A COTS-SOFTWARE REQUIREMENTS ELICITATION METHOD FROM BUSINESS PROCESS MODELS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

ERCAN ASLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF MASTER OF SCIENCE

IN

THE DEPARTMENT OF INFORMATION SYSTEMS

AUGUST 2002

Approval of the Graduate School of Informatics

Prof. Dr. Neşe Yalabık
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Semih Bilgen
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Onur Demirörs
Supervisor

Examining Committee Members

Prof. Dr. Semih Bilgen ……

Assoc. Prof. Dr. Onur Demirörs …….

Assoc. Prof. Dr. Elif Demirörs …….

Dr. Altan Koçyiğit …….

Nusret Güçlü …….

ABSTRACT

A COTS-SOFTWARE REQUIREMENTS ELICITATION METHOD FROM BUSINESS PROCESS MODELS

Aslan, Ercan

M.S., Department of Information Systems

Supervisor: Assoc. Prof. Dr. Onur Demirörs

August 2002, 86 pages

In this thesis, COTS-software requirements elicitation, which is an input for RFP in software intensive automation system's acquisition, is examined. Business Process Models are used for COTS-software requirements elicitation. A new method, namely CREB, is developed to meet the requirements of COTS-software. A software intensive system acquisition of a military organization is used to validate the method.

Key Words: Mission Need Document, Request For Proposal Document, Commercial Off-The Shelf, Business Process Model, Event Driven Process Chain.

ÖZ

İŞ AKIŞ MODELLERİ KULLANARAK HAZIR YAZILIM GEREKSİNİMLERİNİN ELDE EDİLMESİ METODU

Aslan, Ercan
Yüksek Lisans, Bilgi Sistemleri Bölümü
Tez Yöneticisi: Doç. Dr. Onur Demirörs
Ağustos 2002, 86 sayfa

Bu çalışmada, yazılım ağırlıklı otomasyon sistem alımlarında hazırlanan Teklife Çağrı Dosyasına girdi olacak Hazır Yazılım Gereksinimlerinin eldesi incelenmiştir. Hazır Yazılım Gereksinimlerin eldesinde İş Akış Modelleri kullanılmıştır. Hazır Yazılım Gereksinimlerinin eldesi için CREB adı verilen bir method geliştirilmiştir. Methodun geçerliliği için askeri bir kurumun yazılım ağırlıklı bir sistem alımı kullanılmıştır.

Anahtar Kelimeler: Görev İhtiyaç Dökümanı, Teklife Çağrı Dosyası, Hazır Yazılım, İş Akış Modeli, Olay Akış Zinciri.

To My Family

ACKNOWLEDGEMENTS

I express sincere appreciation to Assoc. Prof Dr. Onur Demirörs for his guidance, assistance and insight throughout the research. Also I thank to Mr. Okan Yıldız, Mr. Barış Çayırlı, Mr. Ümit Bak, and Mr. Faruk Yiğit for their support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AHP	**:**	Analytic Hierarchy Process
AQAP	**:**	Allied Quality Assurance Publication
ARIS	**:**	Architecture of Integrated Information Systems
BPM	**:**	Business Process Model
CBD	**:**	Component Based Development
CBS	**:**	Component-Based Software
CBSD	**:**	COTS-Based Software Development
COTS	**:**	Commercial-Off The Shelf
CREB	**:**	COTS-software Requirements Elicitation from Business Process Model
CURE	**:**	COTS Usage Risk Evaluation Social Technical Approach To COTS Evaluation
DoD	**:**	Department of Defense (U.S.)
DB	**:**	Database
DBMS	**:**	Database Management Systems
EEPC	**:**	Extended Event Process Chain
ERP	**:**	Enterprise Resource Planning
FAA	**:**	Federal Aviation Administration
FAD	**:**	Function Allocation Diagram
GIS	**:**	Graphical Information Systems
HW	**:**	Hardware
IEEE	**:**	Institute of Electrical and Electronics Engineers
IIDA	**:**	Infrastructure Incremental Development Approach
IS	**:**	Information Systems
ISO	**:**	International Standards of Organization
IT	**:**	Information Technology
LOC	**:**	Line of Code

MCDA	**:**	Multi Criteria Decision Analyze
NT	**:**	Network
OTSO	**:**	Off-the-Shelf Option Method
OS	**:**	Operating System
OSE	**:**	Open System Environment
PORE	**:**	Procurement Oriented Requirements Engineering
SCA	**:**	System Components Allocation
SEI	**:**	Software Engineering Institute
SRS	**:**	Software Requirements Specification
STACE	**:**	Social Technical Approach To COTS Evaluation
SW	**:**	Software
TLFC	**:**	Turkish Land Forces Command
UK	**:**	United Kingdom
US	**:**	United States

CHAPTER 1

1. INTRODUCTION

Despite the non-existence of a wholly accepted definition for the Commercial Off-The Shelf (COTS) software products, the Software Engineering Institute (SEI) defines a COTS product as:

" A product that is sold, leased or licensed to the general public; that is offered by a vendor trying to profit from it; that is supported and evolved by the vendor who retains the intellectual property rights; that is available in multiple, identical copies; that is used without modification of its internals."[1]

Institute of Electrical and Electronics Engineers (IEEE) defines characteristics of COTS as:

"COTS software is stable and normally well-defined in terms of documentation, known capabilities and limitations. It usually comes with "how to operate" documentation. COTS software is defined by a market-driven need. It is commercially available and its fitness for use has been demonstrated by a broad spectrum of commercial users. Also, the COTS software supplier does not advertise any willingness to modify the software for a specific customer." **[IEEE, 1998]**.

Some examples of COTS software products are operating systems, database management systems, e-mail packages, word processors, anti-virus systems, and requirements management tools. These products are particularly popular in application areas, such as payroll, banking, insurance, accounting, networking, inventory control, and systems software **[McClure, 1989]**.

There are many reasons to consider a COTS product as a silver bullet **[Brooks, 1987]**. "Developing a software intensive system with as much COTS

[1] http://www.sei.cmu.edu/cbs.htm

functionality as possible saves you from reinventing the wheel" **[Breslin, 1986]**. Following functions could be achieved with COTS components:

-Accessed immediately,

-Obtained at a significantly lower price, and

-Developed by someone expert in that functionality **[Voas, 1998].**

"Also, a long development backlog may be a good enough reason to consider a package. In-house projects may take several years to develop and may go seriously over budget at the end of it" **[McClure, 1989]**.

Not only private companies benefit from COTS-software products but also public organizations shift their preferences to COTS products. One of the most far-reaching changes in United States Department of the Defense and Government policy has been in the acquisition and development of computer-based systems. Recent policy statements now strongly favor the use of commercial off the shelf (COTS) products, especially those in the domain of information systems.

While there are numerous reasons for this policy shift, a critical factor is the growing cost of building and maintaining government-unique systems in a short time while there are very similar systems available in the commercial marketplace at a relatively far lower expense[2]. In spite of the boom in the use of COTS products, most organizations experience major problems in their implementation[3]**.** Forrester Research estimated that for every $1 spent on a COTS-software package, $9 is spent trying to integrate it, and this integration accounts for 30% of information technology (IT) development budgets. One reason for these problems is inadequate requirements engineering (acquisition) and COTS product evaluation and selection **[Ncube, 1998].** The most important problem about COTS software is the lack of a mature method or a standard for acquirers to acquire those products.

The definition of functional and non-functional properties of the COTS product is very problematic. Although there are many standards and methodologies for software development process and even for appraisal of a COTS product there is

[2] http://www.sei.cmu.edu/cbs/papers/monographs/dod-cots-policies/dod-cots-policies.htm
[3] http://www.soi.city.ac.uk/~dg571/connie.html

not any well-defined solution. For example IEEE's Software Requirements Specification (SRS) standard states that:

"This recommended practice is aimed at specifying requirements of software to be developed but also can be applied to assist in the selection of in-house and commercial software products. However, application to already-developed software could be counterproductive." **[IEEE, 1998].**

1.1. STATEMENT OF PROBLEM

Information Systems (IS) provide people and organizations with the ability to use the information effectively. Especially in military domain, military institutions and units benefit from IS in military communication mostly as data or vocal transmission. Defense industry has became one of the major investors in IS via command, control, communication, computer, intelligence, surveillance, and reconnaissance (C4ISR) areas.

Among data, personnel, software, and hardware the software constitutes the biggest portion of cost and uncertainty in the IS projects. Besides, requirements elicitation has vital importance in software development process to build the desired system at a reasonable cost and a schedule.

There are many ways to acquire software products. These ways are custom development, reuse (in-house), and package (COTS, components, and ERP) solutions. Among them custom and COTS- software development solutions take an important portion.

The *system definition phase* is one of the most important phases of a software intensive systems acquisition process. System definition phase also takes relatively much time in system development process. Most of the system requirements can be generated after the mature definition of the system. Many techniques can be used in order to define the systems. Process modeling technique is one of them; and it is very useful for reducing the complexity of the system. But, there is not enough information about process-modeling techniques' capabilities, and their ability to support COTS-software products functional, and non-functional requirements elicitation.

On the other hand, COTS-software requirements are regarded as one of the supplier's duties rather than the acquirer's responsibilities in system development process. Current practices usually ignore acquirer's involvement.

Today there is a tendency that COTS-software products and their requirements are determined based on the experiences in the previous projects. However, performance, business, and source code requirements obtained from previous projects do not exactly express current requirements.

Some parts of the software components could be decided to be COTS-software even without executing market availability and performance specifications analysis. Consequently, a risk occurring due to unavailability of COTS-software emerges in later phases of software integration.

1.2. APPROACH

The purpose of this thesis study is to eliminate or mitigate problems stated in the previous section. The study is performed in three phases: literature survey, development of the model, and an application.

In the literature survey, package system's properties, development methods and techniques, and requirements engineering domain are examined. Business process models' effect on the requirements elicitation is studied.

In the second phase, COTS-software requirements elicitation model is developed based on the observations obtained from IS development project applications in Turkish Land Forces Command (TLFC), learned lessons, and literature survey. The developed method is named as CREB (COTS-software Requirements Elicitation Process from Business Process Model). It can be used by acquiring organizations between *Acquisition Initiation Phase to RFP Preparation* activities.

The CREB method consists of five sequential and iterative phases as depicted in **Figure-1.** The first phase is the Existing System Definition phase and includes all pre-studies and preparing AS-IS model via business process modeling (BPM). The System Requirements Elicitation follows the first phase and requirements are

generated from the AS-IS model. At third phase, system components allocation is planned. The fourth phase is the Make/Buy Decision phase, which is especially used for packages' cost/benefit and availability analysis. And the method ends with a Cost Analysis activity, the purpose of which is to identify the feasibility of the project.

In the last phase of the study, the validity of the method is questioned. The method has been applied to an actual automation project undertaken by a department of TLFC. In this application, a business process-modeling tool is used for requirements elicitation activity. In this application, process-modeling tool's functional and non-functional requirements elicitation capabilities are primarily examined. Then, other usage problems in COTS-software products, which are addressed in the statement of problem section, are assessed. As a result of this application, the method is evaluated to identify its advantages and disadvantages.

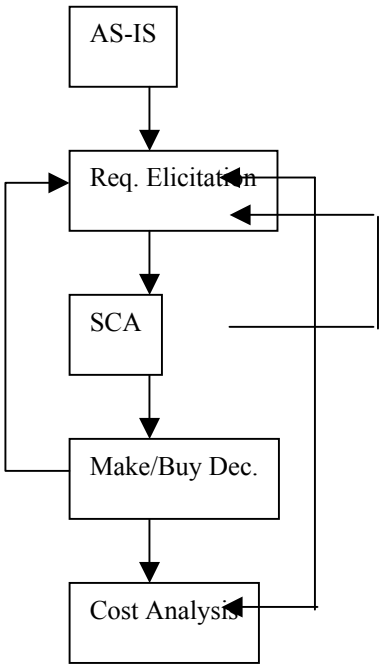


FIGURE-1 CREB PROCESS

1.3. THESIS OUTLINE

This study is divided into six chapters, namely introduction, background, related work, method design, application of the developed method and conclusion.

After an introduction part in chapter 1, the background of the research is described in chapter 2. In the background chapter, software package systems' differences, traditional versus package software development process, business process modeling, and COTS-software development in requirements engineering field are explained.

In Chapter 3, the related works are examined. In this chapter, six COTS-software development methods and DoD (Department of Defence), FAA (Federal Aviation Administration), NATO, and SEI's studies in COTS-software domain are examined.

In Chapter 4, proposed method, CREB, is described. In Chapter 5, application of the CREB is presented.

In Chapter 6, the proposed method is evaluated in the light of obtained gain from the application.

CHAPTER 2

2. BACKGROUND

In this chapter, firstly package software systems and software development evolution is examined. Then a business process modeling method (Architecture of Integrated Information Systems – ARIS), used in the application is examined.

2.1. PACKAGED SYSTEMS

Commercial and packaged based software development can be classified as Enterprise Resource Planning (ERP), Component Based Software Engineering (CBSE) and COTS-Based Development (CBD). This thesis examines only the CBD process.

In his research, Ncube defines those package system's differences according to the size of the packages and vice areas[4].

The ERP systems are designed to change organizations' business activity through IT. Companies have to re-organize their business processes and organizational structures. A clear trend both in private and public companies with regard to their options for software based management information system is the fast and wide proliferation of large packaged ready-made ERP systems, which are among the most extreme instances of current customizable-off-the shelf software packages **[Franch, 2001].**

The goal of CBSE is to enable system assemblers to compose systems and applications from software components. Software components are binary units of independent production, acquisition and deployment that interact to form a functioning system **[Szyperski, 1998].** CBSE on the other hand provides customized system functionality. A component is a software artifact consisting of three parts: a

[4] http://www.soi.city.ac.uk/~dg571/connie.html

service interface, a client interface and an implementation. The service interface consists of the services that the component exports to the rest of the world; the client interface consists of those services used by this component exported from other components, and the implementation is the code necessary for the component to execute its intended functionality[5]**.** A component must interact with other software. CORBA, Sun's Java Beans are examples of CBS products.

COTS-software products can be considered between CBS and ERP systems. COTS-software products provide common functionality for one application. SEI defines a COTS product as:

" A product that is sold, leased or licensed to the general public; that is offered by a vendor trying to profit from it; that is supported and evolved by the vendor who retains the intellectual property rights; that is available in multiple, identical copies; that is used without modification of its internals."

2.2. CURRENT STATE OF THE SOFTWARE DEVELOPMENT

The organizations are increasingly shifting their system development processes away from custom to component-based development. Especially public sectors that are able to use the massive technology investments of the private sectors get the benefits of reduced cycle time and costs, and for greater reliability and availability anymore giving more attention to commercial solutions. But this evolution is bringing more problems to software development process. Traditional, sequential system development activities are not satisfying expectations of COTS-based development activities.

The SEI has developed a process framework for working with COTS-based systems. They are defining this new process driver with those statements:

"CBS development is an act of composition. The realities of the COTS marketplace shape CBS development. CBS development occurs through simultaneous definition and trade-off of the COTS marketplace, system architecture,

[5] http://www.sei.cmu.edu/cbs/icse99/papers/16/16.htm

and system requirements." **[Oberndorf, 2000]**. This fundamental change is shown in **Figure-2 [Oberndorf, 2000]**.

On the figure's left side traditional development approach, right side COTS-based approach is given. System developers must consider requirements, architecture, and marketplace simultaneously. According to Oberndorf, when process changes, people must change as well. This means COTS-based systems development is not just an engineering or technical change; it is a business, organizational and cultural change, too.

