstrate that CAP was a successful approach for improving the COTS acquisition practices at the SBU.

#### 4.2.4 Results for Goal 4

GQM Goal 4 had a more general view on COTS-based development compared to traditional development.

Defining economic indicators Return on Investment (ROI), Cost Effectiveness (CE), and Time Saving Ratio (TSR) as functions of the development approach quantifies the economic advantages of COTS-based development several authors qualitatively write about [22] [6] . The economic indicators are defined as follows [23] :

$$ROI = \frac{TDE - CDE}{CDE}, CE = \frac{TDE - CDE}{TDE}, TSR = \frac{TDD - CDD}{TDD}.$$

The respective values for TDE and TDD are based on expert knowledge elicitation (c.f. Section 3.3). CDE and CDD are the overall effort and time for COTS based development, where CDE is taken from Table 6. See Table 3 for details on CDE, TDE, CDD, and TDD.

Table 8 lists the measures needed to calculate ROI, CE, and TSR and the indicator results as well.

The values for ROI, CE, and TSR were generated using Monte Carlo simulation. The values in the table are the mean values resulting from the simulations. For full details on the simulation results, i.e. minimum, mean, mode, maximum, and standard deviation of ROI, CE, and TSR for both case studies, see Appendix A.

The data in Table 8 demonstrates that COTS-based development was an economic choice in both case studies. The ROI is considerably above 0, which means that for, e.g., every person hour invested in the COTS-based development approach approximately 5 and 3 person hours of effort were saved in comparison to the traditional development approach.<sup>4</sup> The CE in both case studies is also at a considerable level of saving and expected at least 75%. In the first case study, there was also a non-negligible amount of time saving, i.e.

---

<sup>4</sup> One underlying assumption is that – in the case that there is COTS licensing cost for each delivered system containing the COTS software – these cost are passed on to the customer buying the software system developed using COTS.

67%. In the second case study, the time saving turned negative, i.e. -6%. This observation will be further discussed in the conclusions.

Table 8

Measurement results – goal 4.

Measure	Case Study 1	Case Study 2
CDE (selection)	91 ph	2342.5 ph
CDE (integration)	smin: 1600 ph smlv: 2400 ph smax: 3200 ph	smin: 3200 ph smlv: 4000 ph smax: 8000 ph
CDE (expected overall)	2491 ph	7409.2 ph
TDE	smin: 12800 ph smlv: 16000 ph smax: 19200 ph	smin: 16000 ph smlv: 32000 ph smax: 48000 ph
<b>ROI (COTS-based vs. traditional)</b>	<b>5.54</b>	<b>3.40</b>
<b>CE (COTS-based vs. traditional)</b>	<b>84.32%</b>	<b>75.76%</b>
CDD (selection)	0.6 y	0.7 y
CDD (integration)	smin: 0.5 y smlv: 0.6 y smax: 1.0 y	smin: 1.0 y smlv: 1.2 y smax: 2.0 y
CDD (expected overall)	1.3 y	2.1 y
TDD	smin: 3.0 y smlv: 4.0 y smax: 5.0 y	smin: 1.0 y smlv: 2.2 y smax: 3.0 y
<b>TSR (COTS-based vs. traditional)</b>	<b>67.15%</b>	<b>-6.27%</b>

### 4.3 Case Study Variation Factors

A subset of the variation factors used to characterize the contexts of the case studies is listed in Table 9.

Table 9

Case study variation factors.

Variation Factor	Case Study 1	Case Study 2
CTYPE	Database System	Support AdministrativeTasks
NCR	30	94
NEI	1	0 (+1 consultant, part time)
YCE	2	2 (with totally different technical level and domain)
TCE	(sophisticated) user	development manager

As Table 9 demonstrates, both case studies had different contextual settings. The type of COTS software is different; the numbers of requirements specifying the system components largely differ. In addition, there was almost no expertise in case study 2, whereas there was expertise on the COTS software type in case study 1. The variation factors, partially being at opposite ends of scale, largely explain the fact that there is also a substantial effort difference in the assessment and selection phases of both case studies.

In case study 1, the expert involved was able to provide data for almost all metrics by his expertise, which lead to case study 1 having an evaluation depth of almost fully level 1 (documentation, clarification, interviews) and 2 (expert knowledge) and thus very low effort.

