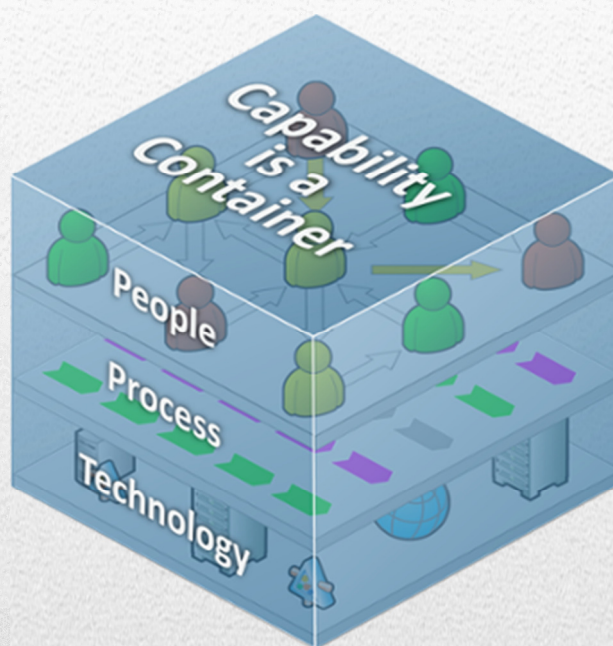




HAROKOPEIO UNIVERSITY OF ATHENS
DEPARTMENT OF INFORMATICS & TELEMATICS
MSc in Advanced Information Systems in Business



Master Thesis: Building a Database and Interface for analyzing systems and processes, by using specific data in a case study from the maritime domain field

Author: Vasiliki Gantzoudi

Supervisor: Professor Pericles Loucopoulos

Athens 2016

TABLE OF CONTENTS

CHAPTER 1: Introduction.....	5
1.1 Dissertation goals	6
1.2 Description of the Problem	8
1.3 Approach	9
1.4 Describing the Maritime Application	11
1.5 Dissertation Structure	14
1.6 Chapter Summary.....	15
CHAPTER 2: State of Art	17
2.1 Capability Related Work.....	18
2.1.1 The beginning of Capabilities	18
2.1.2 Business Capability	22
2.1.3 Capability as Service	31
2.2 Conceptual Modeling	37
2.2.1 Database Design	37
2.2.2 Data Modeling Approaches & Techniques	38
2.3 Discussion – Result of Research	51
CHAPTER 3: Object Role Modeling (ORM)	55
3.1 Background to Meta – Modeling.....	56
3.2 The Conceptual Schema Design Procedure.....	58
3.3 The Relational Mapping Procedure.....	61
3.3.1 Definitions – Notation	61
3.3.2 Rules & Strategies of Mapping	63
3.3.3 Main Steps of Mapping	72
3.4 Chapter Summary.....	74
CHAPTER 4: The Capability Meta – Model	76
4.1 An Initial Version of the Business Capability Meta – Model	77
4.2 Need for Change towards a New Business Capability Meta – Model.....	85
4.3 Chapter Summary.....	99
CHAPTER 5: Mapping the Capability Meta – Model to Relational Schema	101
5.1 Followed Procedure of Mapping in Detail.....	102
5.2 Chapter Summary.....	117
CHAPTER 6: Physical Database	118
6.1 DBMS Architecture	119

6.2	“BC” Physical Tables	121
6.3	Database Testing or Back-End Testing	137
6.4	Chapter Summary.....	161
CHAPTER 7: User Interface Design & Implementation.....		163
7.1	User Interface Design Process & Quality Characteristics	164
7.2	Use Case Diagram as a Description of the Main Windows	166
7.3	Hierarchy of Forms	168
7.4	Basic Flow Chart for Data Entry	170
7.5	Application Screens	175
7.6	Chapter Summary.....	196
CHAPTER 8: Case Study from the Maritime Field.....		198
8.1	DMC Data Description	199
8.2	Inform the Application about a New Business Capability	211
8.3	Executing Queries.....	234
8.4	Removing Current Records in Specific Forms.....	240
8.5	Chapter Summary.....	243
CHAPTER 9: Conclusions.....		244
CHAPTER 10: Appendix.....		249
10.1	Table 1: Literature review for Business Capabilities	250
10.2	Table 2: ORM 2 Graphical Notation.....	256
10.3	SQL Script of BC Tables	262
10.4	SQL Script of Total View	269
CHAPTER 11: Bibliography.....		273

