IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP5088

Identification Card for COTS-C Use and Development

Raphaël Michel
LIUPPA, Université de Pau, Avenue de l'Université , BP 1155, 64000 Pau Cedex, France

Philippe Roose
LIUPPA, Université de Pau, Avenue de l'Université , BP 1155, 64000 Pau Cedex, France

Franck Barbier
LIUPPA, Université de Pau, Avenue de l'Université , BP 1155, 64000 Pau Cedex, France

ABSTRACT

The increase in applications and the accompanying presence of more and more external experts prevents enterprises from remaining the master of their security. Because of this problem a new development in software appeared. Our objective in writing this draft is to overcome these structural problems with COTS-C identification Cards. The card will simultaneously facilitate the development and use of COTS-C (its research, its maintenance, its tests, its integration, …).

INTRODUCTION

The use of Components Off The Shelf (COTS) allows for the realization of application by assembly. It's not necessary to develop the all parts off application but if the developer use a Components Off The Self, he must use a new process of development [7] (selection, qualification, assembly and update). We are interested in the design, and specifically in the phase of COTS acquirement as in [15][26]. This article describes the comfortable integration of heterogeneous COTS-C in software.

According to Basili and Boehm [1] [2] much time and effort is needed for the integration of COTS-C. Besides, versions of COTS-C renew quickly [1], some have event produced formulas allowing to estimate the necessary efforts to use COTS-C.

However, is it possible to follow a method using the works of all actors? And, is it possible to reduce the necessary efforts to use COTS-C? If yes, then, how to select, choose and assemble products if the integration efforts are considerable and the renewal rates high while at the same time preserving the objectives of the developer of COTS-C and its basic application.? We propose a methodology associated with an *identification card* which facilitates the user or the designer of the COTS-C applications..

METHODOLOGY

We define 4 phases for these activities:

- selection of the COTS-C associated to a phase of needs,
- identification of the constraints in integration and cost estimates for the integration,
- coding of the" glue ware ", the interfaces permitting the integration and the assembly,
- spreading in the system targets and tests

In these different phases, their is a compromise between several criteria. [18] Presents the COTS-C solution as a compromise between the:

- Technic: feasibility of the integration of possible solutions,
- Economy: financial feasibility by the assessment of the integration costs,
- Strategy: to satisfy present and future needs while taking into account the technical, political and legal aspects.

In our observation, the technical and strategic phases are relative to the phases of selection and evaluation of costs. [13] and [18] insist on the notion of compromise between these phases. According to [18], "the use of components of COTS-C type is not an universal solution ". For this, cost assessment formulas do not exist. They use measures or valued parameters by the teams doing the integration of the components. Several criteria are identified thus.

Criteria and measures of COTS

COCOMO [2] uses existing models of calculation of the cost (C) and defined by:

$$C = CV * AC * IS * (CS + CC)$$

The meaning of these parameters:

- CV = Component Volatility (number of versions of the COTS-C on the life-cycle of the project)
- AC = Architectural Coupling (number of components with which exits an interface or a link)
- IS = Interface Size (number of points of entry, procedures, functions and other methods used to reach the COTS-C while using a coefficient according to the number of arguments)
- CS = Cost of Screening (masked costs of the COTS-C and other components with which it is linked)
- CC = Cost of making Change (cost of the changes of all components overlapped)

In this example and in [19], parameters of the same type exist:

- Measurable or quantitative (CV, AC, IS) and
- Assessable or qualitative (CS, CC)

Here, we are interested more especially in the measurable parameters, in the parameters obtained after an analysis, a research or tests phase.

So for the calculations of evaluations of the efforts (cost due to the evolutions) [1], it is necessary to a use of COTS-C starts with the identification of the source code, ports and interfaces of all other ways allowing to connect or to exchange with COTS-C. According to COCOMO II [2], we encounter some difficulties particularly on parts corresponding to the documentation, the development of the" Glue Code" (or the code necessary to the integration of the different components) and to the integration of the COTS-C in the target application.

