PORE: Procurement-Oriented Requirements Engineering Method for the Component-Based Systems Engineering Development Paradigm

Cornelius Ncube & Neil.A.M. Maiden

Centre for Human-Computer Interface Design
City University, London, UK

Tel: +44 171 477 8166
Fax: +44 171 477 8859
E-Mail: [C.Ncube@soi, N.A.M.Maiden@]city.ac.uk
PORE URL: http://www.soi.city.ac.uk/pore/welcome.html

Abstract

Most current research in Component-Based Systems Engineering (CBSE) focuses on design and integration processes. There is little interest in the requirements engineering and product evaluation/selection processes that must precede design and integration. Also most current methods and tools support systems design and integration but neglect the requirements engineering and product evaluation/selection processes. However, in spite of this lack of focus on requirements engineering, a consensus seems to be emerging that the CBSE development process should be an iterative one of requirements engineering, systems design, product evaluation/selection and systems integration. This paper proposes a new method, PORE, to address the lack of requirements engineering methods and product evaluation/selection process guidance for the CBSE process. The paper ends with a ‘vision’ for future research directions for component-based systems engineering development process.

Keywords: PORE, systems procurement, requirements engineering and acquisition, COTS software products, product evaluation and selection, process guidance.

1: Introduction

As the next century approaches organisations are increasingly shifting the development processes of their complex software-intensive systems away from bespoke systems development to Component-Based Systems Engineering. Commercial software components that can be procured off-the-shelf (COTS) are now available to perform most of the functions that in the past required bespoke development. This use of COTS components has the potential for reducing the cost and time to develop software intensive systems. However, given the complexities of today’s software intensive systems, the cost and risk of procuring/purchasing wrong package(s)/component(s) due to inadequate requirements acquisition and product selection is large. The success of a Component-Based Systems Engineering, (CBSE) development process largely depends on the successful selection of COTS software component that meet core essential customer requirements.

However, the problem is that when building systems from COTS products, new and different types of requirements, (e.g. contractual and supplier requirements) need to be defined and new methods and techniques of acquiring these requirements and selecting candidate COTS products and suppliers also need to be defined. We are developing a new method, PORE, (Procurement-Oriented Requirements Engineering), (e.g. Ncube & Maiden 1997) which supports the requirements engineering and product evaluation/selection processes for CBSE development process. PORE uses an iterative process of requirements acquisition and product evaluation/selection as its main novel approach. The PORE approach has three main components:

- a process model that identifies four essential goals that should be achieved by any CBSE process and prescribes generic processes to achieve each of these goals as well as a sequence in which these goals should be achieved;

- a method box that includes methods, techniques and tools that are available to help undertake and achieve each of the processes;
- a product model that provides semantics and syntax for modelling software products

The three components are integrated into an approach that provides a requirements engineering team with a coherent process guidance for CBSE development process.

Section 2 outlines PORE's major components and in particular, its life-cycle processes. Section 3 describes PORE's iterative process of requirements acquisition and product evaluation/selection. Also described in section 3 are the methods, techniques and tools used in PORE for requirements acquisition and product identification, evaluation and selection. Section 4 describes PORE's situated process and how the requirement and software product models inform the iterative process of requirements acquisition and product evaluation/selection, therefore guiding the CBSE process. The paper ends with an outline and a vision for future research directions.

2: PORE: A Requirements Engineering Method For the CBSE Process

The basic PORE life-cycle process model has six generic processes. The process model describes the most fundamental processes undertaken during COTS product procurement. The processes are defined at 3 levels according to Humphrey's (1989) process model:

- the universal (U) level processes that describe general guidance for the actors in the process. Each describes a uniform sequence of processes;
- the worldly (W) level processes that are relevant to iterative process of requirements acquisition and product evaluation and selection. Each guides the sequence of tasks during product procurement.
- the atomic (A) level processes that are specific to individual methods, procedures, techniques and tools which enable the W-level processes.

Figure 1 depicts the six generic U-level PORE processes. It shows the processes which are often undertaken, although not all the processes are performed during each product procurement. For example, contract production does not take place if the component is procured internally. Management takes place throughout the procurement process. Software package selection and requirements acquisition are performed iteratively. Supplier selection and package selection often take place at the same time. Each processes has sub-processes and their main objectives are described below.