Table of Figures

Figure 1: Database modeling in context (Halpin, Evans, Hallock, & Maclean, 2003).....	10
Figure 2: The four dimensions of core capabilities (Barton, 1992)	19
Figure 3: Definition of core capabilities (Long & Vickers-Koch, 1995)	20
Figure 4: Capability – driven strategy (Tallman & Fladmoe-Lindquist, 2002)	21
Figure 5: Business capability model taxonomy (Holman, 2006)	22
Figure 6: Level 1 Foundation Capability Model – Operational and Environmental Capabilities (Holman, 2006).....	23
Figure 7: The conceptual framework for modeling Business Capabilities (Brits, Botha, & Herselman, 2007)	24
Figure 8: Capability Construction Feedback Loop (Brits, Botha, & Herselman, 2007)	24
Figure 9: Business Capability Components (Greski, 2009-B).....	25
Figure 10: Value Streams leads to identification of Business Capabilities (Rosen, 2010).....	26
Figure 11: Business Capabilities Centric Extension (BCCE): Changes to Meta-model (Barroero, Motta, & Pign, 2010)	27
Figure 12: Capability Dependency Analysis Method (Freitag, Matthes, Schu, & Nowobilska, 2011).....	27
Figure 13: Capability relation of business aspects (Bakhtiyari & Adel, 2012)	28
Figure 14: Example Level 1 Capability map (Ulrich & Rosen, 2011).....	28
Figure 15: The Role of Business Capabilities in EA (Ulrich & Rosen, 2011)	29
Figure 16: Top – down approach for Capability – Based Service Identification (Frey, Hentrich, & Zdun, 2013)	30
Figure 17: The capability meta-model (Stirna, Grabis, Henkel, & Zdravkovic, 2012).....	32
Figure 18: CaaS upon the existing information technologies services (FP7 Collaborative Project with No 611351 , 2014).....	33
Figure 19: Meta – model for capability design and deployment in cloud (Zdravkovic, Stirna, & Henkel, 2013)	34
Figure 20: Capability driven development environment (Zdravkovic, Stirna, & Henkel, 2013)	34
Figure 21: Goal Model of the project (Espana, Gonzalez, Grabis, Jokste, Juanes, & Valverde, 2014).....	35
Figure 22: CDD Methodology (Berzisa, et al., 2015)	36
Figure 23: Data Modeling Techniques, Methods & Languages.....	39
Figure 24: An example of Bachman Diagrams (Bachman, 1969)	40

Figure 25: An example of ER Model (Ponniah, 2007).....	40
Figure 26: IDEF Methods (Mayer, Painter, & deWitte, 1992)	41
Figure 27: IDEF1X Origins (Mayer, Painter, & deWitte, 1992)	42
Figure 28: IDEF1X Notation (Ponniah, 2007).....	42
Figure 29: IDEF1X – An example model (Ponniah, 2007)	43
Figure 30: Information Engineering Notation (Wambler, 2015).....	44
Figure 31: Information Engineering – An example model (Ponniah, 2007).....	44
Figure 32: Barker’s Notation (Wambler, 2015)	45
Figure 33: Barker’s Notation – An example model (Ponniah, 2007).....	45
Figure 34: ORM – An example model (Ponniah, 2007)	47
Figure 35: Graphic Symbols of FCO-IM (Bakema, Zwart, & Lek, 2002)	48
Figure 36: An example of FCO-IM (Bakema, Zwart, & Lek, 2002).....	48
Figure 37: UML Class Diagram (Ponniah, 2007)	50
Figure 38: The four levels hierarchy of a model (Hinkelmann, 2015)	57
Figure 39: Examples of relational schema in (a) horizontal layout and (b) vertical layout (Halpin & Morgan, 2008).....	62
Figure 40: Business Capability Meta-Model (Loucopoulos, Bravos, Stratigaki, & Vavlis, 2013)	77
Figure 41: A new Conceptual Model for Business Capability.....	98
Figure 42: Most popular DBMS according to DB-Engines ranking (www.db-engines.com) .	119
Figure 43: Popularity trend in DBMS (www.db-engines.com)	120
Figure 44: Database Testing Process (www.softwaretestinghelp.com).....	137
Figure 45: The elements of User Interface Design (Mandel, 2002)	164
Figure 46: User Interface Design Principles (Sommerville, 2007)	164
Figure 47: Software Quality Characteristics (Bevan, 1999).....	165
Figure 48: Quality Criteria for User/System Interface (Oren & Çetin, 1999)	165

CHAPTER 1: Introduction

Structure of this Chapter

- 1.1 Dissertation Goals
- 1.2 Description of the Problem
- 1.3 Approach
- 1.4 Dissertation Structure
- 1.5 Describing the Maritime Application
- 1.6 Chapter Summary