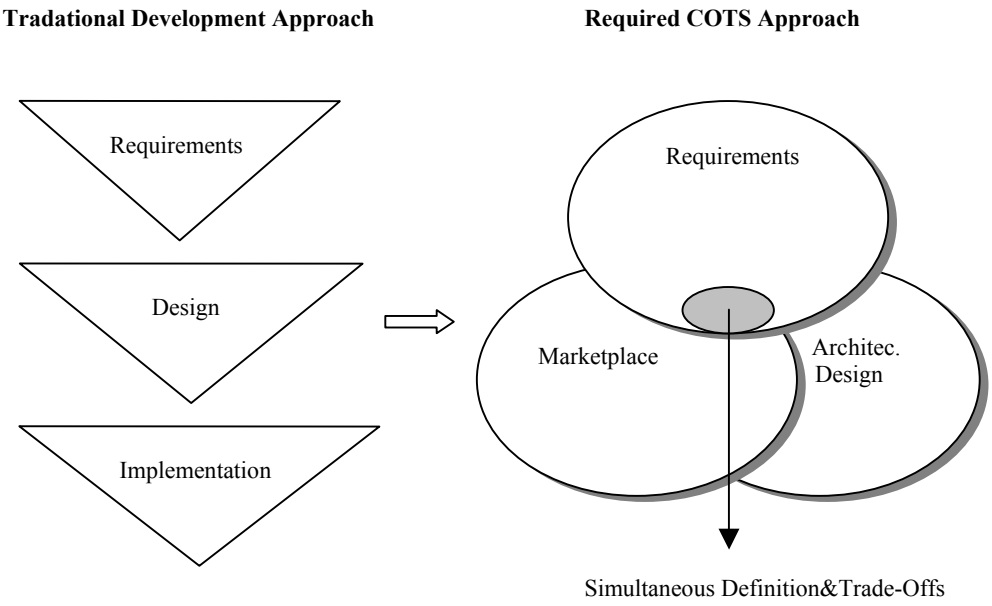


FIGURE-2 TRADITIONAL VERSUS COTS-BASED APPROACH

2.3. COTS-SOFTWARE REQUIREMENTS ENGINEERING

The process of establishing the services the system should provide and the constraint under which it must operate is called requirements engineering **[Sommerville, 1995**]. Indeed, it is a widely agreed view in the systems development community that errors generated during the requirements engineering phase are the most expensive to fix **[Boehm, 1981].**

There are many methods, standards, and guidelines for gathering traditional software requirements, however these methods don't satisfy expectations of COTS software domain. Because needed requirements for previously developed components are different and requirements generally focus on acquisition subjects. Available COTS-software components in the market shape basic requirements, so, market driven approaches or methods are needed for requirements elicitation. Although requirements engineering is important, previous studies on COTS-based development tended to focus on systems integration, evaluation, design and architectures **[Garlan et al, 1995].**

This lack of researchs on requirements engineering and COTS-software development are showed in **Figure-3**[6]**.**

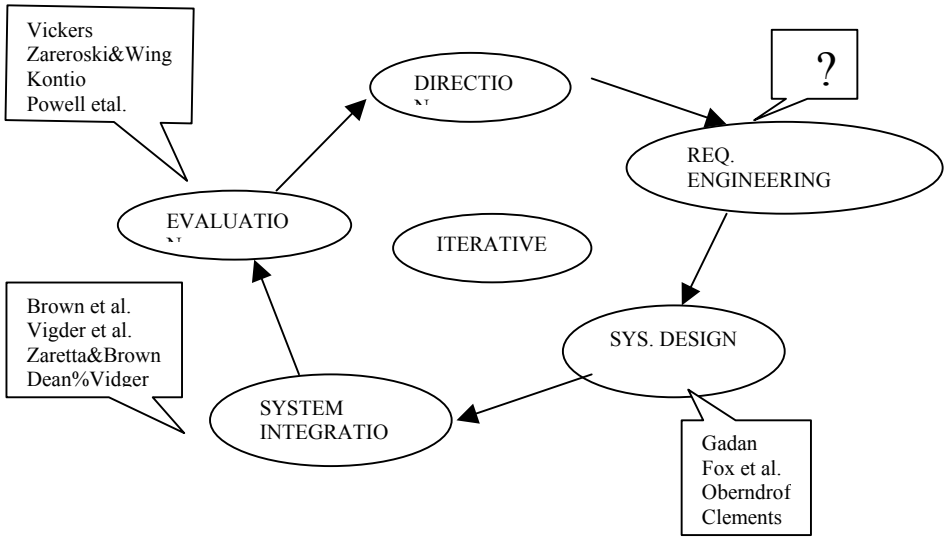


FIGURE-3 COTS-DEVELOPMENT FOCUSES

2.4. ARCHITECTURE OF INTEGRATED INFORMATION SYSTEMS

ARIS is a unique and internationally renowned method for optimizing business processes and implementing application systems.

[6] http://www.soi.city.ac.uk/~dg571/connie.html

Business models are a crucial prerequisite for analyzing business processes, bringing projects in line with the overall company objectives, and finally for finding the perfect information structures in form of a compound of distributed, integrated systems to support these lean organization structures **[Scheer, 1998].**

The ARIS concept provides a full-circle approach from the organizational design of business processes to IT implementations. ARIS-Toolset, developed by Prof. Scheer Co., is an integrated business process-reengineering tool. It provides step-by-step system development through using various business process modeling and analysis. ARIS offers reference models for each different industry. ARIS brings about the efficiency of system reengineering such as time reduction, quality improvement, and risk reduction by interfacing with diverse Computer Aided Software Engineering (CASE) tools (IEF/Composer, KEY, and Oracle).

ARIS methodology introduces an architectural concept of integrated business process to express business process more clearly. It classifies the business process by five different types of view including data view, control view, production, organization view, and function view **(Figure-4).** Such a way of classification helps reduce the complexity of process and allows a systematic and comprehensive analysis. This makes it possible to describe individual views using specialized methods without having to incorporate the corresponding relationships to other views **[Scheer, 1994].**

The views in ARIS are divided according to the relationships between the components. Events define changes in the status of information objects. Statuses and events form the *data view* of the ARIS. The functions to be performed and their relationships form a second view, *the function view*. This view contains the description of the function, the enumeration of the individual sub-functions that belong to the overall relationship and the positional relationships that exist between the functions. The *organization view* represents a combination of the users and the organizational units as well as their relationships and the relevant structures. Information technology resources constitute the *resource view*. The *control view* is introduced as additional view in which the relationships between the views are described.

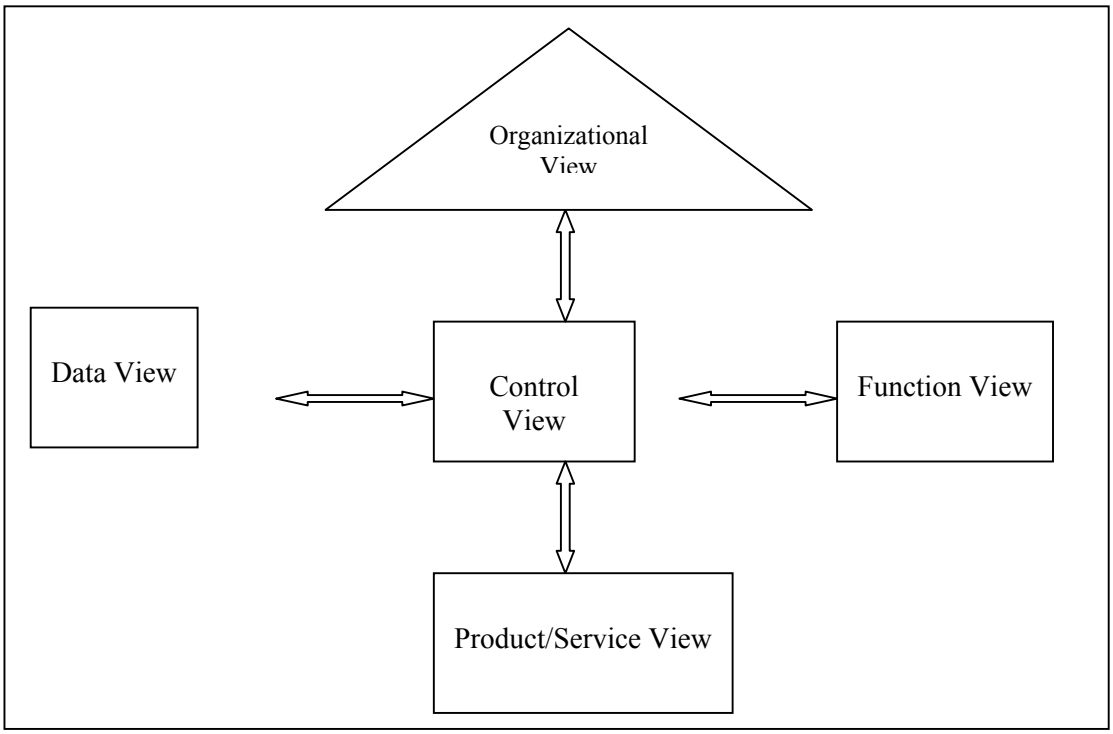


FIGURE-4 ARIS VIEWS OF THE PROCESS MODEL

The integration of these relationships within a separate view makes it possible to systematically enter all the relationships without any redundancies.

In ARIS, however, the lifecycle does not have the significance of a procedural model for developing an information system; rather it defines the different levels based on their proximity to information technology. This follows a three-tier model **[Scheer, 1994]**. This three- tier model and descriptive levels of an information system showed in **Figure-5 [Scheer, 1994].**

According to the ARIS, analysis of the *operational business problem* is the first activity in systems development. This step incorporates the IT options for the support of business processes and decisions. Therefore, only semi-formal descriptive methods are used to represent the description of the business problem.

The *requirements definition* describes the business application which is supported in a formalized description language so that requirements description as the starting point for a consistent translation of the requirements definition into

information technology. The requirements definition is very closely associated with the problem description, as depicted in **Figure-5**.

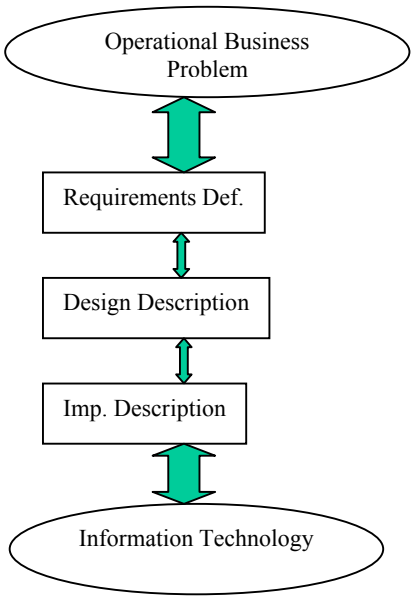


FIGURE-5 DESCRIPTIVE LEVELS OF AN INFORMATION SYSTEM

The *design specification* level is reached as soon as the conceptual environment of the requirements definition is transferred to the categories of an IT-oriented conversion. In design description, the module or user transactions that execute the functions are defined, not the functions themselves. This level can also be thought as an adaptation of the requirements description to the general ways of describing information technology. Thus, the requirements definition and the design specification are loosely linked **[Scheer, 1998].**

At the *implementation description***,** the design specification is transferred to concrete hardware and software components. The implementation description is very closely linked to the development of IT.

The requirements definition level is particularly significant because it is both a repository of the long-term business application and the starting point for further steps in generating the implementation description. The requirements definitions

possess the longest lifecycle and through their close affinity to the description of the business problem also document the greatest benefit of the information system. For this reason, the development of requirements definitions or semantic models has the highest priority **[Scheer, 1998].** The creation of the views and the descriptive levels, combined with the initial business problem solution constitutes the ARIS architecture.

CHAPTER 3

3. RELATED WORK

In this chapter some important COTS-software development models, techniques, and methodologies are explained and COTS-software usages and studies of Software Engineering Institute, Federal Administration Aviation, Defense of Department in United States of America, and NATO's are examined.

3.1. COTS- SOFTWARE DEVELOPMENT METHODS

COTS-software development process includes only procurement of a trade software product for using in a software development process, so COTS-software product development is out of this thesis scope.

COTS-software development process has to be different from traditional software development process. Because COTS-software is a ready to use product, and users only have to find a proper product, which satisfies the requirements and integrates it into the developed project. COTS-software development methods don't follow a sequential process like most software development processes. COTS-software development process is composed of acquiring customer requirements, evaluating available products, selecting products from market, and the design of the system's architecture. The evaluation and selection process' issues are product assessment and decision-making techniques.

In contrast to traditional software development standardization, COTS-software development still doesn't have any common standard or usage method. But there are many studies and regulations about this area. In this section, six well-known COTS-software development methods, techniques and their properties are explained. These methods are Off-the-Shelf Option Method (OTSO), Social Technical Approach to COTS Evaluation Method (STACE), Infrastructure Incremental

Development Approach (IIDA), Ius Ware Method, Procurement Oriented Requirements Engineering Method (PORE) and a technique, which is being used in FAA. Each method addresses different problems on COTS-software development and reflects different properties of COTS-software development process.

The OTSO method[7] is a COTS-software selection method. The main properties of this method include defined, documented process, hierarchical and detailed evaluation criteria decomposition, making alternatives comparable in terms of cost and added value they produce, and use of appropriate techniques for collected data. Evaluation criteria definition shows the central role of the process. Evaluation criteria are defined as selection process progress. OTSO method also assumes the use of Analytical Hierarchy Process (AHP). The AHP is a multiple criteria decision-making technique that is based on the idea of decomposing a multiple criteria decision-making problem into a hierarchy.

STACE[8] is a non-technical properties oriented method as depicted in **Figure-6**. Douglas Kunda and Laurence Brooks has developed this method. Kunda believes that the major causes of most software failures are the human, social and organizational issues rather than technical issues. Secondly the most COTS product selection is based on non-technical factors rather than just technical factors.

The sub factors of social-economic factors are:

- **Business issues:** Cost of adapting and integrating, licensing agreements, maintenance costs, product costs, support costs, technology and training costs.
- **Customer capability:** Customer expectation, and experience; organizational policies and politics.
- **Marketplace variables:** Market trends, product/technology reputation, product/technology restrictions.
- **Vendor capability:** Availability of training and support, vendor certification, reputation and stability

[7] http://mordor.cs.hut.fi/~jkontio/sew20_otso.pdf
[8] http://www.cs.york.ac.uk/mis/docs/ukais 99_chap53_42.pdf

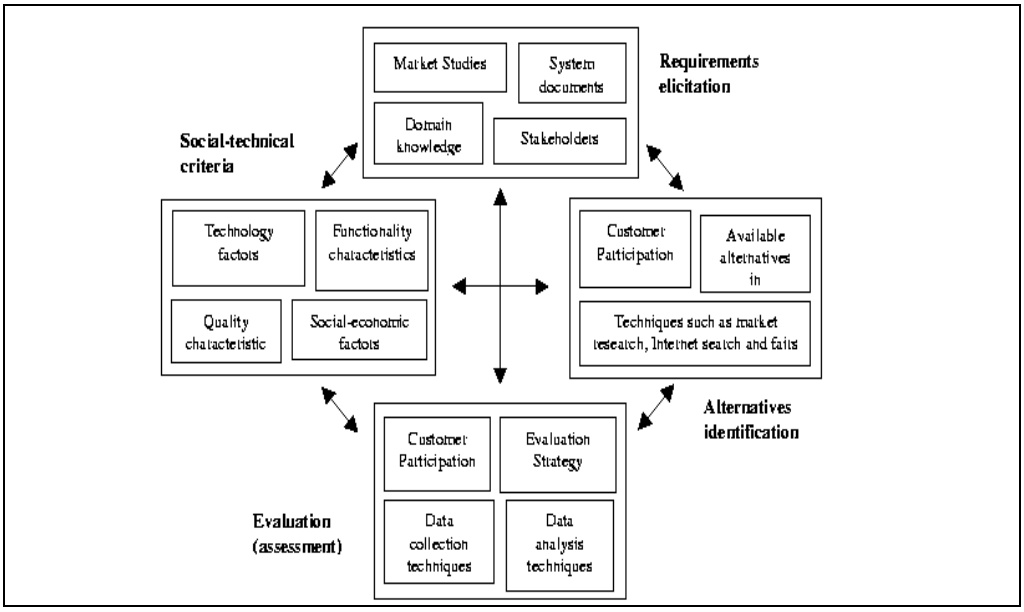


FIGURE-6 STACE FRAMEWORKS

Due to the lack of standards for COTS-software procurement, some organizations, which use COTS-software products, are applying different procurement techniques. FDD's technique is heavily system evaluation, selection and integration oriented. Fifteen projects at the FDD at NASA were studied by Morisio and his friends and the actual COTS process of FDD's was put forward[9]. The fifteen projects ranged from the 29 to 300 KSLOC. Among generic COTS, there were databases, data acquisition, analysis and visualization, Graphical User Interface (GUI) builders, networking and middleware**.** The actual COTS process steps are shown in **Figure-7.**

The Infrastructure Incremental Development Approach (IIDA) method[10] provides a programmatic prototype driven, selection and integration oriented COTS software development approach. This method does not address procurement, selection and evaluation of COTS-software development process. It assumes a selected COTS product and brings solution to the integration of developing project.