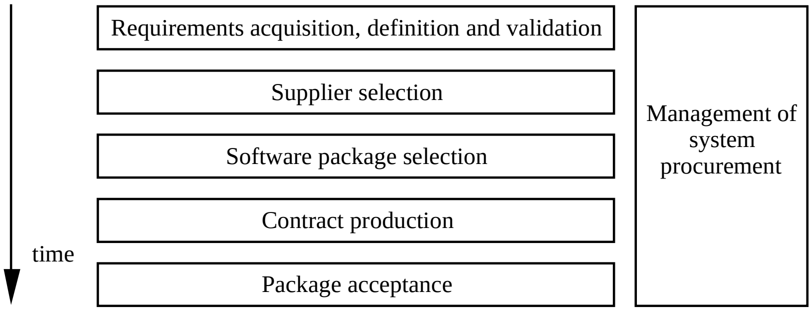
Requirements acquisition, definition and validation
Supplier selection
Software package selection
Contract production
Package acceptance
Management of system procurement
time

Figure 1: The Basic Overview of the PORE Process Model

- Management of System Procurement process – the objectives of this process are to plan and control the procurement process in order to fulfil the needs of the procurer in time and at a reasonable cost.
- Requirements Acquisition process – this process acquires and validates customer requirements. It also determines the current system architecture so that new component(s) can be integrated with it.

- Supplier Selection process – this process's objectives is to establish supplier selection criteria, evaluate suppliers, rank them according to the criteria and select the best-fit supplier(s).
- Software Package Selection – this process's objective is to identify candidate packages, establish selection criteria using customer requirements, evaluate the identified packages, rank them according to the criteria and select one or more package(s) which best meet the core essential customer requirements.
- Contract Production – this process's objective is to negotiate the legal contract with package suppliers and resolve an legal issues pertaining to the purchasing of the package and licensing.
- Package Acceptance – the objective of this process is to check the delivered package or system against the customer's original core essential requirements.

The main focus of this paper is the two processes of requirements acquisition and package selection. We aim to report work on the remaining processes in the near future.

3: Iterative Requirements Acquisition and Product Evaluation/Selection

In the CBSE development process, requirements are the cornerstone for any effective COTS procurement. Requirements become criteria for evaluating and selecting of candidate components and are embedded in the legal contract (contractual requirements). Requirements even provide acceptance criteria to check when the system or package is delivered that it meets customer's expectations. In CBSE the consequences of inadequate requirements engineering can be greater because COTS products are living systems and evolve over a long time. In a COTS intensive system, the integrated components will be developed and updated by different vendors at different development, update and evolution cycles hence requirements changes often will incur great additional costs to the customer.

However, the importance of requirements engineering to CBSE is not reflected in the current range and focus of research activities and in the available commercial methods, techniques and tools. Most current methods and tools support systems design (e.g. Garlan 1995) and integration (e.g. Vigdar et al 1996, Brown et al 1995) but neglect the requirements engineering and product evaluation/selection processes that must precede design and integration. In spite of this lack of focus on requirements engineering, there is an emerging consensus that the process of developing systems from COTS components should be an iterative one of requirements engineering, systems design, product evaluation/selection and product/systems integration (Fox et al 1997, Tran & Liu 1997).

Hence at the heart of PORE is the iterative and parallel process of requirements acquisition and product evaluation and selection. PORE's iterative process selects products by rejection, (i.e. the products that do not meet core customer requirements are selectively and iteratively rejected and removed from the candidate list). At the same time the products are selectively rejected, therefore resulting in a decreasing number of candidate products, the number and detail of customer requirements will be increasing. The result is an iterative process whereby the requirements acquisition process enables product selection and the product selection process informs requirements acquisition. This process is depicted Figure 2 below.

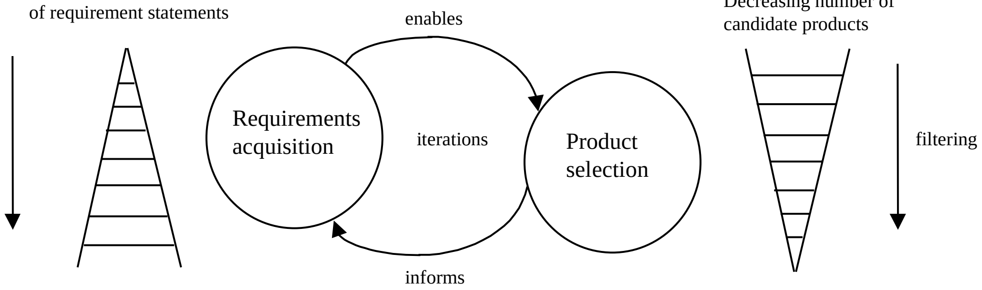
Increasing number and detail of requirement statements
enables
Decreasing number of candidate products
Requirements acquisition
iterations
Product selection
filtering
informs

Figure 2: Overview of the PORE's iterative process. Requirements acquisition enables product selection and product selection informs requirements acquisition. As the number and detail of requirements increases, the number of candidate products decreases.