Firstly, we are interested in the information needed to select and chooses COTS. We get this information from present attributes [12] [10] [6] [23] [19]. They permit:

- The classification, that is to say the identification of the products to group in three categories [10] [19]
  - The Architectural Level [10] [19] [12] or the type of architecture with as examples: centralized, customer server, n-tier, etc.,
  - Product Kind [10] [19] or how to characterize the product for example: executable, standard, service,
  - Life-cycle Phase [10] [19] or how to define in cycle of life when the product is used: development, execution for example.
- The characterization, that is to say the specification of the COTS-C [12] [19] according to 4 categories ,
  - Source (the origin of the product)
  - Customization, that is to say how one can use the COTS in a solution [12] : it is the minimal modifications of type, define or/ and the existence of documentation interfaces, API, OO interfaces,…,
  - Bundle, that is to say the information on what is delivered and under what conventions [12] : it is the package with source code, DLL,… or / and the size of products classified in 4 categories (Small - medium - large - huge)
  - Role, that is to say the role of the product during its use. For example: the horizontal or vertical functionalities, or an architecture (OS, middleware, support…)
- the assessment, that is to say the easy comparison between the products,
  - These are the attributes whose values are quantifiable.
  - The examples of attributes (without being exhaustive) are: the price, the type of license, the size in byte, the memory occupation, etc.
- The quality, that is to say the generic model. It is the basis to specify the requirement of the quality and to value this quality.

According to [6] [10] [12] [19], their attributes permit to recover a set of information. They are necessary, but non sufficient, to realize a selection and a first level of evaluation. It is mainly in this part that our contribution is located.

Proposition of contract

This notion of contract is defined in [4] by Meyer and Dowson [25] [26] [24] as:

- the relation between one and several parts,
- the necessary negotiation before the signature of the contract,
- the normative and measurable description of the definite behaviors,
- the respect of the integrity of the contract. It cannot be modified without common agreement of the different parts

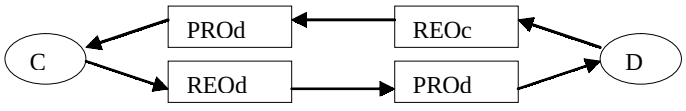
The use of a contract between two interfaces is seen as the specifications of the provides and requires parts.

We take the example provided in [4]. Different works exist when showing the contract between two components C and D. These contracts:

- PROd and REQc, correspond to the" PROvides parts" of D and the" REQuires parts" of C,
- PROc and REQd, correspond to the" PROvides parts" of C and the" REQuires parts" of D,

Each «PROvides parts " and "REQuires parts" can be listed into an attribute. They become the knowledge of the different access points (PROd, PROc, REQc, REQd) necessary to the use of the functions of D by C and reciprocally.

Figure 1 : Contract and reciprocal obligation



Therefore this information, identified in attributes, is:

- Provided by designers,
- Researches by selection, evaluations of the costs and development teams,
- Used by the set of the teams working with COTS-C.

Therefore we propose a structure permitting to catalogue and to define the set of this information, so that every team of the different phases of the development process with COTS-C can benefit from information of all teams. So when one transmits a team's information to other teams during the progression of the process of development, we enhance the process of development and insure the consistency of the information. This is due to the identification card of the COTS-C.

Here after, we present the principle of the identification card, its creation, and use.

ID CARD

Principle

The identification card of COTS-C is an electronic document containing information about the COTS and its use. This document permits to identify distinctly a COTS-C. Different actors of the development process with COTS-C will have the possibility to inform the identification card or to collect different information. These actors of the development process with COTS-C are the set of persons who define, conceive and use COTS-C. They will have, according to their authority and the phase of development, the authorizations to modify or read data contained in the id card.

We use as a base, the life cycle defined in [7] , and we add the actors of the creation of the COTS-C. This cycle is modified; it is composed of the actors (in regard to UML):

- of the creation of the components,
- of the qualification of the components [7]
- of the adaptation of the components [7]
- of the assembly [7]
- and of the update [7]

Their contribution to the identification card is important because their role is to inform regarding their products.