[9] http://wwwsel.iit.nrc.ca/projects/cots/icse2000wkshp/Papers/morisio.pdf
[10] http://www.stsc.hill.af.mil/crosstalk/1998/apr/process.asp

Infrastructures between COTS-software, glue code and developing projects are explained. The main stages of IIDA are shown in **Figure-8.**

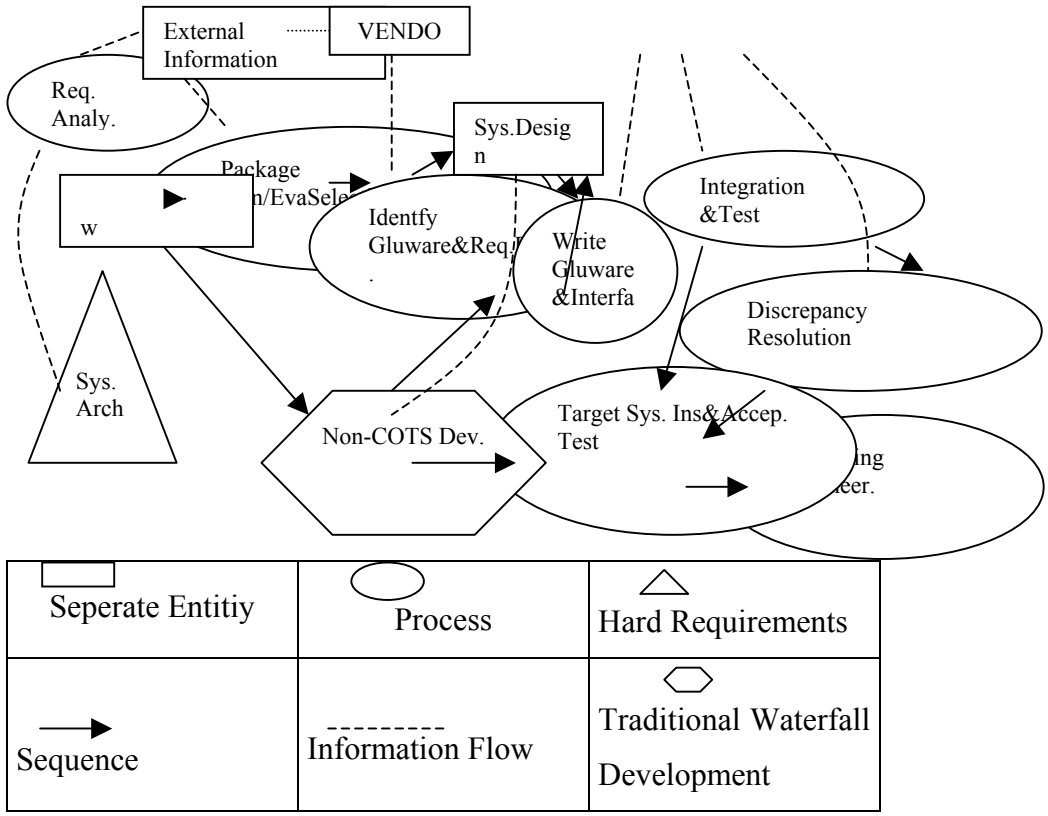


FIGURE-7 THE ACTUAL COTS PROCESS

The Ius Ware method[11] is used for evaluation and selection of software products. Ius Ware method is based on the Multi Criteria Decision Aid (MCDA) approach and encompasses assessment and selection of software products so Ius Ware method does not exactly address COTS-software development. Because of its formal and structural aspect, method provides an expanded view to the COTS-software development. Ius Ware defines an evaluation process that consists of two main phases; these are designing an evaluation model and applying it.

[11] .http://www.morsev.polita.it/Papers/iusware.pdf

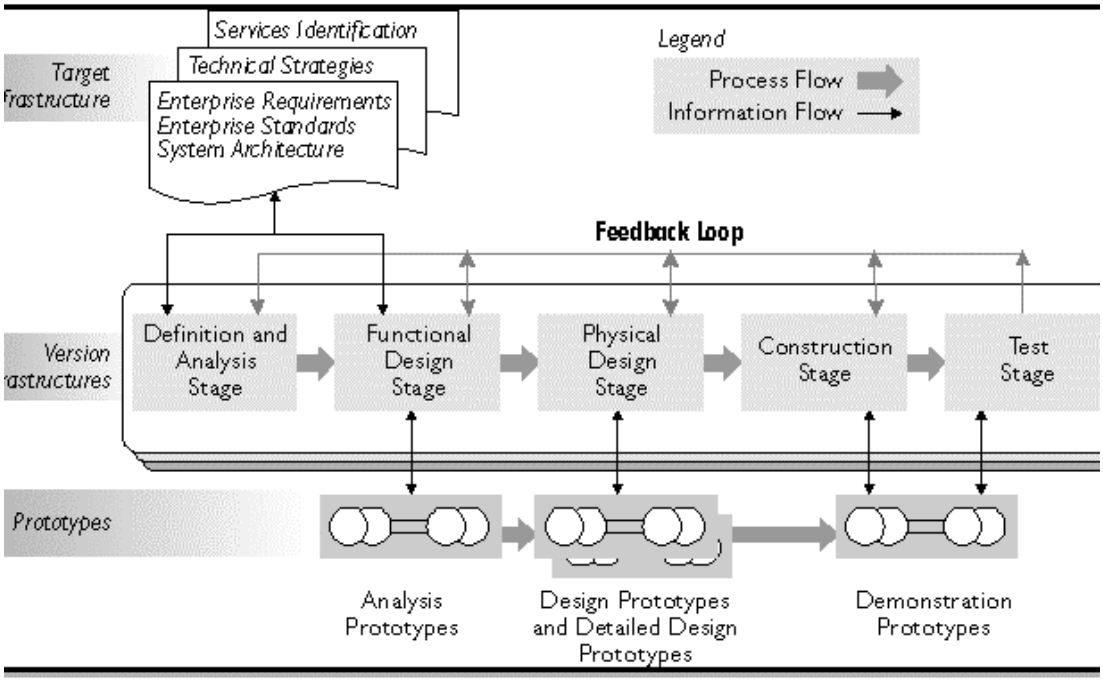


FIGURE-8 IIDA PROCESSES

PORE methodology[12] is developed by Cornelius Ncube. Although it is anew emerging methodology, it has attracted a great attention. It was developed as the first requirements engineering oriented method in COTS-software development domain. In 1998 in the IEEE Software Magazine, Maiden stated the method's evolution with these statements**:**

"To support requirements acquisition for selecting-commercial off-the-shelf products, we propose a method we used recently for selecting a complex COTS-software system that had to comply with over 130 customer requirements. The lessons we learned from that experience refined our design of PORE, a template-based method for requirements acquisition. We report 11 of these lessons, with particular focus on the typical problems that arose and solutions to avoid them in the future. These solutions, we believe, extend state-of the art requirements acquisition techniques to the component-based software engineering process."[13]**.**

PORE method provides iterative requirements acquisition and candidate product selection until one or more products are compliant with sufficient number

[12] http://www.soi.city.ac.uk/~dg571/connie.html
[13] http://www.soi.city.ac.uk/~dg571/connie.html

requirements. PORE defines three necessary goals to choose the most appropriate product. These goals are:

- Atomic customer requirements
- Complex non-atomic requirements
- Non-functional requirements such as architectural, reliability or usability requirements. The requirements engineering team should achieve these goals in a sequence.

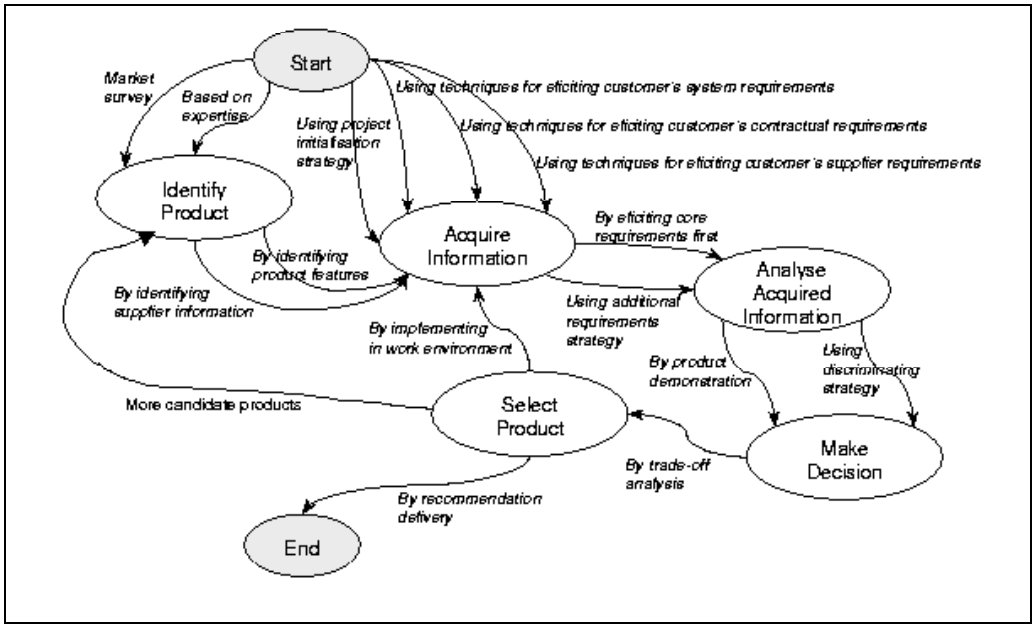


FIGURE-9 PORE PROCESSES

3.2. COTS- SOFTWARE STUDIES

COTS-software studies are heavily conducted by North American countries and England. Many institutes, defense sectors, commercial organizations want to use more COTS-software products in their projects and try to employ more suitable development techniques. In this section SEI, FAA, and DoD in the US and NATO's studies, initiatives about COTS-software usage are examined but especially policies,

guidelines and regulations are explored because of their mandated force over organizations.

3.2.1. Software Engineering Institute (SEI)

SEI has an academic role in COTS-software studies. COTS-software area has a distinct importance according to SEI. Because COTS products as elements of larger systems are becoming increasingly commonplace, due to shrinking budgets, accelerating rates of COTS enhancement, and expanding system requirements. SEI has some COTS activities like research groups, briefings and conferences to the managers, it publishes monograph series about COTS and has developed a COTS Usage Risk Evaluation (CURE) method. CURE is a risk evaluation and mitigation approach aimed specifically at COTS-related issues in acquisition. It consists of a detailed questionnaire and it is not product oriented, it is process oriented. SEI prepares a set of monographs that address issues such as:

- Finding and selecting appropriate commercial products
- Identifying decision criteria for migrating to new or emerging COTS technologies
- Understanding the ramifications of the COTS-Based Solutions (CBS) approach on system architecture
- Developing testing strategies for systems incorporating COTS components.

3.2.2. Federal Aviation Administration (FAA)

FAA has an important place in COTS-software field in the US. FAA uses Radio Technical Commission for Aeronautics (RTCA)'s DO-178B (Software Considerations in Airborne Systems and Equipment Certification) document as a base guideline for software development process activities. This document also includes COTS development orders.

In this section a report, which is submitted by FAA is examined. This report[14] is about COTS usage at safety critical systems and evaluation of COTS-software in the framework of DO-178B guideline. In this report, Krodel examines COTS products in two dimensions. These are wide usage and level of assurance. The wide usage dimension measures the number of different application environments and users of the product. This provides the potential for comparison and possible approval on the basis of in-service operation. At wide usage part, Krodel gives his interview results with some companies, which specialize in safety critical systems (in the avionics, nuclear industry, medical, space, and elevators fields).

From the data obtained, it appears that there is only a small set of COTS components, which are seriously considered for COTS usage in those safety-critical domains. According to Krodel's research, in avionics, a manufacturer of a large airframe reported that they decided not to use COTS products at top level critical software but some line replaceable units under less critical levels planned to use COTS products. Again at nuclear industry Nuclear Regulatory Commission suggests that all COTS products have a safety kernel wrapped around them, not only to prevent inadvertent operation, but also to positively warn and allow the equipment operator to react. At space area, COTS usage is in the absence of the risk to any astronauts, the normal 99.9% reliability coverage was reduced to 97%. By accepting this reliability reduction, the team projected that the COTS aspect of the program reduced their costs by approximately 50%. Except operating systems, network software, compiler libraries COTS products are considered very risky at safety critical area. Level of assurance dimension establishes the source (governmental, industry, third party) of assurance and the standard for that assurance. Main issues of concerning to usage of COTS with DO-178B document are:

- The business relationship between the vendor and applicant,
- Problem reports: Coordination between applicant to vendor has to be provided. Two side, have to share their information about product.

[14] http://www.av-info.faa.gov/software/Research.htm

- Unused or unintended functions: DO-178B provides guidance with regards to both dead code, that is code not reachable due to design error, and deactivated code, that is code which is part of the avionics application, but not enabled for operation.
- Commercial –off the shelf previous environment and operational profile: For COTS products, the operational profile assumed in its design might not be known precisely. Consequently, it is difficult to assess where discrepancies between the design and the specification for COTS usage in a safety critical system might appear.
- Version control: All of the information pertaining to the COTS item to be used should be clearly linked to the specific version to be incorporated into the system.
- New releases: For upgrades additional information needed.
- Product examination: Product development records and operating experience data are the areas that need scrutiny.
- Process examination. Determined 66 product properties showing COTS level of assurance. These levels are A, B, C, and D. Level A and B products are more critical software. Many typical COTS components are more applicable to the software levels C and D. FAA at this point has developed a notice about level D software. Level D is assigned to software that can cause or contribute to no more than a minor aircraft failure condition. FAA states that for low-level requirements DO-178B objectives are stricter, so for level D software previously developed software does not need to meet some DO-178B objectives and N8110.92 notice bringing some regulations and explaining DO-178B's misinterpreted objectives.

The other issue related with COTS usage is verification techniques. Krodel under the DO-178B document gives some examples for verification techniques. These are stress testing, process scenario testing, equivalence-class testing,

boundary-value testing, random-input testing, and performance testing. All these techniques are black box testing techniques.

3.2.3. Department Of Defense's (DoD) COTS-software Studies

As with the many organizations in both the public and private sectors the DoD is committed to a policy of using COTS components in information systems[15]**.** However the DoD also has a long-standing of the security needs for its systems, and the pressure to adopt COTS components can come into conflict with those security needs[16]. DoD has developed a new regulation about acquisition of IS to prevent the problem arising from COTS usage; 5000-2R. Named Major Information System Acquisition Document is examined under the COTS-software development and procurement subjects. This 186- page document has been published in June 2001.It contains the last information about COTS usage in the defense but the most important point is that the regulation has mandatory force over major information systems acquisition. This regulation to be able to follow last technologies, reduce project cycle time and cost, requires using market oriented commercial solutions. Regulation consists of seven chapters. In Chapter-1 regulation's goals are explained, and it is noted that digital environment has to provide Project Manager (PM) to access through a COTS browser. Chapter-2 is about acquisition strategy and here stated that the PM shall use market research as a primary means to determine the availability and suitability of commercial and non-development items.

Chapter-3 is about Test and Evaluation. To provide open system approach, this chapter states that test and evaluation on commercial and non-developmental items shall ensure performance, operational effectiveness, and operational suitability for the military application in the military environment, regardless of the manner of procurement. Test planning for these items shall recognize commercial testing and experience, but nonetheless determine the appropriate developmental test and

[15] http://www.sei.cmu.edu/cbs/papers/monographs/dod-cots-policies/dod-cots-policies.htm
[16] http://www.sei.cmu.edu/cbs/papers/monographs/dod-cots-policies

evaluation, operational test and evaluation, and live fire test and evaluation needed to assure effective performance in the intended operational environment.

Chapter- 4 explains Life Cycle Resources. Chapter 5 is about Program Design and this chapter is the main chapter about COTS software requirements for PM. It is advised PM to use the open systems approach, for achieving the enhance modularity and facilitate systems integration, leveraging commercial investment in new technologies and products, reduce the development cycle time and total life-cycle cost, ensure the system is fully interoperable with all systems with which it must interface, without major modification of existing components. The PM shall base software systems design and development on systems engineering principles, develop architectural based software systems that support open system concepts, exploit COTS computer systems products, and allow incremental improvements based on modular, reusable, extensible software.

The PM shall track COTS software purchases and maintenance licenses, when employing COTS software. The contracting process shall give preference during product selection/evaluation to those vendors who can demonstrate that they took efforts to minimize the security risks associated with foreign nationals that have developed, modified, or remediated the COTS software being offered.