3.1: Methods, Techniques and Tools Integration

Within this iterative process, the PORE method integrates different methods, techniques and tools for requirements acquisition and product identification and evaluation/selection with process guidance for choosing and using each technique. Some of the techniques, methods and tools integrated within the iterative PORE process are indicated below:

- Knowledge engineering techniques such as card sorting and laddering (e.g. Rugg and McGeorge 1995) which are useful when acquiring information about categories of products, suppliers, contracts and hierarchical information about product properties and customer requirements.
- Feature analysis techniques (Kitchenham & Jones 1997) to aid when scoring the compliance of each product feature to each customer requirement;
- MCDM (Multi-Criteria Decision Making) techniques such as AHP (e.g. Saaty 1990) and Out Ranking method (e.g. Fenton 1994) to aid in the decision making process during the complex product ranking and selection process;
- Argumentation techniques (e.g. Buckingham-Shum 1994) to record and aid the decision-making process;
- Requirements engineering methods such as Volere (e.g. Robertson 1997) for aiding the requirements engineering process;
- Requirements acquisition techniques such as ACRE (e.g. Maiden and Rugg, 1996) for acquiring customer requirements;
- Product (or component) identification tools such as the internet or Agora (e.g. Robert et al 1998) for identifying products or components available in the market;
- ATA (Architecture Tradeoff Analysis) for analysing architectures and SAAM (Software Architecture Analysis Method, SEI 1998) for evaluating software product architectures.

As well as integrating these techniques, PORE also provides guidelines for designing product evaluation test cases and for organising evaluation sessions. The guidelines are provided using different templates (e.g. Maiden & Ncube 1998) for requirements acquisition and product selection. The process is structured into stages and the first three templates provided are:

- Template-1, to guide the requirements engineer when acquiring essential customer requirements and product information sufficient to select and reject products as a result of supplier-given information;
- Template-2, to guide the requirements engineer when acquiring customer requirements and product information sufficient to select and reject products from supplier-led demonstrations using test-cases for individual requirements;
- Template-3, to guide the requirements engineer to acquire customer requirements and product information sufficient to select and reject products as a result of customer-led product exploration

Each template defines the types of product information (e.g. supplier, contract, architecture), types of requirements to acquire (e.g. functional, contractual, architectural, non-functional), requirements acquisition techniques to use and the best decision-making techniques to use. Figure 3 below shows each template within the iterative process stages.

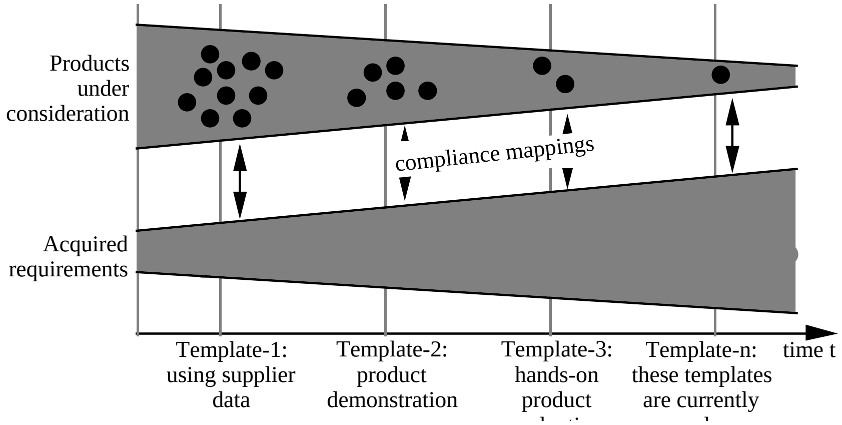
Products under consideration
compliance mappings
Acquired requirements
time t
Template-1: using supplier data
Template-2: product demonstration
Template-3: hands-on product
Template-n: these templates are currently

Figure 3: PORE's templates within the iterative process stages. Because of this iterative nature of the process, this means that each template can be used several times. Also shown in the figure process goals that each template must achieve and these are explained below.

