A COTS-Aware Requirements Engineering Process: a Goal- and Agent-oriented Approach

Lawrence Chung and Kendra Cooper
University of Texas at Dallas
MS 31 P.O. Box 830688 Richardson, TX, USA 75083-0688
chung@utdallas.edu kcooper@utdallas.edu

Abstract. The goals of developing systems better, faster, and cheaper continue to drive software engineering practitioners and researchers to investigate software engineering methodologies. In requirements engineering, the focus has been on modeling the software engineering process and products for systems that are being built from scratch. As the size and complexity of systems continues to grow the use of commercial off the shelf (COTS) components is being viewed as a solution. Effective use of COTS components, however, requires a systematic approach that provides both a set of concepts for modeling the subject matter and a set of guidelines for using such concepts. In particular, the process needs to recognize and address the people oriented problems including the identification and resolution of conflicting goals, bridging the gaps between stated requirements and 'approximately fitting' components while still satisfying the customer. In this paper, we present a goal and agent oriented requirements engineering process model that explicitly addresses the use of COTS components. More specifically, we present (part of) our model for a COTS-Aware Requirements Engineering (CARE) process and illustrate it using a Digital Library System.

INTRODUCTION

The goal of developing systems better, fast, and cheaper continues to drive software engineering practitioners and researchers to investigate software engineering methodologies. In requirements engineering, the focus has been on describing the functional requirements for systems that are being built from scratch. As the size and complexity of systems continues to grow the use of commercial off the shelf (COTS) components is being viewed as a solution to this problem. Effective use of COTS components, however, requires a systematic methodology that would provide both a set of concepts for modeling the subject matter and a set of guidelines for using such concepts. In particular, the methodology needs to recognize and address the people oriented problems including the identification and resolution of conflicting goals as well as bridging the gaps between stated requirements and 'approximately fitting' components while still satisfying the customer.

Our work, the COTS-Aware Requirements Engineering (CARE) approach, is focused on creating and modeling a requirements engineering approach that explicitly supports the use of commercial off the shelf (COTS) components. It is both goal oriented as in (Anton 1998, Chung 2000, van Lamsweerde 2000), and agent oriented (as in Yu 1994). The process is intended to assist the requirements engineer, not to replace their intelligence and experience. There are two models that describe the CARE approach: a process model and a product model. The process model is the focus of this work. Due to space limitations, here we present only part of the process model and demonstrate its use with a Digital Library System application. A more complete descripton is available in (Chung 2001).

In related work, the Rational Unified Process (*RUP*) is an object oriented software engineering technique (Jacobson 1999), which is based on four phases (transition, construction, elaboration, and inception) and five core workflows (requirements, analysis, design, implementation, test), and uses the unified modeling language (UML). In UML, a COTS component is represented as a component, a physical and replaceable part of the system that provides a set of interfaces and typically represents the physical packaging of classes, interfaces, and collaborations (Krutchen 1998). The Model Based Architecting and Software Engineering (*MBASE*) approach considers four types of models: success, process, product and property (Boehm 1998) and is consistent for use with COTS components (Boehm 2000). MBASE uses four guiding principles to develop value-driven, shared-vision-driven, change-driven, and risk-driven requirements. The Procurement Oriented Requirements Engineering (*PORE*) technique supports the evaluation and selection of COTS components (Ncube 1998). The PORE process model identifies four goals that need to be performed in a thorough COTS selection process: acquiring information from the stakeholders, analyzing the information to determine if it is complete and correct, making the decision about product requirement compliance if the acquired information is sufficient, and selecting one or more candidate COTS components.

Research that considers the long-term maintenance impact of using COTS components (Carney 2000, Dean 1999) is also being investigated.

This paper is organized as follows. Following the introduction, we present the CARE approach and illustrate its use with an example. These are followed by a section for the conclusions and future work.

THE CARE APPROACH

The CARE approach draws upon the good ideas available in current RE methodologies including RUP, MBASE and ACRE/PORE. The aim is to complement and extend these methodologies. The scope of the process model described here is restricted to the requirements phase of the software development lifecycle. The description does not, for example, consider the design, implementation, testing, and maintenance phases.

The CARE approach is being developed using a set of examples for a Digital Library System (Chung 2001a, Chung 2001b). In this description of the model, the customer is contracting the development of a large scale system and is involved in the development work. Variations of this model can be developed in the future to consider high volume, shrink-wrap product development.