It provides some useful practices for COTS-software products. W*hen acquiring COTS software products or other commercial items, the PM shall implement a spiral development process*. In this context, integration may encompass the amalgamation of multiple COTS components into one deployable system (or block of a system) or the assimilation of a single COTS product (such as an enterprise resource planning system). In either case, the PM shall ensure that the system co-evolves with essential changes to doctrine (for combat systems) or reengineered business processes (for combat support and IT systems). The PM shall apply commercial item best practices.

In addition, when purchasing a commercial item, the PM shall adopt commercial business practices. The extent to which the DoD business practices

match the business practices supported by commercial items determines the likelihood that the items will meet DoD needs. It is likely, however, that a gap will exist—and the gap may be large. Negotiation, flexibility, and communication on the part of the stakeholders, the commercial vendors, and the program manager are required.

The PM shall plan for robust evaluations to assist in fully identifying commercial capabilities, to choose between alternate architectures and designs, *to determine whether new releases continue to meet requirements*, and to ensure that the commercial items function as expected when linked to other system components. In addition, evaluation provides the critical source of information about the trade-offs that must be made between the capabilities of the system to be fielded and the system architecture and design that makes best use of commercial capabilities. *Evaluating commercial items requires a focus on mission accomplishment, and matching the commercial item to system requirements.*

The PM shall remain aware of and influence product enhancements with key commercial item vendors to the extent practical and in compliance with Federal Advisory Committee Act. Vendors are different from contractors and subcontractors, different practices and relationships are needed. Vendors react to the marketplace, not the unique needs of DoD programs. To successfully work with vendors, the PM shall adopt practices and expectations that are similar to other buyers in the marketplace. *Traditional DoD acquisition and business models are not sufficient for programs acquiring commercial items*, as they do not take into account the marketplace factors that motivate vendors.

The last point is about performance specifications, department shall use performance specifications when purchasing new systems, major modifications, upgrades to current systems, and commercial and non-development items for programs in all acquisition categories.

3.2.4. North Atlantic Treaty Organization's (NATO) COTS- Software Studies

The NATO conducts an international defense service. Its needs are not limited with military equipments. Under the NATO there are many research

organizations and their aim is to make NATO more activated organization. Especially Research and Technology Organization (RTO) and NATO Consultation Command and Control Agency (NC3) have great responsibilities about information systems NATO.

In this section RTO and NC3's COTS software practices are explained. RTO is the single association in NATO for Defense Research and Technology area, and has a technical team for conducting researches. This team organizes workshops, symposia, field trials, lecture series and training courses. RTO organized an Information Systems Technology Panel Symposium In April 2000. Symposium's aim was that how COTS-software researches can be applied to safety critical military projects. This symposium addresses NATO interests and issues in employing COTS hardware and software while maintaining required levels of system assurance. After the symposium a report was prepared which consists of 24 papers. Their subjects are:

- COTS-software acquisition, utilization, and evaluation,
- COTS-software acquisition challenges,
- COTS-software evaluation and assurance,
- Vendor perspective,
- User perspective,
- Integration,

NC3 also prepares NATO's mandated documents for information systems and develops interoperable common (between nations) new open system architecture. One of them, AC317-D/71, is a guideline for COTS usage in NATO. Another guideline "Open Systems Interconnection Profile Strategy"[17] has some important statements about NATO's COTS strategy, NATO has decided in its NATO COTS Software Policy and Acquisition Guidelines to prefer COTS software acquisition instead of the development of new software unless it evidently contains major disadvantages. This preference for COTS solution depends on:

- Demonstrable cost effectiveness over the life cycle,

[17] http://www.ipv6forum.com/navbar/reports/nato_nosip98/1nosip01.pdf

- An already existing system into which has to be integrated will not be impacted adversely to an unacceptable degree,
- Security, safety, or time criticality (response) requirements being satisfied,
- Market stability (e.g. vendor's market status and size, product line quality) in the particular area of involvement being shown to exist, in accordance with the assessment criteria outlined in part 2 of AC317-D/71.

In addition to NATO's COTS software preference, NATO COTS Software Acquisition Guidelines defines its approach to evaluating the risks of integrating COTS-software into a system. These risks in the Krodel's paper are identified at four primary areas:

- Complex integration problems, which may lead to failure to meet requirements,
- Cost and resource escalation due to this integration,
- Lack of product control: license agreements, product discontinues or not supported, and release management,
- Increase configuration management due to mixture of custom and COTS-software.

3.3. ASSESSMENT of RELATED WORK

In this related work, studies of some experts, organizations, and institutions on COTS-software usage are investigated. The purpose of this work is to investigate if there is any solution to the problems related with COTS-software requirements elicitation.

COTS-software development methods are primarily examined. It is observed that almost all of the methods address the COTS-software products evaluation, selection, and integration issues for the current developing projects.

Then usage of COTS-software products and studies of the organizations are examined. It is also observed that the organizations focus on the studies to avoid the current risks and attempts to increase ratio of COTS-software products usage.

Whereas, it is possible to find solutions in later phases of the system development process, it is difficult to employ a reliable method in COTS-software requirements elicitation phase of the system development process. This condition enforces the organizations to develop a process in which the COTS-software requirements are elicited. Thus, a process from the COTS-software requirements elicitation to COTS-software products integration and tracking phases will be defined.

CHAPTER 4

4. COTS-SOFTWARE REQUIREMENTS ELICITATION FROM BUSINESS PROCESS MODELS (CREB) METHOD

In this chapter, proposed CREB method is explained. CREB's main idea is to use Business Process Models for obtaining requirements in early phases of the software intensive automation system's procurement. The requirement set has to match with users expectations. This requirement set's focusing area is COTS-software products.

CREB's scope, knowledge needed and experience for application are explained in section 4.1. Method's phases are described in section 4.2. The notations for business process modeling are given in section 4.3.

4.1. Scope of the CREB Method

Acquisition activities start with a mission need determination phase and ends with the product acceptance by the acquirer. All acquisition activities can be performed under two main sub-sets. First set is named with system activities and conducted directly by the acquirer. Mission need determination, concept exploration, system definition, system design, and RFP preparation consists of system activities. Second set is named with Procurement Activities and conducted by both acquirer and supplier. Some RFP preparation issues are, contract acting, supplier monitoring, and product acceptance consist of procurement activities.

Acquisition of a large and complex IS necessitates a long life cycle to be able to examine and gather all requirements. "The first step in determining requirements is to detail the scope and targets of the proposed projects"**[Bilgen, 1992]**. The CREB

Method's scope is kept in the system activities, after the **Concept Exploration and before the RFP preparation.** This scope is shown in **Figure-10.**

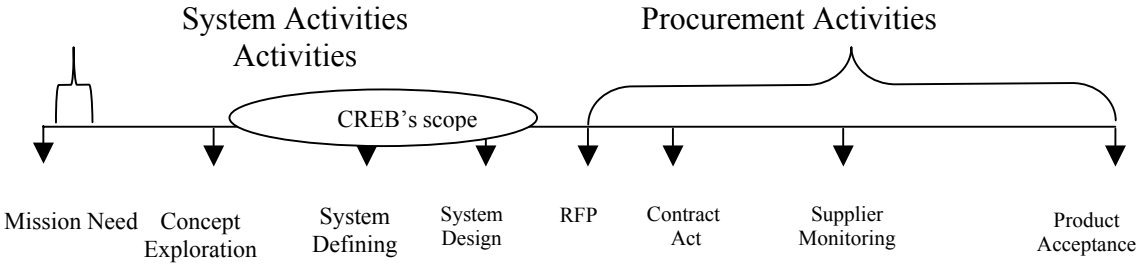


FIGURE-10 PROJECT LIFE CYCLE

The progress between the Concept Exploration and RFP preparation is examined and COTS-software product's requirements are primarily specified for the acquirer.

4.1.1 Audience Knowledge and Experience

Application of CREB method needs some skills and experiences. These are:

-**Decision Making Techniques**: Make/Buy Decision used in fourth phase is an important part of CREB. Some decision-making techniques like MCDA, and AHP can be used in the Make/Buy Decision phase.

-**Software Quality Attributes**: The domain knowledge on SQA and awareness of available standards helps to elicit non-functional requirements.

-**Business Process Modeling**: Method's core activity is modeling, so developer has to establish a team who has modeling skills and experience.

-**System Engineering and Requirements Engineering Activities**: Since CREB deals with system activity scope it needs experienced system engineers. It is important to hire team members for requirements elicitation activity, who worked on this domain before and are aware of common standards.

-**Acquisition Activities**: Method is applied to software intensive system's acquisition. Moreover, COTS-software usage increases the importance of acquisition activities.

-**COTS-software Products' Market Research**: Information about COTS-software databanks, which provide efficient time usage.

-**Information on In-house Repositories and Libraries**

4.2. CREB PROCESS

Before the contract phase, CREB method prescribes 5 generic processes, which are essential to undertake the iterative process for elicitation of COTS-software requirements:

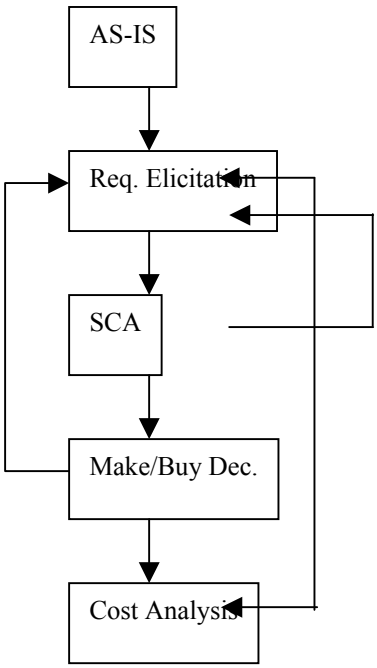
1-Existing System Modeling (AS-IS Model),

2-System Requirements Elicitation,

3-System Component Allocation (SCA),

4-Make/Buy Decisions,

5-Cost Analysis,



CREB Method's phases and every phase's Goals, Inputs, Outputs, Mechanisms, and Functions are explained in this section's following parts.

4.2.1. Existing System Modeling

Goal: This phase defines and models current system's organizational structure, business processes and the data flow for new automation system to be adapted.
Input: Mission Need Analysis Document, Operational Requirements Document, Master Plans, Internal Guidelines and Regulations, End-User Participation.
Mechanism: Business Process Modeling, Structural and Open-Ended Interview.
Function: A new system can be intelligently designed with a through description of the underlying system. Developer uses defined Input and Mechanism items for the description of activity.

Firstly, related system data is gathered from the documents prepared previously and end-user's active participation. These documents are Mission Need Analysis, Operational Requirements documents, other available organizational documents like regulations, guidelines, and charts. They help developer to understand the existing system and its functionality. System activities are divided into sub-systems; and the model developer starts to examine each subsystem. Structural and Open-Ended Interviews or observations help model developer to provide more detailed and real information about system and subsystems.

The obtained data, by the way of a business process model transfers to extended event process chain (EEPC) diagrams. All system activities, which would be in the boundary of automation, have to be modeled with EEPC diagrams. These EEPC diagrams become the AS-IS model of the system. AS-IS model has to be approved by the acquirer and end users. The notations, which are used at CREB and some examples for EEPC diagrams, are explained in section 4.3.
Output: AS-IS Model

4.2.2. System Requirements Elicitation

Goal: Elicitation of functional and non-functional requirements.

Input: AS-IS Model, Existing components.

Mechanism: Sessions, Interviews, and Standards.

Function: Functional requirements describe the behaviors (functions or services) of the system that support user goals, tasks, or activities. Non-functional requirements include constraints and qualities**.**

System's functional requirements are mainly obtained from EEPC diagrams. EEPC diagrams allow developers to see and identify the processes to be automated in AS-IS model. But prepared EEPC diagrams' detail levels are very important to gather requirements. All produced documents, any responsibilities in the system, used resources, and the data exchange points have to be detailed in diagrams.

However, the generation of requirements from diagrams requires experience, domain knowledge, and understanding of user's demands. Elicitation of system requirements starts with the examining of each sub-system's functions. Automation points and optimizations are determined through acquirer, developer, and end user's co-operative studies. This study's work product is named on TO-BE model document, which consists of all functional requirements.

Another requirement elicitation source is using existing components properties. Because choosing a system means, finding the product, closest match between the system requirements and the facilities offered by off-the-shelf systems. These existing components' properties can be found in in-house libraries or in the market and their functionalities can be adapted for the requirements' elicitation phase.

Some BPM tools allow computer-aided requirements' generation, but in CREB, this method hasn't been used, as automatic requirements generation needs, TO-BE model to be prepared in advance. This means that component design has to be finished and allocated to AS-IS model, but it can be very hard to determine system components in early phases with available data. So the component design should be performed after the requirements elicitation activity.

At the same time, when preparing EEPC diagrams, non-functional requirements (efficiency, legislative issues, company information, reliability, portability, performance, and security) are gathered from end-users, and from

acquirer via interviews. In addition to internal sources, some external sources like standards (ISO/IEC 9126 "Information Technology- Software Product Evaluation Quality Characteristics and Guidelines for their use", AQAP-150), and existing component's specifications, are used to identify non-functional requirements.

All functional and non-functional requirements gathered together will be included in the System Requirements Specification Document, which should be approved by the acquirer. The specification's traceability property is the most important value for proceeding phase, because the components will be allocated according to these specifications. The developer to provide requirements traceability, can use some traceability tables like subsystem traceability (categorizes requirements by the subsystems that they govern) table, dependency traceability (indicates how requirements are related to one another) table, source traceability (identifies the source of each requirements) table**.**

Output: TO-BE Document**,** System Requirements Specification, Traceability Tables.

4.2.3. System Components Allocation

Goal: Determination of system components matching with elicited requirements.

Input: System Requirements Specification.

Mechanism: System Components Allocation Matrix.

Function: The architecture encompasses four distinct system components: software, hardware, data, and people. After the System Requirements Elicitation phase, the CREB method allocates previously gathered functional and non-functional requirements to each of these four components. Each of these components is an engineering discipline, but there is an active communication between them.

It is better to use a matrix for the allocation activity. The required communication and the relation among components can be easily tracked via a matrix. The matrix consists of requirement lines and system component columns. Previously determined sub-systems' components are identified according to software, hardware, data, and people elements. For this purpose each requirement in

System Requirements Specification is examined and information needed is gathered via some questions. For example, “is this requirement satisfied by any software or hardware?” and “what kind of software functionality satisfies this requirement”. The components can be identified through these questions that satisfying the system and software requirements.

At this phase, CREB focuses on requirements that are allocated to software components. Software components, to which requirements are allocated, are divided into more specific software components like office software, operating system, database, security software etc. This division will be useful for later steps.

In this phase, some well-known COTS products can be allocated with requirements. Generally, these COTS products are available in the organization or used by the acquirer. The early COTS product definition provides open system architecture. However, limitations of available COTS-software usage mean that requirements have to be modified.

Output: Software Components, System Components Matrix.

4.2.4. Make/Buy Decision

Goal: Determination of developmental and non-developmental software items.

Input: Software Components.

Mechanism: Decision Tree Analyze, Decision Making Techniques, and Market Research.

Function: There are a number of business quality goals that shape frequently a system’s architecture**.** In this phase, the defined software component’s procurement methods are analyzed from the business view by acquirer. This means which software component will be development item (DI) and which component will be non-development item (NDI).

Making an early classification of DI and NDI or make/buy decision prevents extra development effort, and refines functional and non-functional requirements. So, “make” definition is used for scratch-development and “buy” definition is used for commercial solutions. The trade-off between building the item in-house and purchasing it by using an external source is commonly called a make-buy decision.

In first and second phase of CREB, usable and available in-house components were identified. If acquirer has some licensed COTS-software products, after the detail requirements analyzing he/she can decide to use these components. But in this phase acquirer starts market research to determine other defined and unavailable software components. Found COTS-software products and their properties should be recorded. All functional and non-functional requirements of the components are documented at this step. Acquirer should prepare common COTS-software product features list for each system component. It is rare for existing components to match requirements exactly.

Later some techniques can be used for make/buy decision. One of them is decision tree analyze**.** In a decision tree, it is possible to see the development cost and schedule data of each fully developed, reused, or purchased system component. But only cost and schedule data are not enough for a reliable decision.