3.2: Goal-Based Process Guidance

Another essential feature of PORE is that it prescribes four essential goals for selecting/rejecting candidate products. The four essential goals are integrated with templates as shown in Fig. 3. PORE rejects products according to compliance with:

- essential atomic customer requirements, (Goal 1);
- non-essential atomic customer requirements, (Goal 2);
- complex non-atomic customer requirements, (Goal 3);
- customer user requirements, (Goal 4).

The requirements engineering team must achieve these goals in a sequence. For this PORE prescribes four generic processes to achieve each of the four essential goals. The four generic processes are depicted in Figure 4 and are:

- acquire information about software products, customer requirements, suppliers requirements and contractual requirements from stakeholders;
- analyse acquired information
- use decision-making techniques to analyse and determine product-requirement compliance;
- reject one or more non-compliant candidate products

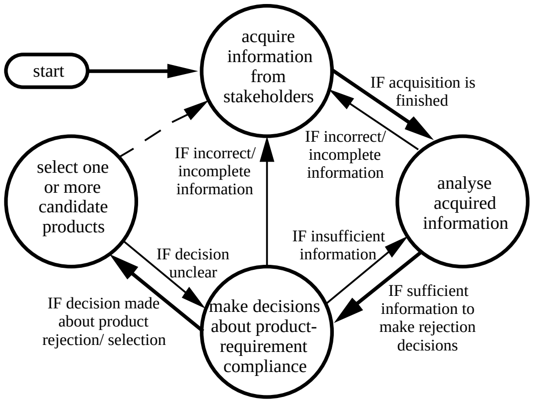
start
acquire information from stakeholders
IF acquisition is finished
IF incorrect/ incomplete information
analyse acquired information
IF incorrect/ incomplete information
select one or more candidate products
IF insufficient information
IF decision unclear
IF sufficient information to make rejection decisions
IF decision made about product rejection/ selection
make decisions about product-requirement compliance

Figure 4: A route map showing PORE's high-level generic processes for achieving each essential goal

The order in which the four processes are undertaken is context-driven and is determined by the current state of the situation model which is depicted in Figure 5. PORE's situations are as defined in Suchman (1987) which states that "every course of action depends in essential ways upon its material and social circumstances" (p50). The CBSE process is a very complex, so a large number of "situations" are possible at any point in the process. In PORE, our solution is to enable the RE (Requirements Engineering) team to model the "material circumstances" as situations that inform process guidance.

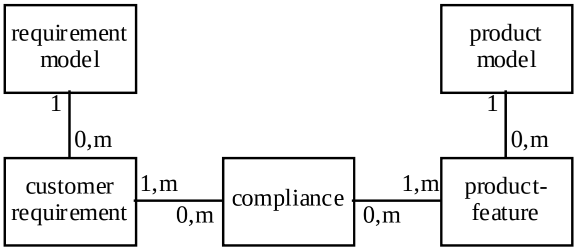

Figure 5: The structure of the situation model, and the relationships between the requirement, product and compliance sub-models.

As Figure 5 shows, a PORE situation is modelled in three parts. The first part models the current state of the customer's requirements. The second part models the degree of compliance between customer requirements model and product model, (product model and requirements model are discussed in more detail in section 4). Compliance is modelled as set of relationships between one or more customer requirements and one or more product features. The third part of the situations is a model of a software product.

PORE's context-driven process is made more complex by the large number of situations which may arise at any point in the process and many techniques from different disciplines (e.g. see section 2) which are available to achieve each process. As a result, PORE provides a multi-layered process guidance. As depicted in Figure 6, at any point in the process, three levels of guidance are provided. The process model level provides guidance to achieve the essential process goals. The situation model provides guidance at the other two levels. At the second level, it recommends techniques(s) to use to undertake a process by inferring general properties about the requirements, product and compliance models. At the third level, it recommends the content focus for applying each technique based on inferences about the current contents of the requirements, product and compliance sub-models. The process guidance is therefore given in the form of a triplet:

{process-goal, techniques-to-use, content-focus}

Of the three triplet contents, the process-goal part changes least and the content-focus changes most during each instance of a process as new information is added to the three sub-models and new inferences about the properties of the model are being made.