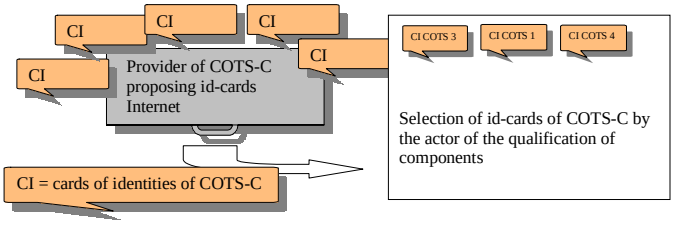
Information is obtained by the actors of the qualification of the COTS-C by identification cards. We can compare the information contained in the identification cards. Id-cards cards have to allow the identification of the product, of the source, of the role or the utility, and its characteristics.

Other information can be added to enrich the description as for example: the interfaces, procedures of tests, technology, etc.

If the structures and attributes used to describe information are identical to all id-cards, it will be easy to compare them. So, we need a rigorous and expandable formalism using the XML language. It will permit actors to manipulate and study the information.

The philosophy of XML consists in separating the data/documents (the XML file) of the process/presentation. A given document will be, at the creation, tagging according to the semantics and regardless of its future restitution (paper, screen or other). This generic aspect permits:

- A very big accessibility (the same XML document can be displayed on the Web, used in a SGBD, etc.)
- And a very big durability/reuse (the document won't become obsolete with the evolution of computer techniques; it will be able



to fully or partially incorporate itself in different documents, and be used by all applications...).

It is especially important then according to the points of view of the different actors of the development with COTS-C, for the needs won't be the same. For example, the applications used for the requests of selections, the treatments of the data and the calculation of the costs will be different from one user to an other. Nevertheless, a common base exists. They all need:

- Informations (List non exhaustive):
  - Architecture type, (customer server, n-tier, etc.)
  - Inputs and Outputs (« PROvides parts » et « REQuires parts »),
- To share data with other actors of the development process with COTS-C as for example the "PROvides parts "and" REQuires parts ". Their modification can have some influences on the final behavior of the application.

The idea is to use the common information so that each can benefit from the work of the other. When the new versions of the COTS-C arrive in the final application, one will use this common information to mark the impacts of the evolution of the COTS-C more easily: comparing the XML files of description for example. It is, for example, the interfaces and their characteristics identified during the phase of selections and used in the phase of development that will be able to be compared between the news and the former version.

Our proposition is based mainly on a structure permitting:

- To receive the information,
- To identify and to provide a number of version for the information,
- To make some requests on this information,
- To permit some treatments on these information.

The objective is:

- To gather the useful quantitative information for the different phases of the development process with COTS-C,
- To reduce processing time and treatment time
- To centralize the common information and to make that information identical to the different actors.
- To fill the identification card with the patterns for facilitating the use of the COTS-C.

We propose in the following chapter an enabling structure for this.

ID CARD STRUCTURE

We call this structure" the informational unit". It must permit:

- The addition and gathering of useful information,
- to describe this information with the use of definitions,
- to modify the associated values,
- to propose a list of values,
- to give rules to allocate values if needed

Take a proposition on the characterization of the COTS-C presented in [12]. This proposition defines attributes and possible values permitting to characterize the COTS-C. Several authors regroup in categories these attributes. For example a few attributes and categories:

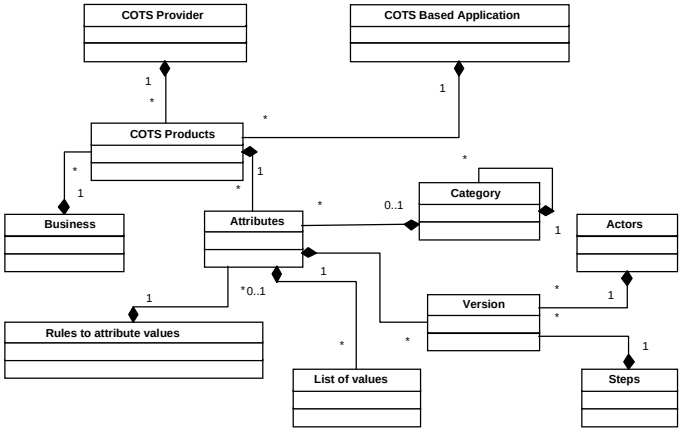
From this example we propose that the supplier of COTS-C defines a set of information. Each attribute of an informational unit will provide information. It can belong to a category which permits to group some information together. In the previous example there are two data (origin and cost) concerning the "source" of the product. The applications with COTS-C will be able to use this information relative to each product while reading the information carried by the attributes.