It is necessary to make a functionality comparison among procurement methods. This functionality comparison and previous cost/schedule analyze provides necessary data for make/buy decision. In CREB method, any decision- making technique is not offered or enforced for make/buy decision. All these decision-making activities can take a long time period, so the acquirer's experience, market skill, and acquired information from other organizations could be summated in this period. The "make/buy" decision forms four situations according to availability and functionality attributes. These are:

1- No COTS-software products can be found in market and/or in-house, which doesn't match with any system's software component requirements.
2- Some COTS-software products can be found in market and/or in-house, which matches with some system's software component requirements.
3- Some COTS-software products can be found in market and/or in-house, which matches with all system's software component requirements.
4- Some COTS-software products can be found in market and/or in-house, which matches with all system's software component requirements and have extra features.

Except first situation, all other three situations enforce the acquirer for the refinement of the system's requirements and SCA matrix. For example, at fourth situation, if "extra functionality doesn't bring extra load", the acquirer has to think about getting benefit from this situation and has to upgrade system requirements.

In contrast, at second situation, if missing functionality doesn't threat the project and gets some profit like cost, schedule, reliability, e.g., the acquirer can consider reducing the system's requirements to get benefit from commercial solutions.

The customer and end-users' approval is again necessary at this step for refined requirements. At the same time there are some risks at this phase. One of them is that the acquirer could add some specific requirements, which result in fully development decision or COTS-software product selection. Another risk is the lack of skill and experience on decision-making techniques. The last and the most important risk is that a wrong decision could threat the project's progress very dramatically.

Output: COTS-software components and requirements, Refined System Requirements Specification

4.2.5. Cost Analysis

Goal: Elicitation of contract based requirements via cost analysis results.

Input: Work Breakdown Structure (WBS)

Mechanism: Estimation Techniques like Function Point (FP) analysis, COCOMO, and Market Research

Function: In acquisition side there are four general classes of requirements; technical, management, corporate, and financial. Until this phase, technical, and corporate COTS-software requirements are elicited while at the previous phase (make/buy decision) restricted financial requirements are analyzed. However, the whole system's financial and management requirements weren't estimated yet. This means that financial and managerial requirements for RFP preparation will affect COTS-software requirements' set.

The system procurement's cost is defined by financial requirements and the schedule by management requirements. The estimation activity necessitates modular architecture of system. WBS can be used for this purpose. The technique of WBS, which structures systematically the work associated with the entire engineering process, is useful in analyzing complex systems. The estimation that is made based on the WBS provides meaningful results about the project.

The CREB method does not offer any technique for cost/schedule estimation. If estimated cost and schedule exceeds allocated budget, the acquirer has to re-analyze this situation. The Acquirer has three options. One of them is refining some system requirements. Another option is trying to increase budget. Finally, delay or cancel the project. The first option also will change the elicited COTS software requirements, so method's phases should be reviewed and reapplied. This chain will continue until RFP preparing activity.

Output: RFP based COTS-software Requirements, Cost Results.

4.3. NOTATIONS USED

There is a business process model modeling technique at the heart of the CREB method for COTS-software requirements elicitation. General features of the process-modeling tool used in CREB method application have been noted in chapter 2. Other specific notations will be presented in this section. In CREB method, process modeling tool's *requirements' description* part is only used and system design activity is conducted by another method, which is explained in section 4.2.

Requirements definition is more important and long-term activity of modeling. The requirements definition has to describe the business application to be supported in such a formalized language so that it can be used as the starting point for a consistent translation into information technology**.** The descriptive views and some object types, which belong to requirements definitions, are detailed.

4.3.1. Requirements Definition: The Function View

A function is a subject-related task or action, performed on an object and aiming at supporting one or more aims of the company **[Scheer, 1992].** There is a function for each process, which describes the **"**what" question. Modeling methods often display functions within the framework of objects from the other descriptive views of BPM. Displaying the relationship between data and functions, for example, allows specifying the transformation process of a function via the input/output data of that function. Functions are displayed as rectangles with rounded corners as shown in **Figure- 12.**

FIGURE-12 FUNCTION DESCRIPTION

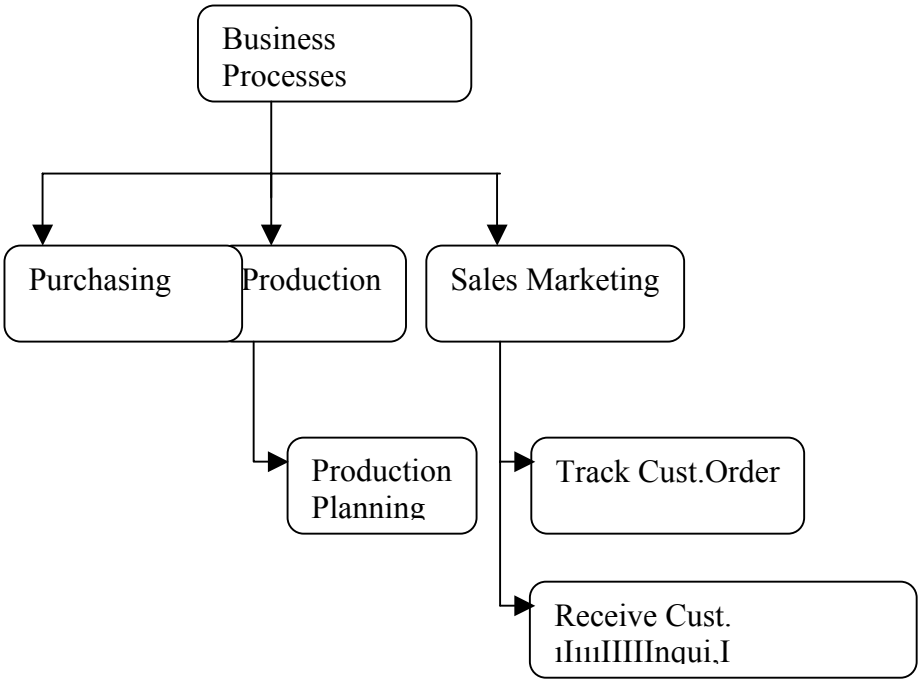


FIGURE-13 FUNCTION TREE DIAGRAM

Functions can be described in different compression levels. A complex function consists of sub-functions and such a description is named with function trees. Hence, the *function* term can be used on all hierarchy levels. Other terms are also used to describe the hierarchy level in a more descriptive way, transaction, process, sub-function or basic function. Dividing functions into their elements can involve several hierarchy levels. Basic functions represent the lowest level in semantic function trees. Basic functions are functions, which cannot be divided up any further for business process analysis**.** Function trees or hierarchy diagrams are used to represent this substructure **(Figure-13).** Function trees serve the purpose of reducing the complexity.

4.3.2. Requirements Definition: The Data View

The requirements definition of the data view includes a description of the semantic data model of the field to be examined. According to the BPM division principle, this description contains both the objects, which specify the start and end events of a process chain as well as the status descriptions of a process chain's relevant environment. It has been found that data requirements definition is playing an increasingly important role in information system development **[Scheer, 1994]**.

The conceptual data models can be formed from the requirements perspective. Than, they can be extended. The approach for building the conceptual data model from the requirements is as follows. The data model representing the particular data cluster consists of some reports. These reports are firstly represented in EEPC diagrams and data designer collects all these reports and their detailed contents. Than, data designer examines all reports' attributes by one by. The super-type entity contains the generic attributes existing in all reports. The attributes are the items in the reports. Other report entities called with their report names are the subtypes of the super-type entity. These entities inherit the attributes of the super-type entity. The generic attributes existing in several different report entities are grouped into entities. After the relationships between the entities are determined, the logical data model is formed.

4.3.3. Requirements Definition: The Organization View

Actually, the organization view is the component, which allows the analysis of a company's organizational structure. A company is a complex social structure, which is divided into handy units. In order to cope with this complexity, structural patterns are defined and rules are established. The result of this process is called organization. The business process modeling architecture provides an independent descriptive view for the organizational structure. In a company's structural organization business process modeling differentiate between organizational structure and procedural organization. The organizational structure encompasses the rules by which the company is statically structured. The procedural organization contains those rules aimed at the tasks to be fulfilled by the company. This task-related structure in the sense of distributing functions to task performers is expressed in the control view of the business process modeling architecture.

The organizational chart is a typical form of representing organizational structures. A chart of this kind reflects the organizational units (as task performers) and their interrelationships, depending on the selected structuring criteria.

Organizational units are task performers, which are carrying out the tasks that must be performed in order to reach the business objectives. The relationships are the links between the organizational units. An example illustrating this is shown in **Figure-14**. Whereas the functional responsibilities are shown in boxes, the organizational chart illustrates the distribution of the business tasks.

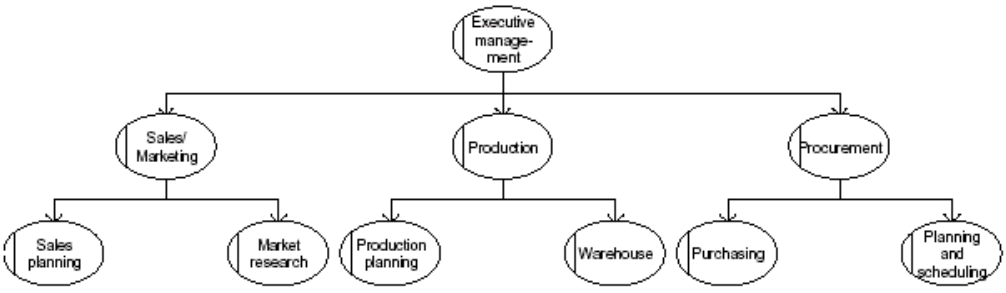


FIGURE-14 ORGANIZATIONAL CHART

In order to furnish a proper job description for the individual jobs within a company, an independent object type *position* is available. One organizational unit can be assigned multiple positions. The meaning of the connections corresponds to that among the organizational units. The positions and organizational units can be assigned persons, which are holding the positions in question. The association of a person with an organizational unit expresses the state that this person is an assigned employee of the organizational unit. The association with an individual position, on the other hand, defines the present job cover within the company Organizational units and persons can also be assigned a type. Thus, it can be defined whether an organizational unit is a department, a main department or a group; employees can be assigned the person types department head, group leader or project manager, for example. The modeling of the company's organizational structure is the starting point for the network topologies which are to be defined at the design specification level and which are supposed to support the organizational structure in the best possible manner. Thus, the location of an organizational unit is the most important link between its requirements definition and the design specification.

4.3.4. Requirements Definition: The Control View

The interrelationships between the objects of the data view, the organization view and the function view are named as control view.

The link between the function view and the organization view serves the purpose of assigning the functions defined in the function tree to the organizational units in the organizational chart. This assignment defines an organizational unit's responsibility and decision- making power for its allocated functions. **Figure-15** shows an example for the allocation of organizational units to functions. In this figure, the individual functions are situated on the left hand side and are assigned the organizational unit responsible for executing a particular function. The functions' position in the hierarchy has been illustrated in the function tree, the interrelations between the organizational units has been shown in the organizational chart.

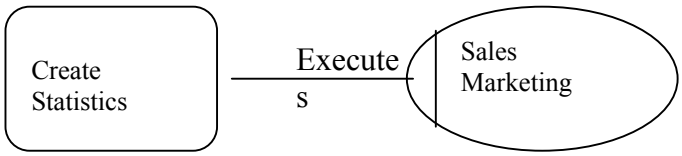
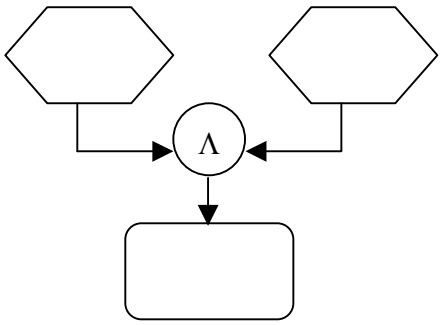


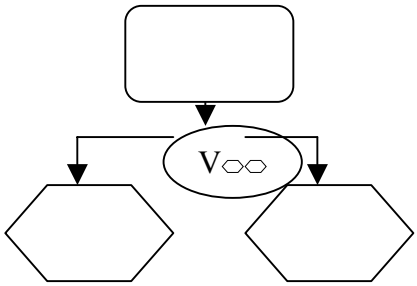
FIGURE -15 ALLOCATION OF ORGANIZATIONAL UNITS TO FUNCTIONS

The procedural sequence of functions in the sense of business processes is expressed in the form of process chains where the start and end events for every function can be specified. Events not only trigger function but also are results of functions. An event is an information object has taken on a business-relevant status, which is controlling or influencing the further procedure of the business process. Events are triggering functions and can be the result of functions. In contrast to a function, which is a time-consuming occurrence, an event is limited to one point in time. Events are graphically represented as hexagons. By arranging a combination of events and functions in a sequence, so called event-driven process chains (EEPCs) are created. By means of an event-driven process chain the procedure of a business process is described as a logic chain of events. The EEPC representation has been heavily used in development of CREB method. Examples for EEPC diagrams are shown in **Figure-19**. Since events determine which state or relationship will trigger a function and which status will mark the end of a function, the starting and end nodes of an EEPC are always formed by events. Several functions can originate simultaneously from one event; conversely, a function can result in several events.

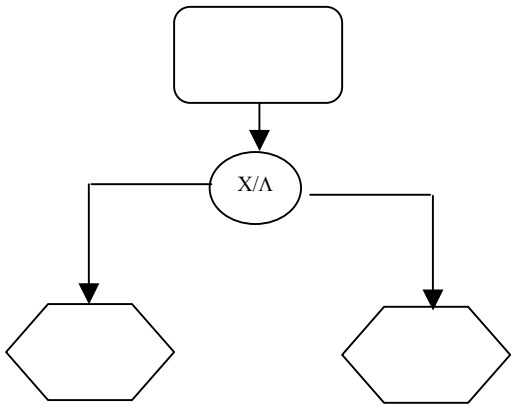
Links:



AND link



OR link

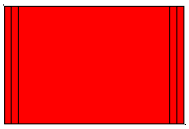
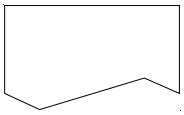
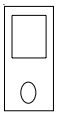


XOR link

FIGURE-16 LOGICAL OPERATORS

Logical operators describe the link between events. In the first example of Figure-16 the starting events are linked by an AND link. This means both events must have occurred before the procedure can begin. The second example shows an *either/or* link (exclusive OR-XOR). The function may either result in accepting or in rejecting this quote. Both results, however, cannot occur at the same time. At the inclusive OR, the function may either result in accepting or rejecting this quote, and the results can occur at the same time.

4.3.5. Object Types

Object Type	Symbol	Definition
Cluster	 (RED)	A cluster instance is a concrete instance of the Cluster/Data model object. It represents a logical view on a collection of data objects or structures.
Document	 (WHİTE)	As an information carrier it is a physical input and output to the systems. Data on it is entered to the computerized system manually and data on it can be produced on physical document.
Application System Type	 (BLUE)	It represents the identification of individual application systems, which have exactly the same technological basis.
Folder	 (WHİTE)	We can consider it as an input and output to the systems. Data in it is entered to the computerized system manually and data in it can be produced on physical folder.
General Resource	 (ORANGE)	A general resource is a resource that does not need to be a person or an operating resource and is not explicitly defined. The general resource allows performing processes.

4.3.6. CREB Method Process Modeling

Notations that are used in CREB have been explained in the previous section. In this section, CREB Method's process definition is denoted by the ARIS notation. It is useful to use some famous models like IDEF, ETWX, or SADT for the process definition. Application of those common methods increases the usability **[Wiegers,**

1999] attributes. In this thesis, business process modeling notation is primarily used at CREB method's Existing System Definition phase and than for method's process definition. The notation is useful for the process definition, and allows representing many parameters, like organizational structure, produced documents, and used resources, complex functionality e.g.. In **Figure-17,** CREB's context diagram is shown. In context diagram via a function tree, it is possible to reach **Figure-18**. This figure consists of CREB's five main steps. Every step or a phase has a function tree for detailed EEPC diagrams. For example in **Figure-19,** it is possible to see all functionality of Existing System Modeling (see sec.4.3.1) phase via EEPC diagrams.