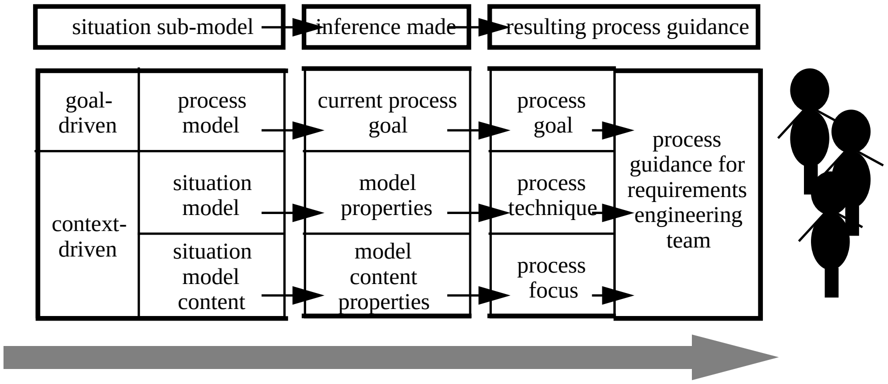

Figure 6: The three levels of process guidance which form the PORE process triplet. The current process goal is inferred from the process model. The technique to achieve this process is inferred from properties of the situation model. The focus of this technique's application is inferred from properties of the situation model content

4: Models for Guiding the CBSE Process

At the heart of the PORE method are three models, the product model, the requirements model and the product-requirement compliance model, see Figure 5. These models are used an as instrument for comparisons and evaluation of candidate software products and for driving and guiding the requirements acquisition and product selection processes.

4.1: Product Model

To enable effective COTS product evaluation and selection, there is a need to understand software products. PORE's product model (Ncube and Maiden 1998) enables each software product to be modelled in three different ways:

- Firstly, the product model models the observable behaviour of the product and in particular how the user interacts with the product. To achieve this, we draw on the existing use case modelling approaches, in particular the CREWS (Co-operative Requirements Engineering With Scenarios, Maiden et al, 1998).
- Secondly, the product model also models the product's articulated goals using goal-based requirements methods (e.g. Anton 1997).
- Thirdly, it also models the product's architecture using architecture modelling techniques such as those reported in Shaw (1996), Garlan et al. (1995) and SEI (1998).

To precisely model the properties of the COTS software product, the PORE approach uses a variety of abstract meta concepts such as goals to be achieved, objects to be used, actions taking place, functions to achieve actions and relationships between the meta-concepts, (Ncube & Maiden 1998). Figure 7 depicts the PORE's software product meta-model and its primitive concepts (e.g. agent, action, software component) and the meta-relationships linking the meta-concepts (e.g. performs, achieves, depends). The meta-concepts provide specific ways and strategies for traversing the product meta-model to instantiates its instances during requirement-product compliance mapping. Each step in the product-requirement mapping becomes a validation of the meta-model concepts against essential customer requirements.

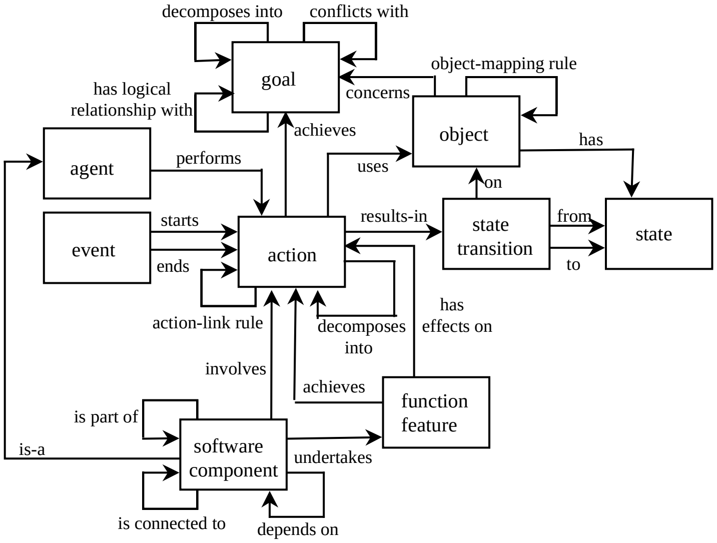

Figure 7: A meta-model for the software product showing the primitive concepts and relationships with which to model each software product.