In case study 2 the expertise on the COTS software type was comparatively low and the effort of performing the COTS software assessments thus was very high. This is since people mostly had to rely on performing the assessments in evaluation depths of level 1 to 4, where level 4 (scenario-based testing) was predominant.

#### **4.4 Threats to Validity**

This section discusses the major threats to validity of case studies [25] .

##### **4.4.1 Construct Validity**

The measures applied in Goal 1 originate from objective measurement and thus are reliable and correct. The measures applied in Goal 3 are fully based on subjective measurement. Due to the very small number of participants in the case studies it is not possible to validate the measures applied. However, the items very clearly explained to the respondents. The measures in Goals 2 and 4 are partially based on objective measurement – thus correct and reliable – and partially based on expert judgement [12] , where the attributes being measured by expert knowledge elicitation were clearly explained to the respondents and the process of expert knowledge elicitation [12] was obeyed.

##### **4.4.2 Internal Validity**

This threat to validity is non-existent [25] since the case study has an exploratory character.

#### **4.4.3 External Validity**

The case study was performed in the Information System Domain and for two specific types of COTS software. From this it is not possible to generalize widely across other domains in Software Engineering. However, currently there are no apparent issues that the results must not be generalized to other domains and other types of COTS software. Future case studies will verify or falsify the generalization aspect.

#### **4.4.4 Reliability**

For Goal 1 the data collection procedure was GQM which follows a defined and systematic procedure. It can be assumed that using GQM at least reduces some of the reliability risks, which holds also for those measures of Goals 2 and 4 that were collected by objective measurement. Regarding Goal 3 it was possible to verify one respondent's answers by iterating the subjective measurement several weeks after the first response resulting in identical answers as the prior ones.

## 5 Conclusions and Future Work

This paper demonstrates that COTS-based development (partially) was an economic choice compared to traditional development for the very specific context of the information systems developed at the SBU.

Moreover, it could be demonstrated that in the case studies CAP was more efficient than brute-force approaches to COTS assessment and selection.

The additional cost caused by CAP for COTS assessment and selection seen over the full COTS-based development cycle (c.f. Figure 1) was below 2% in both case studies. This means that the considerable advantages of CAP over an ad-hoc COTS selection practice (c.f. GQM Goal 3, Section 4.2.3) just cause marginal additional cost. But the perceived benefits gained from the 2% additional cost have been considerable. CAP was perceived to achieve an objective and reliable decision, while the process was traceable, repeatable, explicit, and efficient. The management levels were willing to preferably accept a COTS selection decision made using CAP instead of a – at least partially – subjective ad-hoc COTS selection decision.

The economic indicators defined as functions of the development approach demonstrate that COTS-based development – maintenance excluded since there is no data on this issue yet – was fully economically beneficial for the SBU (all indicators positive) in the first case study. This quantitatively substantiates the frequent claim that COTS-based development is economically beneficial. In the second case study ROI and CE remained positive but TSR had a negative expected value (c.f. Appendix A). This fact conforms to the statement “the promise to reduce costs and development time on software development projects has often gone unfulfilled” [15]. Generally, any delayed sub-project, e.g. COTS-based with negative TSR, can lead to a delayed market entry of the product, especially if the respective sub-project is located on the project’s critical path. This, in turn, can lead to substantial loss of profit. In order to tackle such kind of economic pitfalls, one direction for future work could be the full exploitation of the economic indicators in relation to the evaluation depth in a CAP project. By continuously monitoring ROI, CE, and TSR jointly with evaluation depths, and using them as stopping criteria, it could be possible to manage COTS assessments and assure positive indicators values and sufficient evaluation depths, i.e. making a sound trade-off between cost and “quality” (evaluation depths).

Moreover, future work will also concentrate on developing models to explain the cause-effect mechanisms of factors influencing the cost and quality of COTS assessments.