Overview. The CARE approach is characterized as goal-oriented, agent oriented, knowledge based, and has a defined methodology, or process (refer to Figure 1). When the development of a system begins, the stakeholders' goals for the system under development are defined first. Our CARE approach is softgoal-oriented for the purpose of tradeoff analysis. For this, the nonfunctional requirement (NFR) framework (Chung 2000, Mylopolous 1992) is used which allows for a systematic approach to dealing with non-functional requirements. Using the NFR Framework, functional alternatives can be explored and represented through OR-dependencies.

Once defined, the goals are used to drive the development of the system. First, the goals are refined into system requirements. In turn, system requirements may be refined into software, hardware, or interface requirements. Currently, the latter refinement step is not considered in the CARE approach.

The CARE approach is agent oriented. The agents are the stakeholders for the system under development. An agent is intentional and possesses intentional properties including goals, beliefs, abilities, and commitments. An agent is also autonomous and makes decisions, choices, and depends on other agents to accomplish goals, complete tasks, or furnish resources. An agent can analyze its opportunities and risks in the various proposals and configurations for a system. The notation used to support this aspect of the CARE approach is drawn from the i* framework (Yu 1994), which describes goal and softgoal dependencies among agents and how they accomplish a goal in terms of subgoals, softgoals, and tasks. These concepts are embedded into the conceptual modeling language Telos (Mylopolous 1990), a descendent of RML (Greenspan 1994)- an object oriented requirements modeling language for functional requirements. As a result, i*

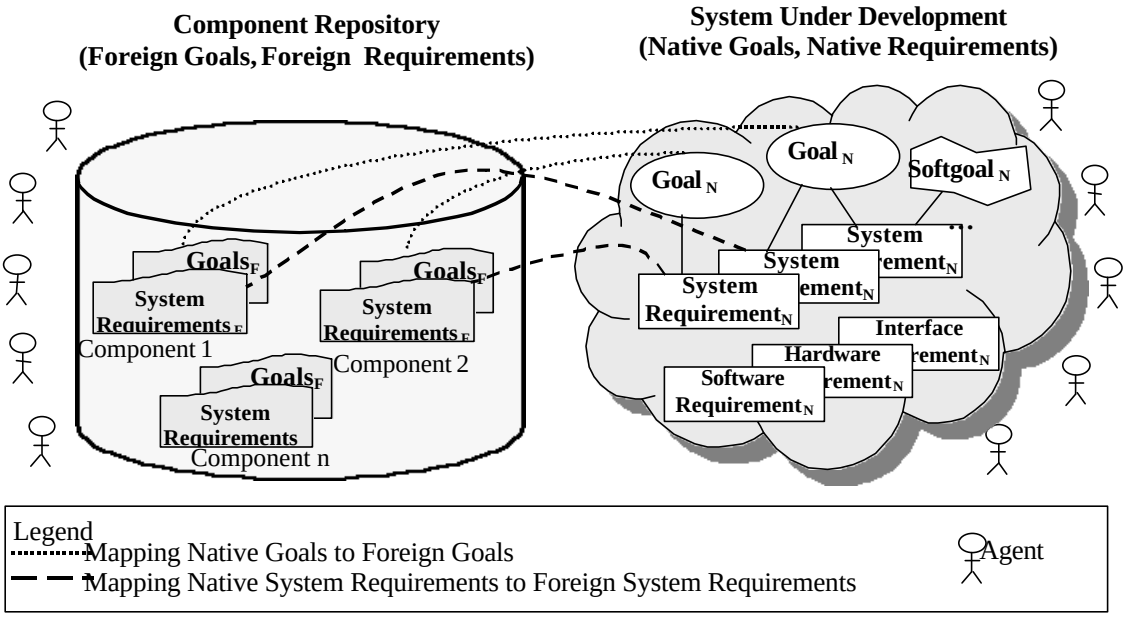

Figure 1. Mapping Native Goals and Requirements to Foreign Goals and Requirements

provides an extensible, object-oriented representational framework with classification, generalization, aggregation, attribution, and time.

The CARE approach has a defined process. An overview of the CARE process is described in IDEF0 (FIPS 1993) (refer to Figure 2). At a high level, the process has activities to define the goals, system requirements, software requirements, hardware requirements, and interface requirements (with COTS). Each of the activities corresponds to developing a product artifact for the system. The IDEF0 notation is activity oriented, and is suitable for describing a process, or methodology. Activities in the IDEF0 notation have inputs, mechanisms, control, and outputs. An activity is used to transform inputs into outputs. Mechanisms are the people or tools that perform the activity. Controls are used to constrain when the activity can be done. In CARE, the requirements engineer (hereafter RE) is the mechanism.