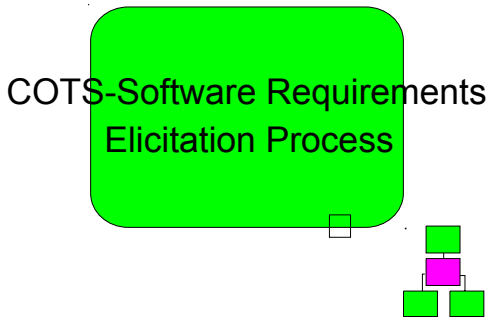


FIGURE-17 CREB CONTEXT DIAGRAM

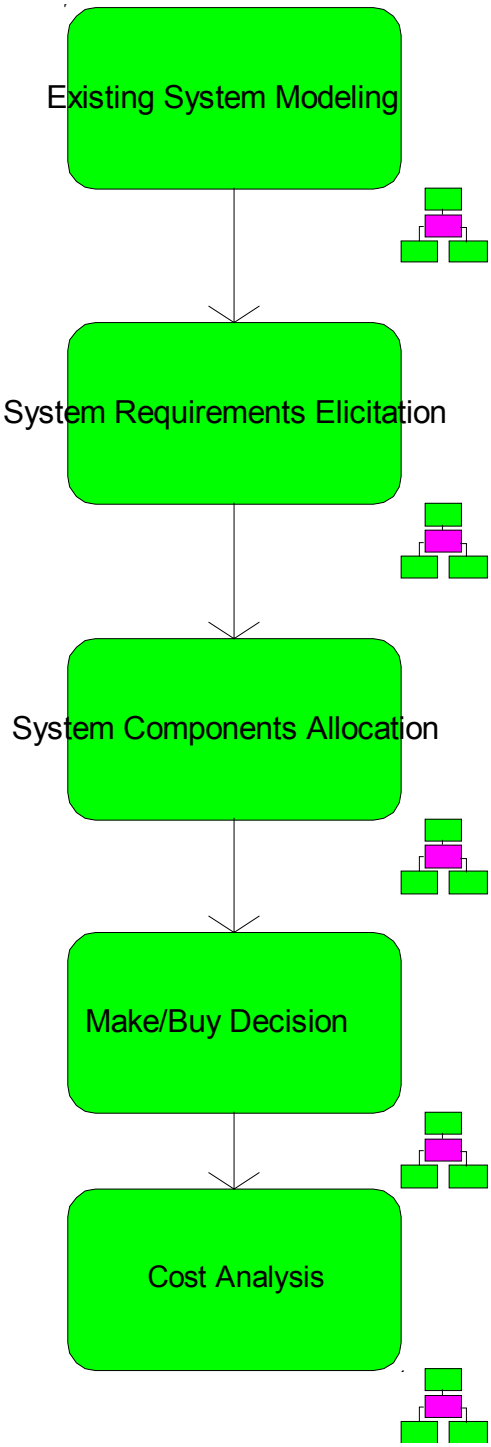


FIGURE-18 CREB PROCESS

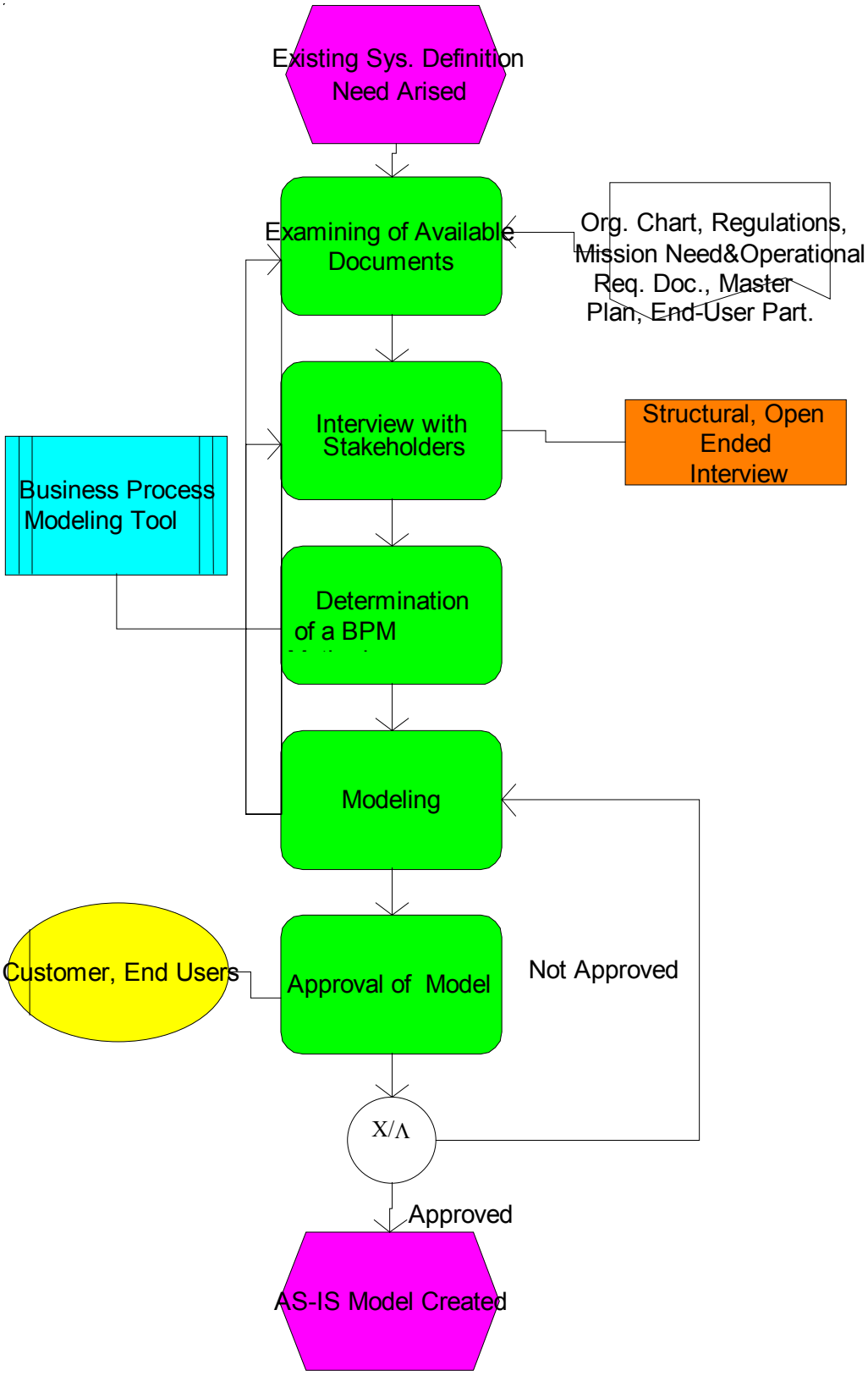


FIGURE-19 EXISTING SYSTEM MODELLING EEPC DIAGRAM

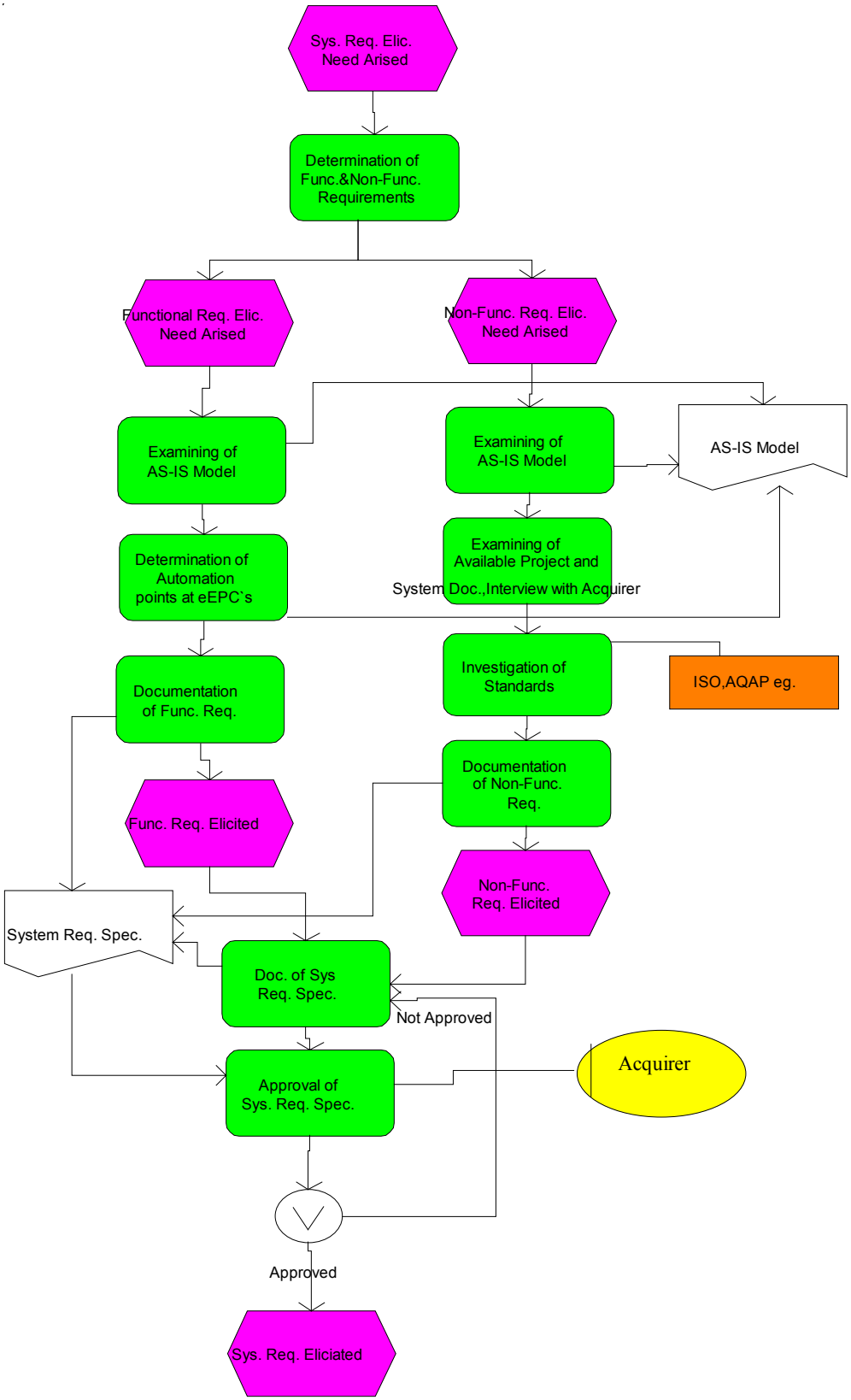


FIGURE-20 REQUIREMENTS ELICITATION EEPC DIAGRAM

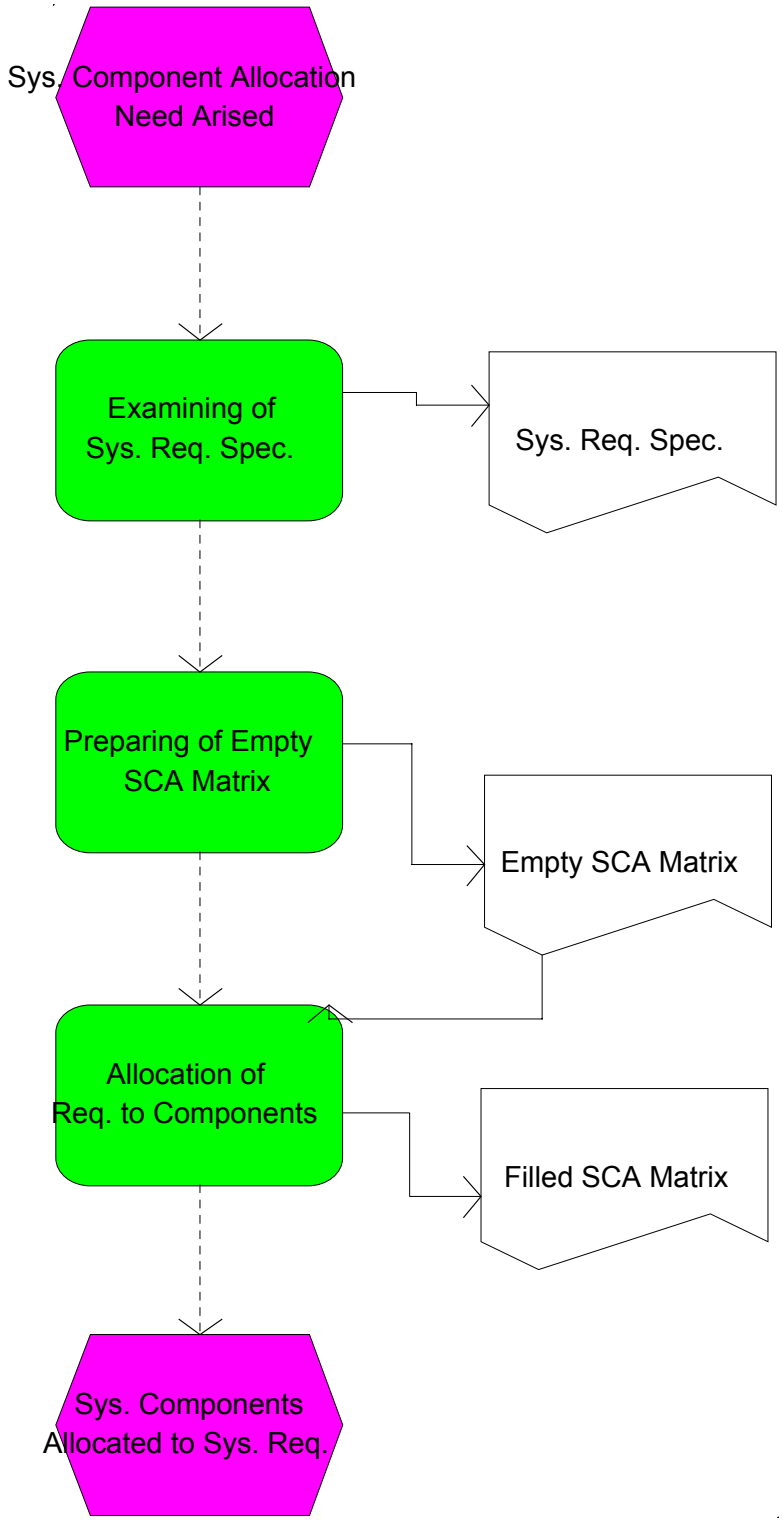


FIGURE-21 SYSTEM COMPONENTS ALLOCATION EEPC DIAGRAM

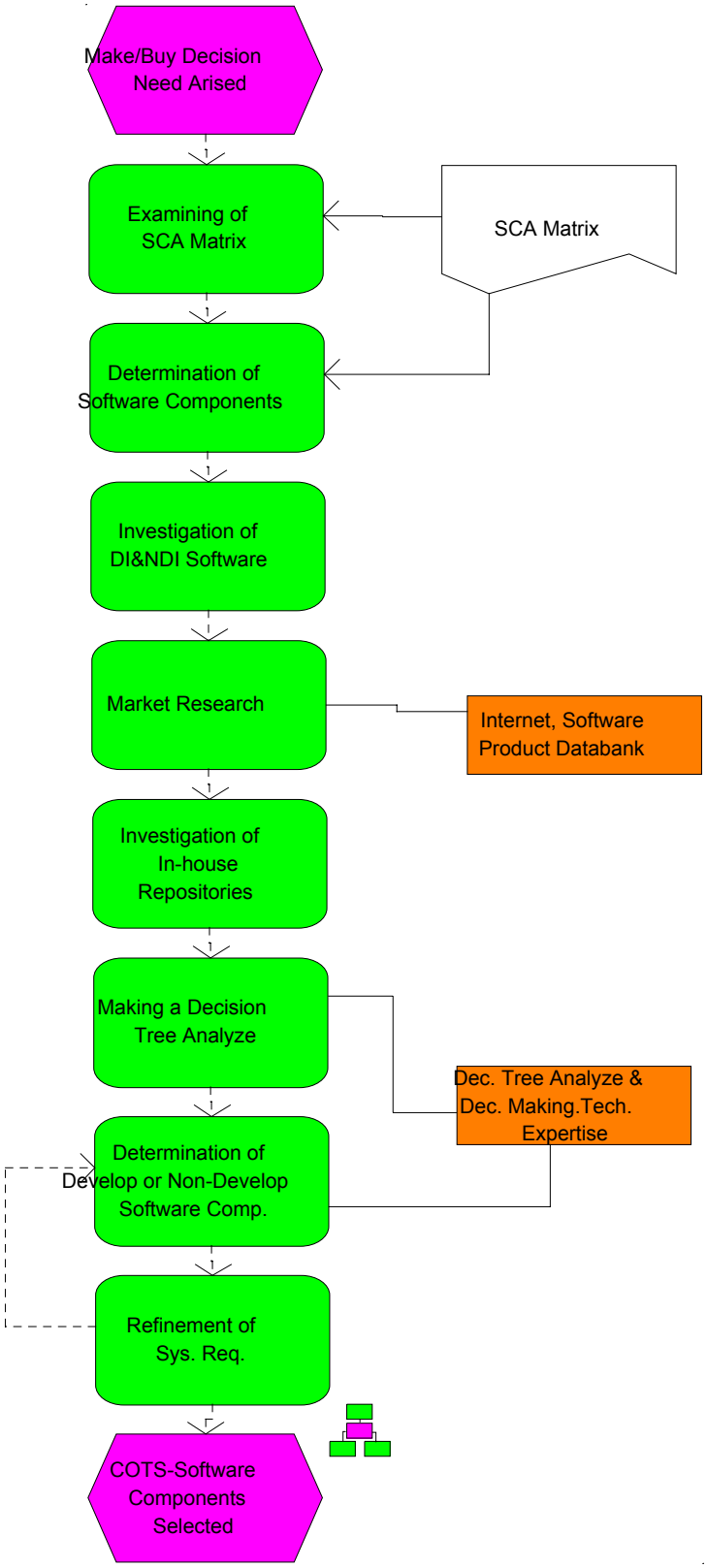


FIGURE-22 MAKE/BUY DECISION EEPC DIAGRAM

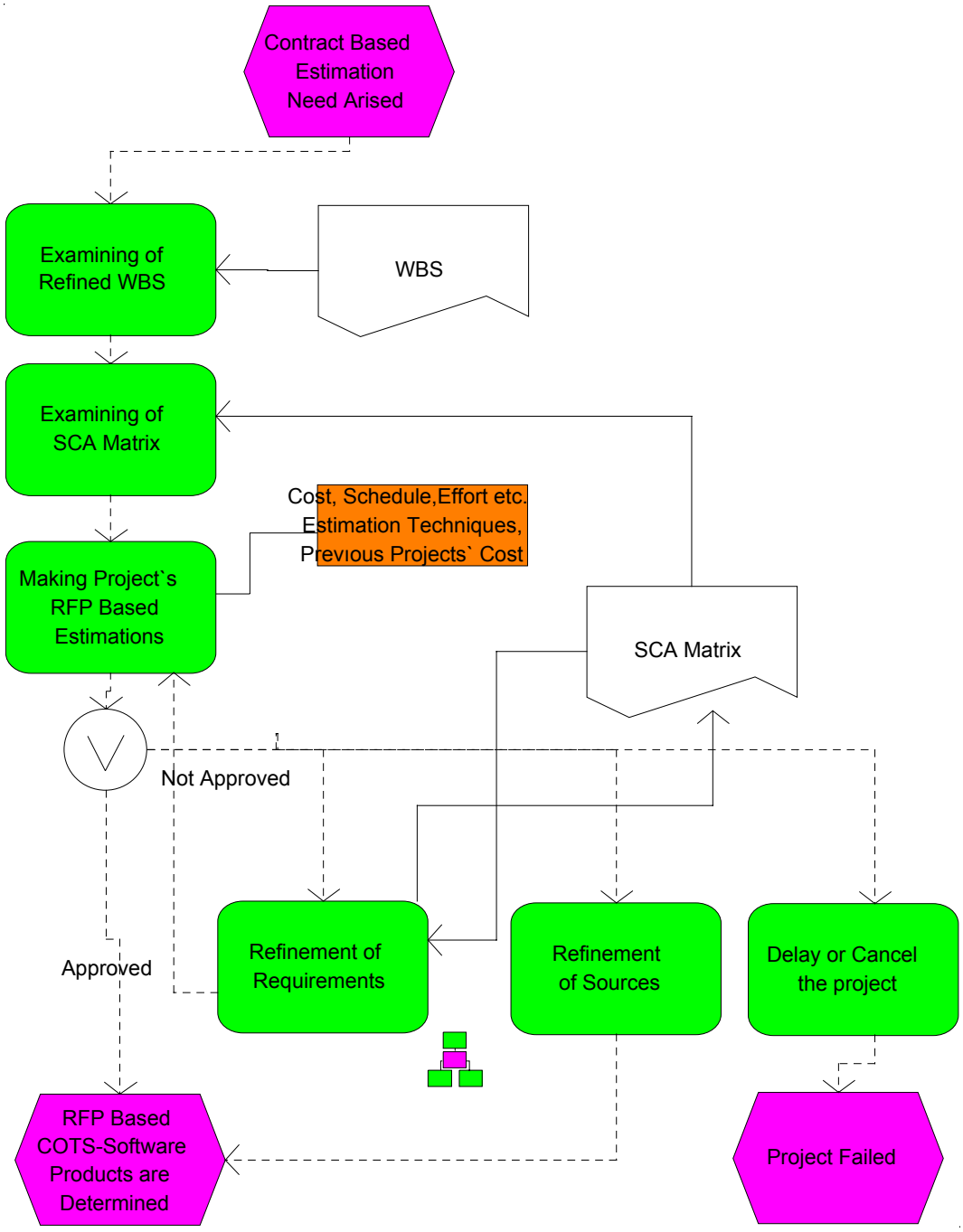


FIGURE-23 COST ANALYSIS EEPC DIAGRAM

CHAPTER 5

5. APPLICATION OF THE CREB

5.1. INTRODUCTION

The CREB application has been performed to provide better understanding of the method's applicability also to obtain enough data about COTS-software requirements elicitation. That is, the method is modified based on the results of this application.

The method is applied to an acquisition project of a software intensive information system. In the application there is an acquirer project office, and a third party organization that is responsible for the contract preparation. In section 5.2 implementation design, in section 5.3 implementation progress, and in section 5.4 assessment of prepared technical contract are presented. Finally, section 5.5 consists of the results from the application.

5.2. IMPLEMENTATION DESIGN

The main purpose of this thesis, which reviews the COTS-software requirements elicitation from business process models, is examined in a case study. The case's applied organization, project properties, data collection method, and anonymity information are provided in this section.

5.2.1. Organization

All participants of the case study were from the different sectors. Examined process was applied in an intelligent unit of TLFC. Requirements Elicitation, contract preparation activities are conducted by the IS department members of the Middle East technical University (METU). For the software intensive IS acquisition

activity, TLFC has started a project about three years ago and during these three years a Project Definition Document, and a Master Plan have been prepared. A stable project Working Group that consists of three members is established in this period and then this Working Group is replaced with a Project Office (PO). PO members have consisted of four participants. These members were the Project Manager and three System Engineers. Two of System Engineers had adequate IT knowledge. But the Project Office members didn't have any experience in acquisition activities. The University, prepared necessary documents, conducted pre-studies of RFP preparation activity and prepared project's technical contract. There were 12 personnel from Electrical Engineering, Information Systems, and Software Engineering disciplines who established the Project Group. Three personnel of Project Group had about twenty years experience, two of them had about four years experience, one participant was one year, and six personnel had no experience.

5.2.2. Project

The project's goal was the adaptation of a new software intensive automation system. It was expecting from the system to automate operational area activities by digitally modeling the complex battlefield.

The project's acquisition progresses were planned in seven sequential phases. These are:

1. Project Definition,
2. Master Planning and Technical Architecture Design,
3. Model System Acquisition,
4. Model System Testing,
5. System Re-Evaluation and Designing,
6. Model System Demonstration,
7. System Expanding.

Project Definition and Master Planning and Technical Architecture Design phases had already been completed by TLFC. TLFC hired the METU's mentioned Project Group for performing the Model System Acquisition phase's necessary

activities (solicitation package's preparation activity). Eight months were given to Project Group, for the Model System's Acquisition.

5.2.3. Implementation Anonymity

For the comfortability of the organizations and to increase the willingness of the participants we paid of most attention to the anonymity and confidentiality of the organizations, persons, and the project.

5.2.4. Data Collection Method

Data collection became very easy to execute due to the recorder's active participation in the project. However, this participation, according to Yin, raises some biases like disregarding of faults and huge amount of supervision over the project **[Yin, 1994].** Project's history, prepared Master Plan, and Operational Requirements' Document are supplied by the Project Office. Meetings, reviews, and studies are chronologically documented and stored in digital environment.

5.3. IMPLEMENTATION PROGRESS

5.3.1. APPROACH

At the Model System's Acquisition Phase, Project Group, who was given contract preparation responsibility, conducted project studies following three milestones. These milestones are explained in this section from the COTS-software requirements elicitation point of view. These milestones are:

1- Pre-Study of Acquisition Package Preparation,
2- Detailing of Acquisition Package Sub-Functions,
3- Preparation of Acquisition Package Requirements,

Until first milestone, the following initial activities:

-Establishing a Project Group, sub-research groups,
-Collecting available documents, provided by Project Office,
-Reviewing collected available and pre-prepared documents,
-Discussing the system definition, were conducted.

The primarily result of this first milestone is that the acquirer's expected ratio of COTS-software usage is very high. But in advance, some software components are kept out of the packaged solutions due to security considerations and technology availability demands.

Application of the CREB Method became possible in the second and third milestones. In this section, the studies of the Project Group between the first and third milestones are evaluated from the standpoint of obtaining of COTS-software requirements and applicability of the developed method. The CREB method is applied to the project, phase by phase during this evaluation. It is aimed to attain the following data by comparing the values of inputs, activities, and outputs used in the project with their assumed method values:

1-Whether the assumed work products will be obtained, if so, whether they are necessary or not,

2-Before all, to query capability of each phase as a whole to reach the desired aim of the method (that is obtaining the ready software products and requirements.),

3-The applicability of the method to a project in real life,

4-Even it proves to be applicable, whether it is feasible in respect to its cost, time, and effort,

5-Necessary data for the future work.

5.3.2. Application of the CREB Method

In this section, CREB method is applied as a comparison table to the project phase by phase. The values of the proposed method are presented at the left part of the table while the obtained values of the project are at the right.

5.3.2.1. Existing System Modeling

PROPOSED	**İN THE PROJECT**
Inputs:	**Inputs:**
Mission Need Analysis (MNA),	MNA, Operational Requirements Document, and according to C4ISR guideline a Master Plan were

Operational Req. Doc. (ORD), Internal Guidelines and Regulations, Active End-User Participation	prepared by acquirer. It wasn't benefited at a satisfactory level from the internal guidelines and Regulations since there were different resources contradicting with each other. Even though end-users were provided with useful information excessively, difficulties were met in the arrangements about time and place, which are under the responsibility of PO.
Activities:	**Activities:**
BPM, Structural and Open-Ended Interviews	A sub-research group was formed with 7 members in PG for Existing System Modeling activity. Many open-ended and structural interviews have been performed together with PO, end-users, organization inspectors and instructors. The numbers of the interviews were high since the system's complexity, and the fact that the interviewers have given different answers to the same questions. AS-IS Model, which had been scheduled to be prepared in 2 months, was barely completed in 4 months with a 2-month-delay. During this period, 7 people spent an 1800-hour- effort. The prepared AS-IS model consisted of EEPC diagrams. On the diagrams workflows were indicated and the requirements for business implementations such as reports (received, sent), used devices (communications, storage, transmissions) and responsibilities (main, sub) were depicted. A similar EEPC diagram, which is used in the application, is shown in **Figure-24**. In this diagram the battlefield intelligence flows which is obtained from enemy is modeled via ARIS tool. The sub-functions of

	the flow, every sub-function' responsibilities, saving and documenting activities are shown in same diagram.
Outputs:	**Outputs:**
AS-IS Model	AS-IS Model

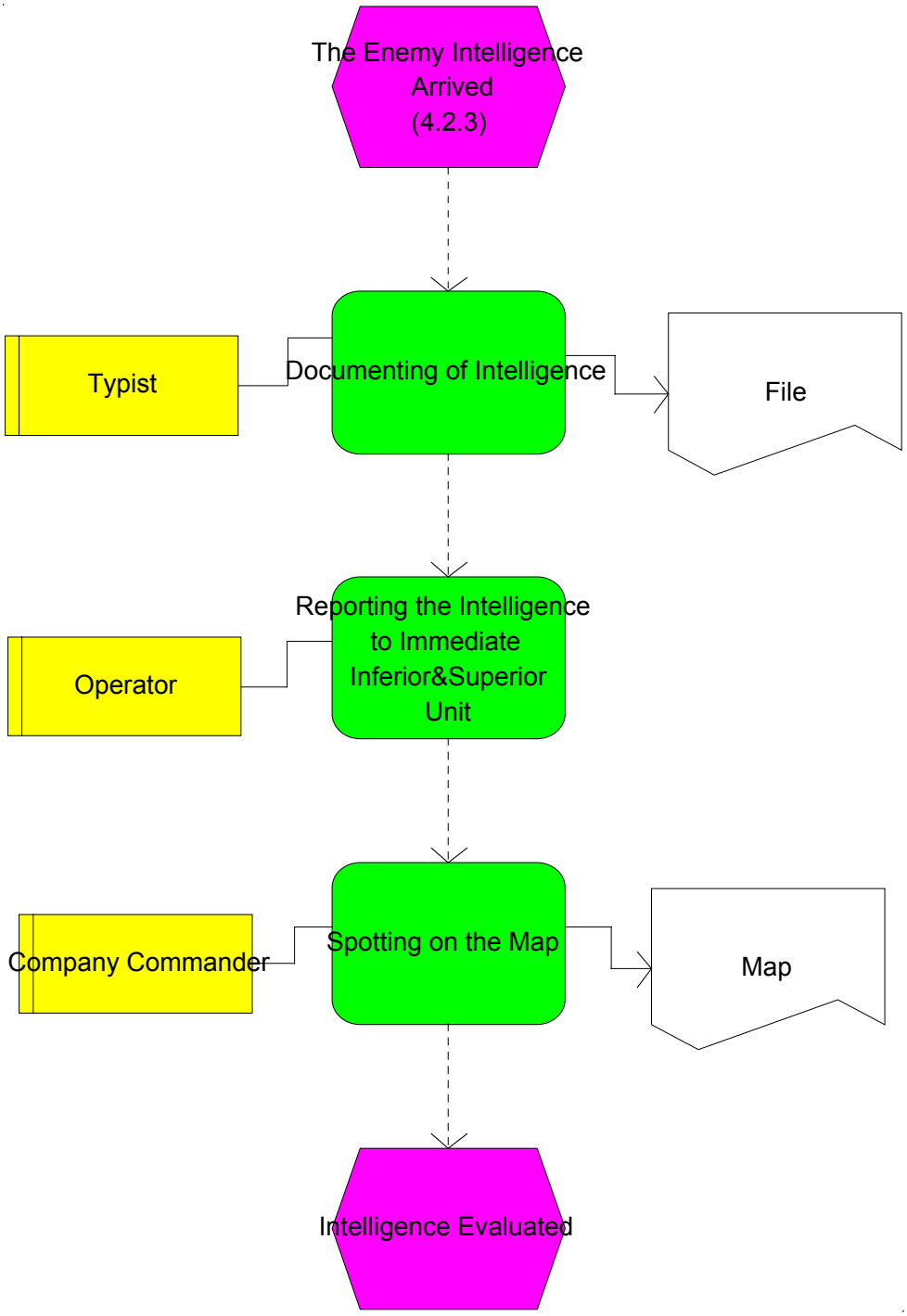


FIGURE-24 SAMPLE EEPC DIAGRAM

5.3.2.2. System Requirements Elicitation

PROPOSED	IN THE PROJECT
Inputs:	**Inputs:**
AS-IS Model, Existing Components' Specifications	AS-IS Model
Activities:	**Activities:**
Sessions, Interviews, Common Information Standards' Review Process	Functional requirements were formed after collaborative studies of PG, PO and end-users. In its study: 1) EEPC diagrams were examined, 2) The points of optimization and automation were determined, 3) The requirements of each function were documented. Non-functional requirements were not examined in this phase. For this reason, the document prepared was a TO-BE document that was including functional requirements rather than all system requirements. For the documentation activity a word processor was used. This phase was conducted by 4 personnel who worked in previous phase. A 20-day-period was needed to prepare that TO-BE document. In this phase, critical decisions about optimization and automation of the system were given. It was observed that acquirer was in contradiction with the supplied information. A sample transition from AS-IS model to TO-BE document is given in **Table-1**. In this table generated seven requirements by PG is presented. For the requirements generation activity *Acquiring the Enemy Intelligence* function's EEPC diagram, which is

	depicted in Figure-24, is used.
Outputs:	**Outputs:**
TO-BE Doc. and Sys.Req.Spec. Doc., Traceability Tables	TO-BE Doc.