## 6 References

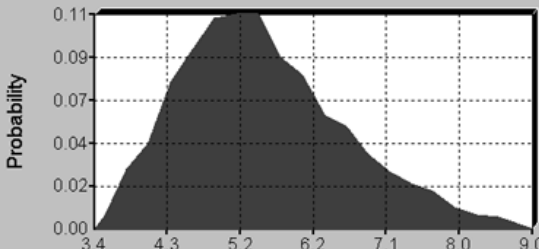
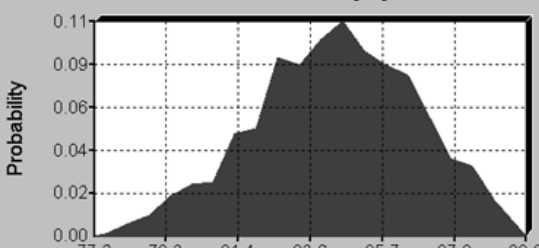
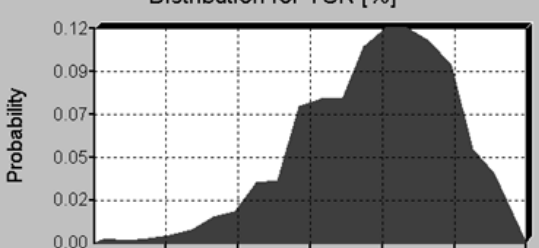
- [1] C. Abts, B. Boehm, E. Bailey Clark, "COCOTS: a COTS software integration cost model – model overview and preliminary data findings", *Proceedings of the 11<sup>th</sup> ESCOM Conference*, Shaker, Maastricht, 2000, pp. 325-333.
- [2] B. Boehm, "Software Engineering Economics", *Prentice-Hall*, Englewood Cliffs, 1981.
- [3] B. Boehm, "A spiral model of software development and enhancement", *IEEE Computer*, May 1988, pp. 61-72.
- [4] L. Briand, C. Differding, D. Rombach, "Practical Guidelines for Measurement-Based Process Improvement", *Software Process Improvement and Practice* 2(4), 1996, pp. 253-280.
- [5] L. Cohen, "Quality Function Deployment. How to Make QFD Work for You.", Addison-Wesley, Reading, 1995.
- [6] H. Erdogmus, J. Vandergraaf, "Quantitative Approaches for Assessing the Value of COTS-centric Development", *Proceedings of the 6<sup>th</sup> International Software Metrics Symposium (METRICS)*, IEEE, 1999, pp. 279-290.
- [7] N. Fenton, S. Pfleeger, "Software Metrics – A Rigorous & Practical Approach", Thomson Computer Press, London, 1996.
- [8] ISO/IEC 9126 Standard, "Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for their Use", International Organisation for Standardisation (Ed.), Geneva, 1991.
- [9] J. Kontio, G. Caldiera, V. Basili, "Defining Factors, Goals, and Criteria for Reusable Component Evaluation", *CASCON'96 Conference*, Toronto, 1996.
- [10] J. Kontio, "A Case Study in Applying a Systematic Method for COTS Selection", *Proceedings of the 18<sup>th</sup> ICSE*, IEEE, 1996.
- [11] N. Maiden, C. Ncube, "Acquiring COTS Software Selection Requirements", *IEEE Software*, March/April 1998, pp.46-56.
- [12] M. Meyer, J. Booker, "Eliciting and Analyzing Expert Judgement – A Practical Guide", *Knowledge-Based Systems (vol.5)*, Academic Press, London, 1991.
- [13] G. Nemhauser, L. Wolsey, "Integer and Combinatorial Optimisation", Wiley & Sons, New York, 1988.
- [14] M. Ochs, D. Pfahl, G. Chrobok-Diening, B. Nothhelfer-Kolb, "A COTS Acquisition Process: Definition and Application Experience", *Proceedings of the 11<sup>th</sup> ESCOM Conference*, Shaker, Maastricht, 2000, pp. 335-343.
- [15] S. Polen, L. Rose, B. Philips, "Component Evaluation Process", Software Productivity Consortium, SPC-98091-CMC, Herndon, 1999.
- [16] S. Polen, L. Rose, B. Philips, "Component Evaluation Process – Appendix G: CEP in ETVX Notation", Software Productivity Consortium, SPC-98091-CMC, Herndon, 2000.
- [17] T. Saaty, "The Analytic Hierarchy Process", McGraw-Hill, New York, 1990.
- [18] R. van Solingen, E. Berghout, "The Goal / Question / Metric Method", McGraw-Hill, London, 1999.