The CARE approach uses a knowledge base (repository) that is populated with descriptions of components (refer to Figure 3). These descriptions are at two levels of abstraction: goals and product specifications. The goals provide high level descriptions of the functional and non-functional capabilities of the component. The product specifications provide detailed descriptions of the functional and non-functional capabilities in addition to the system requirements (e.g., memory requirements) for the component. The components in the repository are called "foreign" in the sense that they are (initially) unrelated to the system under development. In the CARE approach the goals and requirements in the repository are called foreign goals and foreign requirements. The goals and requirements developed using the CARE approach (refer to Figure x) are called native goals, native system requirements, native software requirements, etc. When components are selected for use in the system under development, native goals are mapped to foreign goals and native requirements are mapped to foreign requirements.

The CARE Process Model. In this work, part of the process model is described in the i* framework, the NFR framework, and in the IDEF0 notation (refer to Figures 2,3,4).

The customer's overall goal is to receive a system that satisfices them (i.e., the customer is satisfied enough with the system to use it). Subgoals include receiving product development artifacts (product goals, system and software requirements) and product planning artifacts (quality plan, test plan, schedule, etc.). The customer's softgoals include receiving a system that is delivered on schedule, within budget, and with high quality.

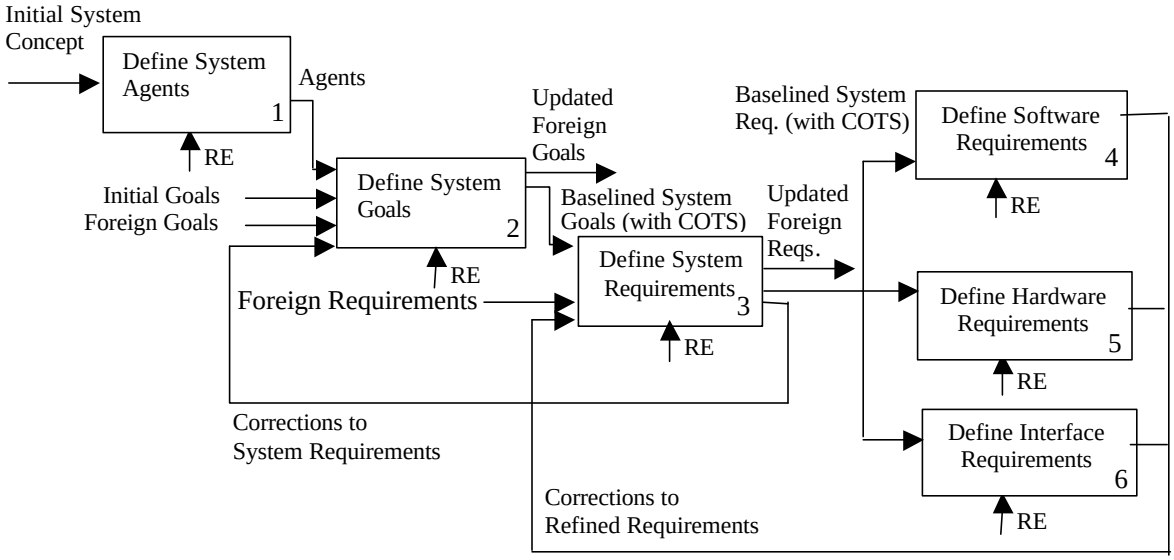

Figure 2. High Level View of the CARE Process

The RE depends on the customer to validate the system as it is being developed and provide (ideally) complete and correct information. The RE actor is decomposed using the Strategic Rationale Model (refer to Figure 3). The model describes how the RE can accomplish the goals of creating the baselined system goals and the baselined system requirements. The task of creating the Baselined System Goals can be decomposed into a set of subtasks. Some of these include to Identify Stakeholders and then Elicit, Analyze, Correct, Validate, Define COTS, and Baseline the system goals. The RE, an intentional agent, decides