4.2: The Requirement Model

The PORE method uses the requirement model to both acquire and elaborate the requirements statements and to check requirement-product compliance during the product evaluation and selection process. A critical factor in a successful requirements acquisition is to understand not only what the system under consideration should do (functional requirements), but also the way in which it should provide its services (non-functional requirements). A broader view of requirement acquisition, therefore should go beyond the description of what the system is expected to do (the system's functionality) and include system properties and constraints under which the system must operate, e.g. (architecture requirements). In the CBSE development process, this view is even taken further to include information about product suppliers such as their technical capabilities, application domain experience, ISO standard certification, CMM level, etc. and legal issues involved in product procurement such as negotiating contract terms and conditions, licensing arrangements, etc. When taken in this context requirements represent both a model of what is needed and statements of the problem under consideration in various degrees of abstraction. Figure 8 depicts the meta-concepts of the requirement model.

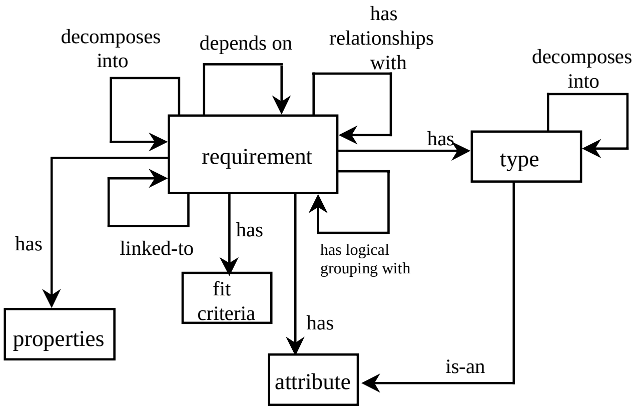

Figure 8: The requirement model and its meta-concepts.

5: Discussion and Further Development of PORE.

We believe that PORE is a novel approach to guiding the CBSE development process. In particular, it addresses the lack of process and method guidance for requirements acquisition and component/product evaluation/selection processes which must take place before system design. However, the PORE approach still has significant limitations that we aim to overcome in future developments. One of the problems, as mentioned above, is that the iterative process of requirements acquisition and product evaluation/selection is very complex. At any point in this complex process, a large number of possible situations can arise. For example, stakeholders may define a large number of requirements, so the requirements sub-model can give rise to a large number of different situations. The requirements engineering team can also evaluate a large number of software products, so the product and compliance sub-models as well, can give rise to a very large number of situations. In addition, PORE can sometimes recommend a large number of processes and techniques to use in a single situation. To handle this scale of complexity, we believe that a software tool is needed that has a computational model to detect situations and generate process guidance.

As a result, a prototype tool known as PORE Process Advisor is being developed to support the PORE approach. The main components of the tool are a process engine which will analyse the current set of goals to be achieved (stored in the goal agenda), model properties (inferred by the situation inference engine) and instructions from the requirements engineering team to recommend process advise in the form of the process triplet, i.e. <process-goal, technique, focus>. The PORE Process Advisor tool is being developed to integrate with existing software tools like Microsoft Access, Visual Basic, and Kappa-PC. It is also linked to Rational's RequisitePro requirements management tool (for requirements management) and CREWS-SAVRE tool (e.g. Maiden et al. 1998) for generating scenarios. We shall be reporting usage trials of PORE Process Advisor tool in the near future Further details of PORE in general can be found in the dedicated web site: http://www.soi.city.ac.uk/pore/welcome.html. We welcome any feedback, suggestions and participation in the development of PORE.

6: Future Research Directions for the CBSE Paradigm

We believe that the vision for the future research directions will be in the ‘shared knowledge’ development process and in new directions in the training and education of the systems developer of the future as well as the organisations which will be developing these COTS-based systems and the user organisations. A whole new set of skills in both project management and systems development will be required for the CBSE process. A brief outline of the two research directions is given below.

6.1. Shared Knowledge Development Process

As the shift to CBSE development will rapidly gather speed in the near future, we envision a ‘vision’ of a process whereby shared knowledge of products, development skills and experiences, new techniques and methods as the way forward. This shared knowledge will result in a supply chain of components/products, skills and experiences and personnel. Also organisations will have access to each other’s development infrastructure along the supply chain and share business strategies and objectives, expertise, ideas, risks and information. Joint technology development programmes will be possible through shared knowledge and close relationships. This vision of a shared knowledge process is depicted in Figure 9 below.