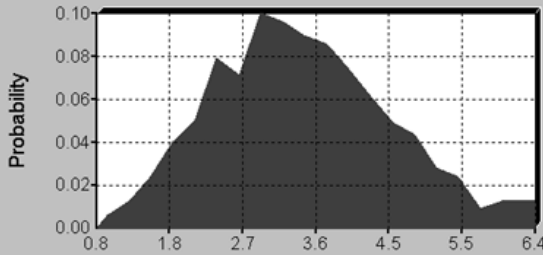
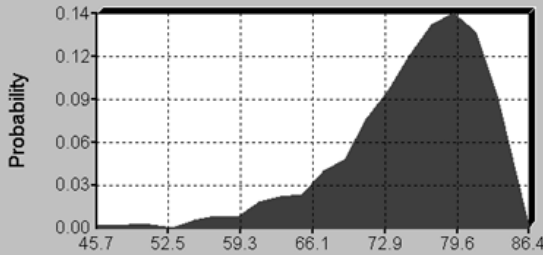
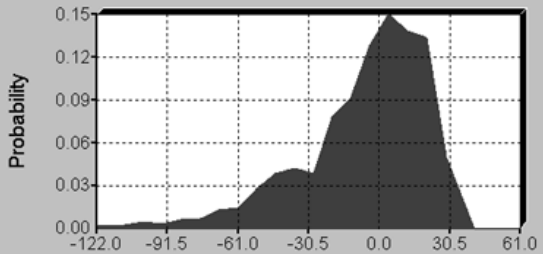
- [19] P. Spector, "Summated Rating Scale Construction", Sage, Thousand Oaks, 1992.
- [20] Taweel, P. Brereton, "Evaluating Off-The-Shelf Software Components: A Repository Selection Case Study", *Proceedings of EASE99*, Keele, 1999, p. 12.
- [21] J. Voas, "The Challenges of Using COTS Software in Component-Based Development", *IEEE Computer*, June 1998, pp. 44-45.
- [22] J. Voas, "COTS software – the economical choice?", *IEEE Software*, March 1998, pp.16-19.
- [23] H. Vollmuth, "Unternehmensstruierung mit Kennzahlen", Vahlen, Munich, 1999. (in German)
- [24] D. Vose, "Quantitative Risk Analysis: A Guide to Monte Carlo Simulation", John Wiley & Sons, Chichester, 1996.
- [25] R. Yin, "Case Study Research – Design and Methods", Sage, Thousand Oaks, 1994

## 7 Appendix A. Monte Carlo Simulation Results

Figure 7 shows the distributions for ROI, CE, and TSR based on expert opinion [12] using Monte Carlo simulation [22] for both case studies. The setting was 1,000 iterations per simulation employing Latin-Hypercube sampling [22] .

Figure 7 Monte Carlo simulation results: Values and graphics for illustration.

Case Study 1 Simulation Graphics	Associated Data and Information	
<p>Distribution for ROI</p>  <p>Function of development approach: ROI</p>	ROI: Case Study 1	
	Minimum: 3.38	Mode: 4.84
	Mean: 5.54	StdDev: 1.04
	Maximum: 8.96	
<p>Distribution for CE [%]</p>  <p>Function of development approach: CE</p>	CE: Case Study 1 [in %]	
	Minimum: 77.19	Mode: 83.87
	Mean: 84.32	StdDev: 2.41
	Maximum: 89.96	
<p>Distribution for TSR [%]</p>  <p>Function of development approach: TSR</p>	TSR: Case Study 1 [in %]	
	Minimum: 49.97	Mode: 70.87
	Mean: 67.15	StdDev: 4.41
	Maximum: 76.33	

Case Study 2 Simulation Graphics	Associated Data and Information		
<div><p>Distribution for ROI</p><p>Function of development approach: ROI</p></div>	ROI: Case Study 2		
	Minimum:	0.84	Mode: 2.85
	Mean:	3.40	StdDev: 1.08
	Maximum:	6.37	
<div><p>Distribution for CE [%]</p><p>Function of development approach: CE</p></div>	CE: Case Study 2 [in %]		
	Minimum:	45.70	Mode: 81.45
	Mean:	75.76	StdDev: 6.59
	Maximum:	86.44	
<div><p>Distribution for TSR [%]</p><p>Function of development approach: TSR</p></div>	TSR: Case Study 2 [in %]		
	Minimum:	-121.98	Mode: 10.62
	Mean:	-6.27	StdDev: 26.71
	Maximum:	40.78	

# Document Information

Title:	A Method for Efficient Measurement-based COTS Assessment and Selection – Method Description and Evaluation Results
Date:	December, 2000
Report:	IESE-055.00/E
Status:	Final
Distribution	Public

Copyright 2000, Fraunhofer IESE.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means including, without limitation, photocopying, recording, or otherwise, without the prior written permission of the publisher. Written permission is not needed if this publication is distributed for non-commercial purposes.