Function Number: 4.2.3 (See Figure-24)	**Function Name: Acquiring the Enemy Intelligence**
4.2.3.1	The system shall give the opportunity of acquiring the enemy intelligence.
4.2.3.2	The system shall allow save, query and print functions about the enemy intelligence.
4.2.3.3	The system shall ensure that the information about the enemy is transferred to the superior as well as to inferior units regarding urgency and secrecy.
4.2.3.4	The system shall be signed the intelligence digitally which will be transmitted to superior and inferior units.
4.2.3.5	The system shall provide the sender with a message that his/her message reaches to the receiver.
4.2.3.6	The system shall indicate the intelligence gathered about the enemy on the Operational Status Map Overlay.

4.2.3.7	The system shall provide the opportunity of saving, deleting or querying the ex-intelligence about the enemy.

TABLE –1 Functional Requirements of Function 4.2.3

5.3.2.3. System Components Allocation

PROPOSED	**IN THE PROJECT**
Inputs:	**Inputs:**
Sys.Req.Spec.	TO-BE Doc.
Activities:	**Activities:**
System Components' Allocation (SCA). According to CREB, SCA could be realized with an allocation matrix like offered in **Table-2**. In this table previously determined seven requirements of Acquiring the Enemy Intelligence functions are allocated to each system components.	In the project, PG's experiences and brainstorm activities were used for components' allocation instead of any structural or proposed approaches. Every functional requirement in TO-BE document was examined. The information was gathered via some questions (see sec.4.2.3). On the other hand a sub-group was focused on the elicitation of non-functional requirements. At the same time in this phase system's general non-functional requirements (like security, capacity, speed were identified. Additionally, the software components, which met with these requirements, were identified. In this phase all components and non-functional requirements were identified by PG's experiences.

Outputs:	**Outputs:**
SCA Matrix; SW, HW Components, Data, and People	Software and Hardware components, required data and responsibilities

REQ/Sys. Components	SOFTWARE	HARDWARE	DATA	PEOPLE
R.4.2.3.1	X	X	X	
R.4.2.3.2	X		X	X
R.4.2.3.3	X		X	X
R.4.2.3.4	X		X	
R.4.2.3.5	X		X	
R.4.2.3.6	X		X	
R.4.2.3.7	X			

TABLE-2 System Components' Allocation Matrix of Function 4.2.3

5.3.2.4. Make/Buy Decision

PROPOSED	**IN THE PROJECT**
Inputs:	**Inputs:**
Software Components	Software Components
Activities:	**Activities:**
Using Decision Tree Analyze, Decision Making Techniques, and Market Research. Function 4.2.3's (Acquiring the Enemy Intelligence function)	The decision-making techniques proposed in the Method were not used to identify COTS-software products. Instead of those, at first step 17 software components were identified as COTS-software by the help of highly experienced personnel in PG. These components are such as GIS, DBMS, Operating Systems, Web Server, Anti-Virus, Security, Work Flow

software components allocation could be made as well as depicted in **Table-3**. In this table previously determined software requirements (see Table-2) are allocated to more specific software components. For example function 4.2.3's second requirements (R4.2.3.2) can be satisfied with custom development, DBMS, and O/S software components.	Management, Document Management, Transaction Management, Network Management, Desktop Management, and Help Desk&Trouble Ticketing. Among the identified software components; DBMS, Operating Systems, Anti-Virus, Firewall, GIS, and Document Management, which were currently using by the organization, and acquirer was preferred in order to use existent products as much as possible. Moreover, acquirer was demanded to develop security software from scratch due to security requirements **[Özcan, 2002].** Acquirer's considerations on COTS-software usage were decreased to the pre-determined 17 software products to 6. As a result, COTS-software products' make/buy decision issues originating from the acquirer's demands became unimportant so this phase was skipped by PG.
Outputs:	**Outputs:**
COTS-SW Comp. Refined Sys. Req. Spec.	COTS-software components

REQ/Software Components	Scratch Develop	GIS	DBMS	O/S	Transaction Mgt.	Doc. Mgt.	…..
R.4.2.3.1	X			X			
R.4.2.3.2	X		X	X			
R.4.2.3.3			X	X		X	
R.4.2.3.4	X		X	X	X		
R.4.2.3.5	X				X	X	
R.4.2.3.6		X	X	X			
R.4.2.3.7		X	X	X			

TABLE-3 Software Components' Allocation Matrix of Function 4.2.3

5.3.3.5. Cost Analysis

PROPOSED	**IN THE PROJECT**
Inputs:	**Inputs:**
WBS	WBS
Activities:	**Activities:**
FP Analysis, COCOMO, Market Research	The cost analysis of the model system was completed after the preparation of the Technical Contract by using WBS and FP analysis techniques. The cost extension was within the limits of the budget.
Outputs:	**Outputs:**
RFP Based COTS-software Req., Cost Results	Cost results

5.4. ASSESSMENT OF TECHNICAL CONTRACT

A Technical Contract is prepared for TLFC in eight months by the Project Group as mentioned in the process explained in previous sections. Requirements,

essential for COTS-software in Technical Contract are mentioned in that section as the project's main work product.

In the contract, the Supplier's responsibilities in acquisition, integration, and training in COTS-software are completely mentioned in Mandatory and Optional Requirements' section.

COTS-software product's general features are defined in Special Requirements section along with five COTS-software products requirements. General features consist of non-functional requirements. It is not a direct result of the method, but these requirements are generated after interviewing with the acquirer, and examining the available documents and regulations. COTS-software products' general features are defined as:

- All package software offered in contract shall be in their latest version at the time of submission,
- Packages' models and brands shall be mentioned,
- The requirements in the contract met by offered COTS-software products shall be expressed,
- The COTS-software products shall be able to be integrated with the system, and it shall be interoperatable with the system,
- Updated versions of the COTS-software products' shall be provided within a month without any extra cost in the guarantee and maintenance period,
- A written permission shall be provided for the COTS-software usage that is not defined in the contract to the acquirer,
- The Supplier shall provide the licenses of the packages used. These shall not be leased packages and packages shall not be beta versions.
- The COTS-software products that shall be used must have necessary references,
- The Supplier shall not demand extra charge for the license,
- The Supplier shall be responsible for any problem related to license and copyrights.

Five COTS-software products mentioned in Technical Contract are GIS (Graphical Information System), Problem Resolution, Office Programs, DBMS, and

Operating Systems. Identification technique of five components is explained in the application section. Fifty-four functional and non-functional requirements are identified for these five packages according to criteria mentioned in section three **[TLFC, 2002].** These fifty-four requirements consist of,

- Functionality,
- License issues,
- Integration issues,
- Performance specifications, and
- In-house COTS-software solutions.

5.5. APPLICATION RESULTS

Most of the benefits of CREB method were gained through applying it. The method is extended after observing project's progress. The objective of the application was to elicit COTS-software requirements for Technical Contract. Five mandatory, seven optional COTS-software components are identified and fifty-four functional and non-functional requirements are determined. As a result, the first aim of the study was achieved. But at the same time some important lessons were learned about the CREB method during the application:

- Lesson-1: *Using process-modeling tool for existing system's definition*. Process modeling provided complete understanding of the target system by the acquirer or project team who prepares contract. But for that purpose, the acquirer shall provide the group, which will model the system, with the necessary documents and information on time and accurately.
- Lesson-2: *Difficulty of non-functional requirements' elicitation*. If additional structural approaches to elicit the non-functional requirements are not applied, the requirements cannot be compared with the need.
- Lesson-3: *Time constraints need to be considered as part of CREB*. During determining the components of system and software, the experiences of PG were used instead of the structural matrix option

that the method assumed. Besides, make/buy decision was not used and cost analysis was carried out after the requirements elicitation step had been completed. It means that project groups will need more time for fully adaptation of CREB than for this application.

- Lesson-4: *Importance of Project Group's and Office's skills, experiences, domain awareness, and collaboration*. It has been proved that having experienced personnel about business process modeling and acquisition activities minimize the development time.
- Lesson-5: *Sub-research groups' need*. In addition to traditional requirements determination groups, some sub-research groups are needed to perform the necessary analysis and researches especially during the System Components Allocation and Make/Buy Decision phases of the method.

All the lessons learned in the study have brought the need of improving the method for the future work and those future work subjects are discussed in Chapter 6.

CHAPTER 6

6. CONCLUSION AND FUTURE WORK

This chapter summarizes the work documented in this thesis and presents the benefits of the approach taken in providing solutions to the COTS-software requirements elicitation problems for software intensive systems development. This chapter also describes the limitations of the CREB method developed in this thesis and proposes future work that is needed to improve it.

COTS-software products are gaining more importance as time passes in software sector and software-intensive systems include a high ratio of COTS-software components. However, as stated in previous chapters, few studies are conducted for COTS-software products' requirements elicitation.

The developed CREB method is examined within the scope of the activities between System Definition phase and RFP preparation. The method is performed by the acquirer in five phases. In the first phase, Existing System Definition, the current structure of the system that will be automated is described by using a business process modeling tool. In the second phase, System Requirements Elicitation, target system requirements are elicited by determining the optimization and automation points on the current system structure modeled. In the third phase, System Components Allocation, the requirements obtained are classified as software, hardware, data and people. Then, the method focuses on acquisition of COTS-software requirements. In the fourth phase, Make/Buy Decision phase, software components to be obtained are determined as scratch-develop or as commercial solution. The last (fifth) phase, Cost Analysis, carries out the cost analysis of the entire system from the standpoint of finance and refines the requirements according to the obtained results.

The proposed method had an opportunity to be applied in an intelligence project that belongs to TLFC and applicability of the method was verified by this way. Five mandatory and seven optional COTS-software products and fifty-four functional and non-functional requirements are identified in the project. The experts in TLFC stated that this identification caused the COTS-software requirements to have been explained in a clear way.

Results of this study can be summarized as:

-The study deals with general lack of requirements engineering research for packaged-based systems development.

-A method, CREB, is proposed for process model based requirements elicitation of COTS-software components.

-In the application based on proposed method, importance of process modeling tools for acquisition of requirements is understood. Furthermore, it is understood that using of process-modeling tools for system definition applies at satisfactory level for functional requirements elicitation. Additional techniques are necessary for non-functional requirements' elicitation.

-In the study, acquirer's responsibilities for COTS-software components usage in software intensive projects are identified. Acquirer can:

1. Determine COTS-software components. (This decision can include specifying brands and versions.)
2. Assert the specifications of components.
3. Track conditions for components' availability status. (If proper COTS-software components are available or are not available, acquirer can refine the requirements.)

-Experiences shouldn't prevent the observation and research of newly developed or upgraded components.

-Lastly, mature system definition eliminates uncertainty about usage of COTS-software components. Especially in safety critical projects, eliminating uncertainty increases usage ratio of components.

6.1. FUTURE WORK

The method reported in this thesis aims to address problems in requirements elicitation for package-based solutions. Although the thesis contributed useful research in requirements engineering for software intensive system's procurement, the CREB method has some weaknesses and limitations:

- The CREB doesn't address non-functional requirements. Method for non-functional requirements elicitation mainly proposes interviewing stakeholders at System Requirements Elicitation phase. However, during the application, it is understood that such a study is not sufficient. Therefore new techniques are needed for non-functional requirements' elicitation. For non-functional requirements elicitation, STACE method explained in chapter 3 can be merged with the CREB.
- Further studies in different organizations should be performed to enhance the method.
- The CREB method needs to address more specific make/buy decisions. With the current situation, method doesn't force any technique. Some make/buy decision techniques shall be used in the method, according to the projects' complexities.
- Other capabilities of business process modeling tools shouldn't be ignored. For the functional requirements' generation activity, business process modeling tools can be used for semi-automated requirements generation **[Yıldız, 2002].** This increases the traceability of the requirements to the AS-IS model. At the same time process-modeling tools allow different representations. With the different representations, method might be extended. For example a modeling technique of ARIS, Function Allocation Diagram (FAD) allows the transformation of input data into output data. At the level of implementation description, the design specification can be transferred to concrete hardware and software components, through the usage of FAD modeling technique and EEPC modeling technique together.

REFERENCES:

[Bass, 1998] Bass Len, Clements Paul, Kazman Rick, " Software Architecture in Practice", Addison Wesley, ISBN 0-201-19930-0, pp.84.

[Bilgen, 1992] Bilgen, Semih, "Niçin Yazılım?", Türkiye Bilişim Derneği Yayınları, pp.116.

[Boehm, 1981] Boehm, B.W., "Software Engineering Economics", Prentice-Hall, Englewood Cliffs.

[Boehm, 1989] Boehm, B.W., "Risk Management", IEEE Computer Society Press.

[Bredemeyer, 2000] Bredemeyer Dana, Malan Ruth, "Defining Non-Functional Requirements", COTS Workshop: Continuing Collaborations for Successful COTS Development, Limerick, Ireland, pp. 3.

[Breslin, 1986] Breslin, Jud, "Selecting and Installing Software Packages", Quorum Books, ISBN 0-89530-158-4.

[Brooks, 1987] Brooks, Frederic, " No Silver Bullet: Essence and Accidents of Software Engineering", IEEE Computer, pp.10-19.

[Conger, 1994] Conger, Sue, "The New Software Engineering", Wadsworth Publishing, ISBN 0-534-17143-5, pp.666.

[Dorsey, 2000] Dorsey Paul, Kolletzke Peter, "ORACLE Designer/2000 Handbook", OSBORNE McGraw-Hill, ISBN 0-07-882229-7, pp 83.

[Özcan, 2002] Özcan, Fuzuli, "An Approach for Defensive Information Warfare in TLFC", M.S. Thesis , METU, 2002.

[Garlan, 1995] Garlan D., Allen R., Ockerbloom X., "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.

[ISO, 1995] ISO/IEC 12207, "Software Life Cycle Process".

[ISO, 2001 ISO/IEC CD 15288 CD3, "System Engineering-System Life Cycle Processes".

[IEEE, 1998] IEEE Std 830-1998, " Standard for Software Requirements Specification", pp.11.

[Jackson, 2000] Jackson H.John, Mathis L. Robert, "Human Resource Management", South-Western College Publishing, ISBN 0-538-89004-5,pp.295-297.

[McClure, 1989] McClure, Carma, "CASE IS Software Automation", Prentice Hall, Englewood Cliffs, ISBN 0-13-119330-9.

[Ncube, 1998] Ncube Cornelius Maiden A Neil, "Acquiring COTS Software Selection Requirements", IEEE Software, pp.46-56.

[Oberndorf, 2000] Oberndorf Patricia, Carney David, " Developing New Processes for COTS-Based Systems", IEEE Software, pp.48-55.

[Pressman, 1997] Pressman,Roger S., "Software Engineering :A Practitioner's Approach", Fourth Edition,,McGraw-Hill, pp.261-262.

[Ross, 1985] Ross T. Douglas, "Applications and Extensions of SADT", IEEE Computer, pp.25-33.

[Scheer, 1994] Scheer, W., A., "Business Process Engineering: Reference Models for Industrial Enterprises", Springer-Verlag, ISBN 3-540-58234-7.

[Scheer, 1998] Scheer, W., A., "ARIS Methods: Version 4.0".

[Scheer, 1998] Scheer, W., A., "ARIS-Business Process Frameworks", Springer-Verlag, ISBN 3-540-64439-7.

[Sommerville, 1995] Sommerville, Ian, "Software Engineering", Addison-Wesley, Fifth Edition, ISBN 0-201-42765-6.

[Szyperski, 1998] Szyperski, C., "Component Software: Beyond Object-Oriented Programming", Addison Wesley.

[TLFC, 2002] "A Technical Contract of Intelligence Department of TLFC", 31072001 v.19.

[Voas, 1998] Voas, Jefffrey, "Maintaining Component-Based Systems," IEEE Software, pp.22-27.

[Weyuker, 1998] Weyuker,j.,Elaine, " Testing Component-Based Software: A Cautionary Tale", IEEE Software, pp 54-61.

[Wiegers, 1999] Wiegers E. Karl, "Software Requirements", Microsoft Press, ISBN 0-7356-0631-5, pp.201-202.

[Yeh, 1997] Yeh T. Raymond, Ng A. Peter, "Software Requirements-A management Perspective", IEEE Computer Society Press, pp.380.

[Yin, 1994] Yin K. Robert, "Case Study Research Design and Methods", Sage Publications, ISBN 0-8039-5663-0, pp.84-86.

[Yıldız, 2002] Yıldız, Okan, "An Approach For Eliciting Functional Requirements Of The Software Intensive Systems Based On Business Process Modeling", M.S. Thesis, METU 2002.