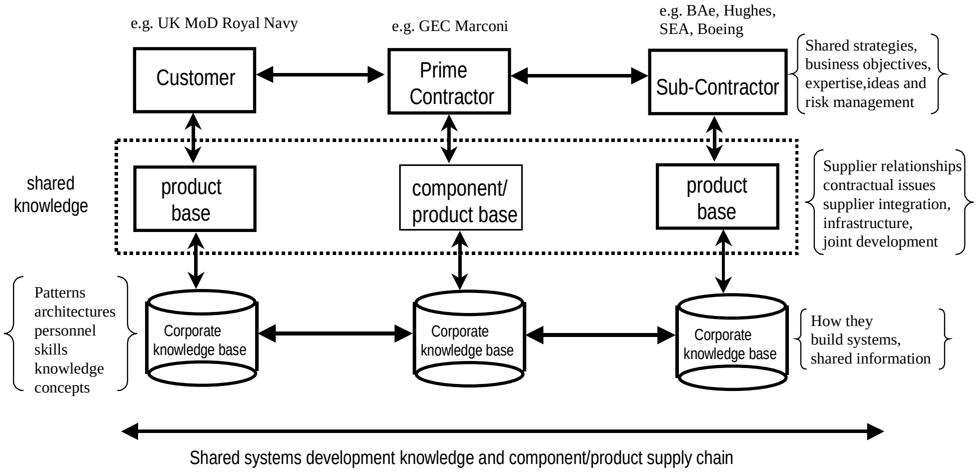
e.g. UK MoD Royal Navy
e.g. GEC Marconi
e.g. BAe, Hughes, SEA, Boeing
Customer
Prime Contractor
Sub-Contractor
Shared strategies, business objectives, expertise,ideas and risk management
shared knowledge
product base
component/ product base
product base
Supplier relationships contractual issues supplier integration, infrastructure, joint development
Patterns architectures personnel skills knowledge concepts
Corporate knowledge base
Corporate knowledge base
Corporate knowledge base
How they build systems, shared information
Shared systems development knowledge and component/product supply chain

Figure 9: A vision of shared knowledge as the cornerstone for the future success of the CBSE paradigm.

6.2: The ‘soft’ Issues: Training and Education

The number of techniques and knowledge from different disciplines required in the CBSE process for it to be a success will mean that it will be impossible for any individual to possess all the necessary skills. As a result the development team of the future will be composed of team members from many backgrounds therefore forming ‘smart teams’. The project management and development skills that are required for the CBSE process are significantly different from those required for traditional systems development. Even the project manager of the future will be required to do significantly different task as opposed to what a traditional project manager has to do. Organisations themselves will probably have to change the way they do their business. For example, selecting a product to be included in the integrated systems largely results in the selection of the product supplier. Therefore, in a COTS-intensive system, the integration of the different products results in the integration of product suppliers as well. As such, the development organisation will need to manage not only its relationships with the individual suppliers but also the relationships between the integrated suppliers. As a result, we have identified personal or ‘soft’ issues as major research area. A different sets of skills will be required and therefore CBSE principles need to be incorporated into the training and education of systems developers of the future, be it in universities, colleges or organisation training programmes.

7: References

Anton A. I., 1997, 'Goal Identification and Refinement in the Specification of Software Based Information Systems, PhD Thesis, Georgia Institute of Technology, June 1997.

Fenton N., 1994, 'Out Ranking Method for Multi-Criteria Decision Aid: with emphasis on its role in systems dependability assessment, Centre for Software Reliability, City University, London, UK.

Fox G., Marcom S. & Lantner K, 1997, A Software Development Process for COTS-based Information System Infrastructure, Proceedings of the 5th International Symposium on Assessment of Software Tools and Technologies (SAST'97), pp133-143

Garlan D., Allen R. & Ockerbloom X., 1995, 'Architectural Mismatch or Why it's hard to build systems out of existing parts', Proceedings 17th International Conference on Software Engineering, IEEE Computer Society Press.

Humphrey W.S., 1989, 'Managing the Software Process', Addison-Wesley

Kitchenham B & Jones L., 1997, 'Evaluating Software Engineering Methods and Tools: Part 5, The Influence of Human Factors', Software Engineering Notes 22(1).

Maiden N.A.M. & Rugg G., 1996, 'ACRE: Selecting Methods for Requirements Acquisition, Software Engineering Journal 11(5), 281-292.