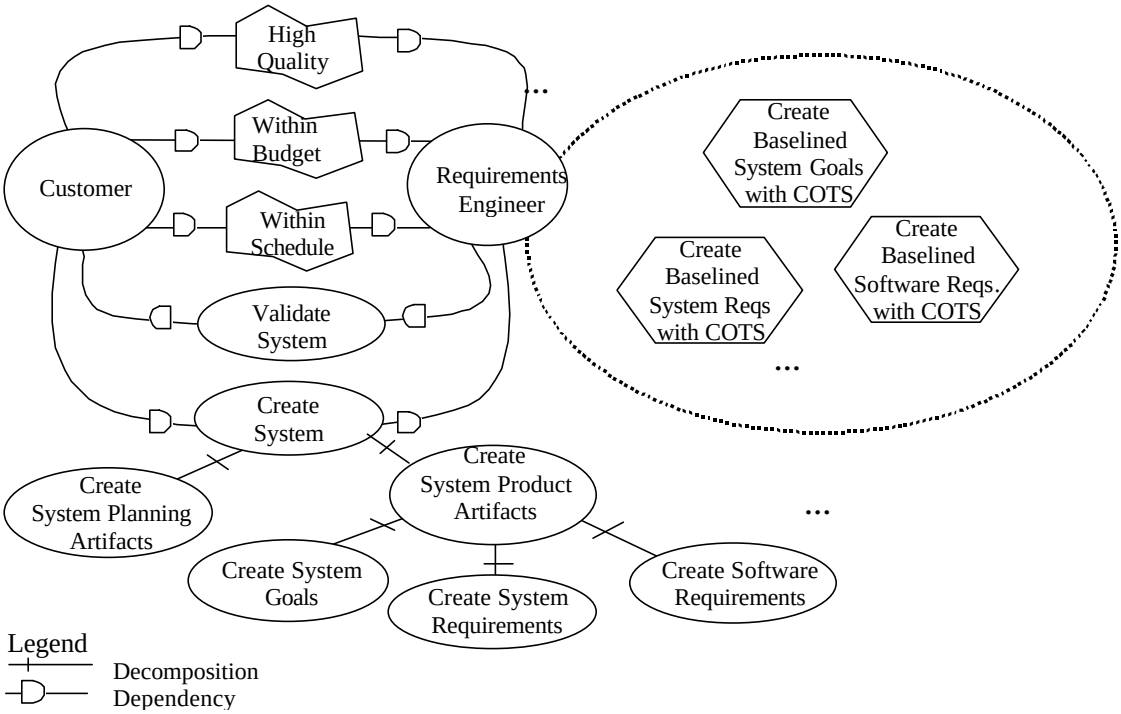

Figure 3. Strategic Rational and Dependency Models for CARE

when to use each of the subtasks to accomplish the goal of Creating the Baselined System Goals. The CARE approach uses terminology to distinguish between goals that are going to be developed from scratch ("native goals") and goals that may be implemented with COTS ("foreign goals").

The analysis subtask identifies the errors of commission (conflicting, incorrect, and redundant goals) and omission (missing goals). To characterize the relationships between goals, the NFR framework is used. In the NFR framework, the focus is on non-functional goals, or softgoals. It provides five rankings of the relationships between two softgoals: very positive (++), positive (+), neutral (no line, label), negative (-), and very negative(--). The relationships are defined informally as follows:

Softgoal 1 is *very positively* related to Softgoal 2: Softgoal 1 significantly helps to accomplish Softgoal 2.

Softgoal 1 is *positively* related to Softgoal 2: Softgoal 1 helps to accomplish Softgoal 2.

Softgoal 1 is *negatively* related to Softgoal 2: Softgoal 1 hurts the accomplishment of Softgoal 2.

Softgoal 1 is *very negatively* related to Softgoal 2: Softgoal 1 significantly hurts accomplishing Softgoal 2.

When characterizing the relationships, goals and subgoals are characterized in the same manner.

The RE corrects incorrect native goals, removes redundant native goals, and adds missing native goals, and negotiate conflicting goals. The stakeholders validate the native goals to ensure the RE has a common understanding of their needs and wants. Since goals are written at a very abstract level, they are open to (varying) interpretation.

The RE evaluates each goal and determines if it is a candidate for implementing with one or more COTS components. For each candidate, the RE performs a search on the repository that returns functional goal descriptions of the components that match the search criteria. The RE evaluates the results of the preliminary search and determines which of the components may be a possible match to the goal. The RE performs a search on the repository for the components that satisfy the preliminary match. Detailed descriptions of the functional goals and non-functional softgoals of these components are returned. The RE evaluates the results of the detailed search and determines which of the components is a close or exact match to the goals. The RE selects one or more components. The RE can add, update, or delete the goal descriptions in the repository as needed.