We propose the diagram of the identification card to the UML formalism under the format of a diagram of classes.

Attribute name	Definition	Value	List of values
Origin	Where the source code comes from	Exclusively from a list	Internal External existence External development Commercial release Independent product
Cost and proprietary	License	Exclusively from a list	Acquisition License Free

Category	Definition	A list of informational units
Source	Where the product comes from	Origin Cost & Property
Customization	How much the product can or should be customized	Required Modification Possible modification Interface
Role	What is the intrinsic role the product can assume in the final system	Functionality Architectural level

Figure 2 : Class Diagram of id-card using UML formalism



From this diagram, we get a DTD (XML) that represents the model of the identification card and from which one will create some identification cards henceforth comparable: Same structure, same formalism and same representation. A "massive" use of XML will permit an easy research of the id-cards existing in UDDI-like directories.

THE USE OF ID-CARD

The card may be used at all stages of the development with COTS-C. Let's see how.

Various steps of use:

- the creation of the card,
- the information of the card with information,
- the extraction of the information from the identification card,

The creation of the card must be done by:

- The creators or the seller of the COTS-C will provide basic information for purchasers/users.
- The selectors of COTS-C. If the attributes of the different cards do not correspond with needs, the selectors of COTS-C create his own id-card to identify the same information from other id-cards corresponding to the various selected COTS-C.
- The other actors can also create attributes relative to needs.

During the creation of the id-card, the actors use the defined structure. They create the attributes that are necessary to identify information and to assign it a value. The declared attributes correspond to information that will be useful to do their work.

Thereafter, the card must be filled by:

- The creators or the sellers of COTS-C. They assign values to all fields of the id-card corresponding to all created attributes.

- The selectors of COTS-C. They also have the possibility to fill the fields of the id-card corresponding to the attributes that they have created. They can also import the id-cards from the creators or sellers of COTS-C. They can gather the set of the corresponding information for the COTS-C that they are working with.
- The other actors can also fill this identification card. They will complete it with values necessary to their activity.

Each actor can be identified as the one who added or updated information at a precise step. It will permit follow-up and memorization of the evolution.

CONCLUSIONS

The objective is to be able to identify information vital to the realization and to the use of a COTS-C from the attributes of the Id-card.

Thus, we created a structure based on an "informational unit". It provides the attributes representing usable information for the actors of the qualification of the components. The value of this attribute informs these actors about the characteristic of a part of the COTS-C. All attributes have an associated definition.

The set of these "informational units" can be grouped in "categories" to sort them out or to group them. Requests made on these identification cards must permit the extraction and comparison of desired data.

We hope to facilitate the work of selection and comparison of the COTS-C; thus and we will establish a guide for the selection of the COTS-C for design and development phases with COTS-C.

We are currently implementing the id-card using the XML format. The uses of a DTD permits for the modification and compatibility of the ID Cards: same structure, same formalism, and same representation. Moreover, a "massive" use of XML facilitates research if the identification cards exist in UDDI-like directories.

Thereafter, we will be able to use the identification card in a collaborative process of development. It will permit testing the id-card and completion with the attributes necessary for design and diffusion phases. The work of collaboration between actors must permit enrichment of the identification card with the attributes necessary to this task and to propose the patterns for collaboration.

REFERENCES

[1] C.Abst, B.Boehm, E.Clark. "COCOTS: A COTS Software Integration Lifecycle Cost. Model - Model Overview and Preliminary Data Collection Findings", Technical report. USC-CSE-2000-501, USC Center for Software Engineering, 2000.

[2] C.Abst, B.Boehm „ COTS/NDI Software Integration Cost Estimation & USC CSE COTS Integration Cost Calculator V2.0 guide, Rev 1.0, 30 sept 1997.