Maiden N.A.M., & Ncube C., 1998, 'Acquiring COTS Software Selection Requirements', IEEE Software, March/April 1998 Issue, 46-56

Maiden N.A.M., Minocha S., Manning K. & Ryan M., 1998, 'CREWS-SAVRE: Scenarios for Acquiring and Validating Requirements', Proceedings 4th International on Requirements Engineering (ICRE98), IEEE Computer Society Press.

Ncube C & Maiden N.A.M., 1997, 'Procuring Software Systems: Current Problems and Solutions', Proceedings REFSQ97 workshop, CaiSE97, Barcelona, Spain, June 16-17

Ncube C & Maiden N.A.M., 1998, 'Why Model Software Products: Why, Guiding the CBSE Process, of Course!', Proceedings of the CBISE98 workshop for CAiSE' 98, Pisa, Italy.

Robert C. S., Scott A. Hissam, & Kurt C.W, 1998, 'AGORA: A Search Engine for Software Components, Software Engineering Institute, Carnagie Mellon University, USA.

Robertson S., 1998 'Volere: Requirements Specification Templates, Edition 6' , Atlantic Systems Guild, http://www.atlsysguild.com/GuildSite/Robs/Template.html,

Rugg G. & McGeorge P., 1995, 'Laddering', Expert Systems 12(4), 339-346

Saaty T. L., 1990, 'The Analytic Hierarchy Process', New York, McGraw-Hill.

SEI, 1998, 'Architecture Tradeoff Analysis (ATA) and Software Architecture Analysis Method (SAAM)', Software Engineering Institute, Carnegie Mellon University, USA, http://www.sei.cmu.edu/ata/ata_init.html

Shaw M., 1996, 'Truth vs Knowledge: The Difference Between What a Component Does and What We Know It Does', Proceedings of the 8th International Workshop on Software Specification and Design, IEEE Computer Society Press, March 1996.

Suchman L., 1987, ‘Plans and Situated Actions: The Problems of Human-Machine Communication’, Cambridge University Press.

Tran V. & Liu D, A Procurement-centric Model for Engineering Component-based Software Systems, Proceedings of the 5th International Symposium on Assessment of Software Tools and Technologies (SAST’97), pp70-80

Vigdar M.R., Gentleman, M. W. & Dean J., 1996, ‘COTS Software Integration: State of the art’, National Research Council, Canada, NCR-CNRC Report, January 1996.

A Component Model Proposal

Jim Q. Ning

Andersen Consulting

3773 Willow Road

Northbrook, IL 60062, U.S.A.

+1 847 714 2537

jning@cstar.ac.com

Abstract

This position paper describes a conceptual model for Component-Based Software Engineering (CBSE). The model is an attempt to define what CBSE is essentially about and help answer critical questions concerning how CBSE relates to and distinguishes itself from other software development paradigms/concepts such as object-orientation.

1. Motivation

As clearly stated in the opening statement of this year's CBSE Workshop's theme description:

> "There is growing interest in the notion of software development through the planned integration of pre-existing software components. This is often called component-based development (CBD), component-based software engineering (CBSE), or simply componentware. While the broad concepts of CBSE are well known and easily stated, a closer look reveals that the term CBSE is used in a diverse set of situations, encompasses a variety of characteristics, and is often given many different interpretations."

To clarify the misconception and confusion, this position paper proposes a model or framework which, hopefully, will be simple and easy to explain to people on one hand and yet rich enough to capture all the key component-related concepts. In the next section, a concept diagram is presented to depict the key concepts and their relationships. Then, a glossary is provided to further define/describe the concepts. Finally, I will explain why such a component model is important and beneficial to the CBSE community.

During the workshop, I anticipate to receive comments from the participants to improve the proposed model. The objective is for this young community to reach some consensus on what CBSE is essentially about and help answer critical questions concerning how CBSE relates to and distinguishes itself from other software development disciplines such as object-orientation.

2. Concept Diagram

The concept diagram is shown on the last page of this paper (Appendix). It is meant to illustrate the important CBSE concepts, the relationships among themselves and with other software engineering concepts. The component concepts are shown in plain (white) boxes, and other concepts in gray boxes. The following link types are used to describe the relationships between the concepts:

- **Aggregation**. This is shown as a solid-line path with a hollow diamond at one end. The concept that is connected by the diamond end is the aggregate.

- **Association**. A binary association is shown as a solid-line path that connects two concepts. A ternary