To negotiate conflicting goals, the RE may make a request to change a component's goal and/or change a goal for the system under development. To change the component, the RE sends a request to the vendor for a change to a component. For example, if a component's goal is described as high performance and high cost, the request may be to provide a component with moderate performance and moderate cost. To change the goal, the RE sends a request to the stakeholders for a change to a goal. Once the native and foreign goals are agreed upon, the set can be baselined. This part of the CARE process is depicted in Figure 4. Additional parts of the model are available in (Chung 2001a).

ILLUSTRATION

The CARE approach is being developed iteratively by working through a set of examples in a Digital Library System (DLS). The initial CARE process is based on the examples used to develop the system requirements (with COTS) for a database component and a communication component (Chung 2001a, Chung 2001b). In this work, we present another example in order to validate and refine the step in the process in which the system goals are defined. This example addresses the conflicting goals of ease of use, capacity, performance, and cost for the system.

Here, the digital library (DL) refers to the digital library organization as a whole. It may consist of people, software, hardware, and interfaces to external agents. The digital library system (DLS) refers to the goals that need to be met by the software system. The DLS stakeholders identified include: the librarians (Levels I and II), the system administrator, the borrowers, and the managers that authorize the contract for the system to be built. The system goals are available in Appendix A. In this example, we select five, initial goals to refine into system level requirements (with COTS) using our CARE process:

1. The DLS should be delivered with high quality, on time, and within budget
2. The DLS should be easy to use
3. The DLS should have fast performance
4. The DLS should be scalable
5. The DLS should be moderately priced

The first goal drives the use of COTS components. The use of COTS is viewed as a solution to the problem of how to develop systems that are increasingly complex with high quality, on time, and within budget. This goal needs to be refined in a quality plan, project schedule, and project budget and are not considered in the refinement process for technical requirements. The remaining goals are considered in the analysis task.

The RE analyzes the native goals by checking for errors (omission and commission) and refining the goals. The second goal is interpreted to mean that the CHI is easy to use for users with different levels of expertise (novice to expert). To accomplish this goal the CHI needs to be configurable and provide a variety of interface options (e.g., touch screen, voice recognition software, audio and video players, etc.). As

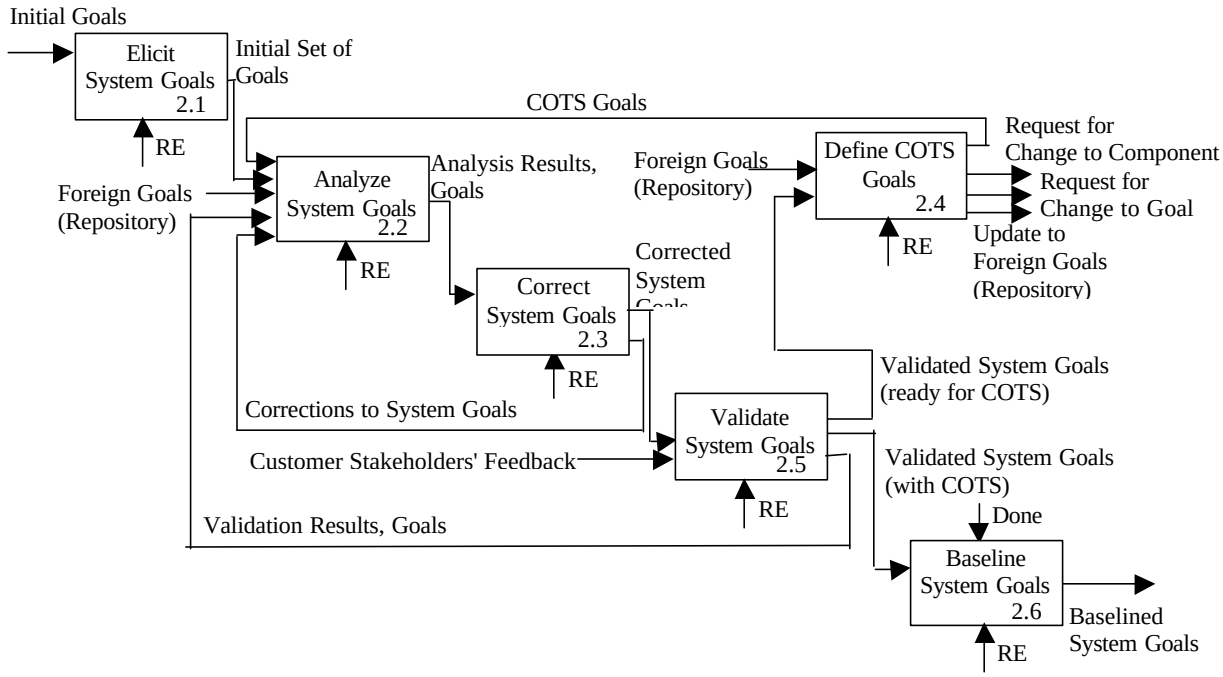

Figure 4. Define System Goals (with COTS)

a result, the goal is refined into the following goals and softgoals. The DLS should offer:

2a. a configurable CHI (goal)
2b. at least one CHI configuration so that it is easy to use for novices (softgoal)
2c. at least one CHI configuration so that it is easy to use for experts (softgoal)
2d. a variety of CHI options that include touch screen, voice recognition, traditional mouse and keyboard, audio and video players (goal).

The third goal is interpreted to mean that the performance of the DLS is fast even when the maximum number of users is loading the system. As a result, this goal is refined into the following:

3a. The DLS should have fast performance even when the system is loaded with the maximum number of users (softgoal)

The fourth goal is interpreted to mean that the library can be expanded to include new collections or additions to the collections. This goal is refined into the following:

4a. The DLS should be scaleable in order to accommodate new collections or additions to the current collections (softgoal).

The fifth goal is interpreted to mean that the system is neither very cheap (with few features) or very expensive (with a large number of features). The customer's perception of "moderate cost" needs to be clarified. With input from their procurement personnel, the goal may be refined to the following:

5a. The DLS should be moderately priced, with a budget of $100,000 (softgoal).

Now that the goals are refined, the RE needs to identify and characterize the relationships among the native goals and softgoals. In this example, the RE characterizes the relationships between the goals and softgoals (refer to Figure 5).

The RE corrects incorrect native goals, removes redundant native goals, and adds missing native goals. The conflicting goals need to be identified. Domain experts validate the interpretation and refinement of the goals. In this example, a request for comments is sent to the Digital Library community for feedback. The experts do not return any comments that apply to the goals of the system. The RE evaluates each goal and determines if it is a candidate for implementing with one or more COTS components. This evaluation and selection is based on the RE's experience. The RE selects the following goal as a candidate:

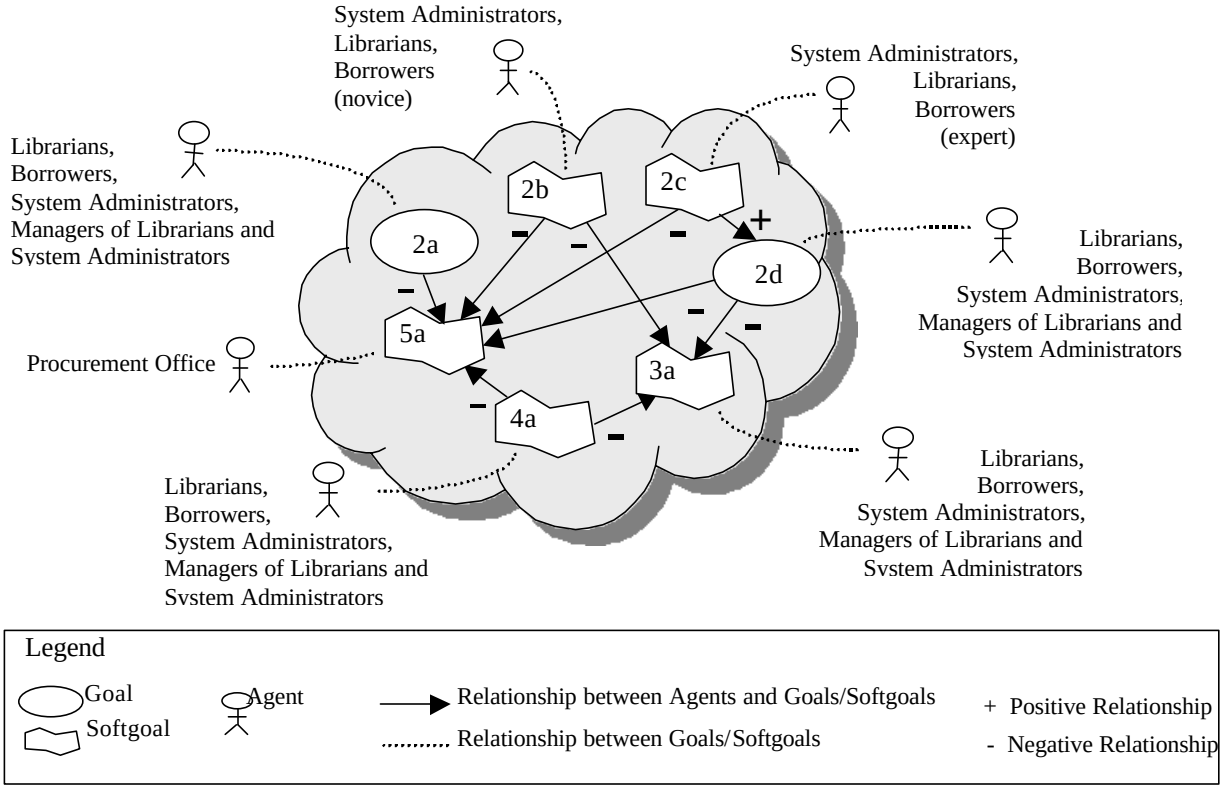

Figure 5. Relationships Among Native Goals and Softgoals

2d. The DLS should provide a variety of CHI options that include touch screen, voice recognition, traditional mouse and keyboard, audio and video players (goal).

The RE performs a search on the repository that returns descriptions of the components' functional goals that match the search criteria. In this example, the RE performs a preliminary search in the repository using keywords such as "touch screen", "voice recognition", "audio" or "video". Since the mouse and keyboard are standard equipment the RE chooses not to search for these.

We continue in this example with the results of the search for "audio" or "video". The goals for the following components are returned:

1. RealSystem Server 8, RealPlayer 8 and SDK by RealNetworks, Inc. should support 1.a) streaming and playback of audio and video, 1.b) multiple file formats.
2. Java Media Framework (JMF) by Sun Microsystems, Inc should support 2.a)The product should support adding audio and video to Java applications and applets, 2.b) streaming and playback of audio and video, 2.c) capturing audio and video, 2.d) multiple file formats, 2.e) transcoding multiple file formats, 2.f) extending the multimedia capabilities on the J2SE™ platform.
3. Java MFastView Plus, web browser plug-in by RealNetworks, Inc. should support 3.a) compatibility with a variety of web browsers, 3.b) opening and printing audio and video on a Web page, view files and e-mail attachments, 3.c) multiple file formats.

The RE evaluates the results of the preliminary search and determines which of the components may be a possible match to the stakeholder's goal. Based on the preliminary results, the RE determines that either of the first two components may be matches. The plug-in is rejected by the RE as a possible match.

The RE performs a search on the repository for the components that satisfy the preliminary match. The functional and non-functional goals (softgoals) of these components are returned. The non-functional goals are:

1. RealSystem Server 8, RealPlayer 8, and SDK should support the high quality presentation of audio and video.
2. Java Media Framework (JMF) should support the high quality presentation of audio and video.

The RE evaluates the results of the detailed search and determines that the non-functional softgoals of the components do not map to the softgoals of the system under development. After investigating this, the RE determines that the "quality" softgoal for the system under development is missing and needs to be added:

1. The DLS should present high quality audio and video to the user (softgoal)

Based on the descriptions available, the native goals appear to match the foreign goals of two components in the repository. In the absence of a complete specification, the RE may need to contact the vendor for additional product information before selecting one or both of the components if they both appear to match the goals of the system under development at the current (coarse-grained) level of refinement. After analysis and correction, the RE determines that the refinement step is complete and the goals are baselined.

CONCLUSIONS AND FUTURE WORK

We have presented part of our process model for a requirements engineering approach that explicitly supports the use of COTS components. Our model is agent- and goal-oriented, allowing for the representation and analysis of goal conflicts among the different stakeholders. An example is given to demonstrate the applicability of the model to a Digital Library System. The example describes some of our own experiences using the approach while more details can be found elsewhere (Chung 2001a). For example, it highlights some of the complexities involved in selecting COTS components, especially when the specifications available from vendors are not complete, hence an RE being unable to select a component without contacting the vendor for additional information.

There are a number of important aspects of the research to investigate. Our next step is to more fully use the process to develop the Digital Library System and also develop other applications in order to validate and improve the model. As the process model is validated and refined, the product model for the CARE approach also needs to be developed. Defining the product model which would allow the consistent use of a "mixed bag" of the various OO, AO, and GO ontologies and methodologies for CARE is expected to be challenging. We will also extend the approach to consider the impact over the software development lifecycle.

REFERENCES

Antón, A.I. and Potts, C., "The Use of Goals to Surface Requirements for Evolving Systems", Int. Conf. on Software Engineering, Kyoto, Japan, pp. 157-166,

19-25 April 1998.