[3] C. Abst, B Boehm, "COTS Software Integration Cost Modeling Study", USAF Electronic Systems Center, Massachusetts, University Of Southern California, June 1997.

[4] F. Bachmann, L. Bass, C. Buhman, S. Comella-Dorada, F. Long, J. Robert, R. Seacord, K. Wallnau; "Vol II : Technicla Concepts of Component Based Software Engineering"; Pittsburg, PA 15213 3890; CMU/SEI 2000 TR 008; ESC TR 2000 07; May 2000.

[5] V.Basili, B.Boehm. "COTS-Based Systems Top 10 List" IEEE Computer 34(5), May 2001, pp 91-93

[6] Manuel F Bertoa, A Vallecillo, "Quality Attributes for COTS Components" in Proceedings of the 6th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2002), (Malaga, Spain), June 2002..

[7] D.Carney. "Assembling Large Systems from COTS Components: Opportunities, Cautions, and Complexities". SEI Monographs on Use of Commercial Software in Government Systems, Software Engineering Institute, Pittsburgh, USA, June 1997.

[8] D.Carney, F.Long. "What Do You Mean by COTS?". IEEE Software, March/April 2000, pp. 83-86.

[9] F Duclos, J Estublier, P Morat, « Describing and using non fonctional aspects in Component Based Applications » AOSD 2002, Enschede.

[10] L. Jaccheri, M Torchiano; « Classifying COTS Products"

[11] Karl RPH Leung, Hareton KN Leung, "On the efficiency of domain-based COTS product selection method", Information and Software Technology 44 (2002) pp703-715.

[12] Maurizio Morisio, Marco Torchiano. Definition and classification of COTS: a proposal, Accepted at ICCBSS, Orlando (FL) February 4-6, 2002

[13] M Morisio, CB Seaman, VR Basili, AT Parra, SE Kraft, SE Condon. "COTS-based software development : Process and open issues". The journal of Systems and Software 61 (2002) 189-199

[14] National Archives and Records Administration, Archival Research Catalog, Permissions & privileges; Universal Hi-Tech Development, Inc. 1383 Picard Drive, Rockville, MD 20850; Feb 2002.

[15] T.Oberndorf. "COTS and Open Systems – An Overview ". 22 septembre 2000, http://www.sei.cmu.edu/str/descriptions/cots.html#ndi

[16] M. Ochs, D. Pfahl, G. Chrobok-Diening, B. Nothhelfer-Kolb; "A COTS Acquisition Process : Definition and Application Experience";

[17] Santiago Cornella-Dorada, John Dean, Edwin Morris, Patricia Obendorf, A process for COTS software Product Evaluation, CBSS 2002, pp 86-96, NRC 44925.

[18] SEI, COCOTS – Constructive COTS, http://sunset.usc.edu/research/COCOTS/index.html

[19] M Torchiano and L Jaccheri, "Assessment of Reusable COTS Attributes", ICCBSS, 10-12 feb 2003.

[20] G Valetto, G E Kaiser, IEEE Seventh International Workshop on Computer Aides Software Engineering, July 1995, pp 40-48.

[21] MR Vigder et J Dean « COTS Software Integration : State of the Art », Technical Report NRC 39190, 1996.

[22] MR Vigder et J Dean. Maintaining a COTS-Based Systems : Proceedings of the NATO information systems Technology Panel Symposium on Commercial Off the Shelf Products In defence Applications, Brussels, Belgium. April 3-5, 2000. 6 pages. NRC 43626.

[23] S. Yacoub, A. Mili, C. Kaveri; "A Model for Certifying COTS Components for Product Lines"; Software Product Lines Conference, Denver, Colorado, Août 2000.

[24] Dowson, "ISTAR and the Contractual Approach", 287-288, IEEE,Avril 1987

[25] Meyer, "Applying 'Design by Contract.'", Computer 25, pp 40-51, 10 octobre 1992

[26] Meyer, Object-Oiented Software Construction, 2 ed London, UK : Prentice-Hall International, 1997.

[27] Meyers, B., and Oberndorf, P., 2001. Managing Software Acquisition Open Systems and COTS Products, Addison-Wesley.

[28] http://www.uddi.org