Boehm, B. “Requirements that handle IKIWISI, COTS, and Rapid Change", IEEE Computer, Volume: 33 Issue: 7, July 2000, pp. 99 –102.

Boehm, B., Port, D., Abi-Antoun, M., and Egyed, A. “Guidelines for the Life Cycle Objectives (LCO) and the Life Cycle Architecture (LCA) deliverables for Model-Based Architecting and Software Engineering (MBASE)”, TR USC-CSE-98-519, USC-Center for Software Engineering.

Brownsword, L., Oberndorf, T., and Sledge. C.A., "Developing new processes for COTS-based systems", IEEE Software, Volume: 17 Issue:4, July-Aug. 2000 pp. 48 –55.

Carney, D., Hissam, SA.., and Plakosh, D., “Complex COTS-based software systems: practical steps for their maintenance”, J. of Software Maint.: Research and Practice, Nov.-Dec. 2000, vol. 12, no.6, pp. 357-76.

Chung, L., Nixon, B., Yu, E., and Mylopoulos, J., *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishing, 2000.

Chung, L. and Cooper, K., “A COTS-Aware Requirements Engineering (CARE) Process: Defining System Level Agents, Goals and Requirements”, TR UTDCS-23-01, Department of Computer Science, The University of Texas at Dallas, 2001.

Chung, L. and Cooper, K., “Towards a Model-based COTS-aware Requirements Engineering Process”, to appear MBRE '01, November, 2001.

Dean, J.C., “Ensuring the capability of COTS products”, Twenty-Third Annual International Computer Software and Applications Conference, 1999, pp. 96-97.

Federal Information Processing Standard (FIPS) Publication 183, Integration Definition for Function Modeling (IDEF0), December 21, 1993.

Greenspan, S., Mylopolous, J., and Borgida, A., “On formal requirements modeling languages: RML revisited”, 16th International Conference on Software Engineering, 1994, pp. 135-147.

Jacobson, I., Booch, G., and Rumbaugh, J., *The Unified Software Development Process*, Addison Wesley Longman, Inc., USA, 1999.

Kruchten, P., “Modeling Component Systems with the Unified Modeling Language”, International Workshop on Component-Based Software Engineering, 1998.

van Lamsweerde, A. and Letier, E., “Handling Obstacles in Goal-Oriented Requirements Engineering”, IEEE Transactions on Software Engineering, Special Issue on Exception Handling, Vol. 26, Sept. 2000.

Mylopoulos, J., Chung, L., and Nixon, B., “Representing and using nonfunctional requirements: a process-oriented approach” IEEE Transactions on Software Engineering, Vol. 18, Issue 6, June 1992, pp. 483-497.

Mylopoulos, J., Borgida, A., Jarke, M., Koubarakis, M., “Telos: Representing Knowledge about Information Systems”, ACM Trans. Info. Sys., 8 (4), pp. 325-362, October 1990.

Ncube C. and Maiden N, **“**Guiding parallel requirements acquisition and COTS software selection**”**, Proceedings of the IEEE International Symposium on Requirements Engineering 1999, pp. 133-140.

Yu, E., “Modelling Strategic Relationships For Process Reengineering”, DKBS-TR-94-6, The University of Toronto, Canada, December 1994.

APPENDIX A. The Initial System Goals

1. The DLS should be delivered with high quality, on time, and within budget
2. The DL should comply with current standards
3. The DL should be easy to use
4. The DL should have high availability
5. The DL should have fast performance
6. The users should be able to access a large number of diverse objects
7. The users should be able to search, browse, and retrieve objects quickly and efficiently
8. The librarians should be able to maintain the library quickly and efficiently
9. The librarians should be able to provide reference support quickly and efficiently
10. The administrators should be able to maintain users’ accounts quickly and efficiently
11. The DL should be scaleable
12. The DL should be secure
13. The DL should be inexpensive to operate
14. The DLS should be moderately priced

BIOGRAPHY

Dr. Lawrence Chung is an Associate Professor in the department of Computer Science at The University of Texas at Dallas. He received a Ph.D. in Computer Science from The University of Toronto, is on the editorial board of Requirements Engineering Journal, and is the principal author of "Non-Functional Requirements in Software Engineering".

Dr. Kendra Cooper is an Assistant Professor in the department of Computer Science at The University of Texas at Dallas. She received a Ph.D. in Electrical and Computer Engineering from The University of British Columbia. Her research interests include requirements engineering, formal methods, and the empirical assessment of software engineering methodologies.