This chapter is an introduction into the scope of this dissertation. In Section 1.1 we present the specific goal of this dissertation which is the creation of a maritime application that refers to a DBMS for Business Capability, and also some of sub goals that deals with how we engaged with this application. In Section 1.2 a brief description of the problem is presented and in Section 1.3 the approach that will be used in order to achieve the main goal. Then in Section 1.4 a description of the application is presented in relation with a real case company from the maritime domain field, the Danaos Management Consultant. Finally in Section 1.5 a structure for the rest of this dissertation is presented and in Section 1.6 a brief summary of this Chapter.

1.1 Dissertation goals

In modern times maritime organizations operate in dynamic business environments where competition is order of the day. Also the international and global environment consists of factors that have a significant impact on their operation (e.g. global or local laws, global strategies etc).

The main objectives of those organizations are to achieve growth, to be success, to survive in these conditions, and thereafter to be leaders in the maritime marketplace. In order to fulfill the previous they must be able to gain competitive advance and to provide a business value. Also it is imperative for those organizations to increase the demand of their provided services, to increase their profits, to hold a significant market share and to have a good reputation.

On the other hand organizations themselves are very complex systems with a large number of business process, followed by rules, goals, a changing context, etc. Because of this complexity, managing those organizations is fundamentally difficult than it was in the past. A solution to that was given by identifying what an organization actually does and aligning this with the Information Technology (IT), who is the application of computers and telecommunication equipment to store, retrieve, transmit and manipulate data often in the context of a business or other enterprise ([Wikipedia: The Free Encyclopedia](#)).

Recently a Business Capability definition has being used for describing what a Business does ([Holman, 2006](#)) and this definition become the Rosetta Stone for the communication between two separate Worlds those of Business and IT ([Ulrich & Rosen, 2011](#)). More lately Business Capability became the centric idea for the development of the digital enterprises of tomorrow ([European Commision, 2013](#)).

Thus in order to be comprehensive for a reader the notion of Business Capability, ***one first subject of concern in this dissertation is to answer the questions of:***

- *What is the Capability of an organization in general;*
- *Why is important for an organization to focus in Business Capability;*
- *Why Business Capability must be used for the development of software in digital enterprises of tomorrow;*
- *How Capability must be used for the development of software for digital enterprises of tomorrow;*

Also databases and database management systems have become an emergency component for the operation of modern organizations. More especially organizations success depends on its ability to acquire accurate and timely data about its operation, to manage this data effectively and to use them to analyze and guide its activities (Rapakrishnan & Gehrke, 2003). Those systems are putted under the framework of Information Technology (IT) and are an important part of software engineering.

Thus the main goal of this dissertation is to provide in practice a real case of alignment between Business and IT, by creating a specific maritime application that will combine the previous two. This maritime application will be ***a Maritime Database Management System for Business Capability***, meaning a software system for maintaining the information about Business Capability and answering queries about that.

However in the framework of IT and more specific in software engineering an important area is Information Modeling, who concerned with the constructions of computer – based symbols structures, which capture the meaning of information and organize it in a ways that make it understandable and useful to people (Mylopoulos, 1998). In the area of database design Information Modeling refered as Data Modeling and provides the necessary methodologies, techniques and languages for supply data modelers in their work. So one ***another concern of this dissertation is the presentation of the data modeling methodologies, techniques and language for designing a database, and the proposition of the most suitable for Business Capability Maritime Database Management System.***

After choosing the most suitable methology or technique or lanquage, then the ***sub goals*** of this dissertation will be:

- To follow the specific modeling procedure for designing this applicaton, that is indicated according to the choosen data modeling methology or technique or lanquage.
- To choose a specific architecture for our DBMS and the most suitable supporting program systems that can be used for creating this application.
- To carry out the necessary checks in the database of our application.
- To create a User Interface according to a specific quality criteria and to describe it in a way that will be undersantable even for nont-technical persons by using a Use Cases Diagrams, a hierachy diagram and a flow chart for data entry.
- Finally to provide a real Use Case of data for a specific maritime company, the Danaos Management Consultant, in order to manage data (actions fo insert, delete and update) and to export queries about that in a real enviroment.

1.2 Description of the Problem

Nowadays modern organizations are facing a dynamic business environment which characterized by change. Crucial role in the adaption of the organization in these conditions is the development of new solutions that will fill the gap between the business and IT alignment and will make more predicable the context of use and the circumstances in which an Information System operates.

In those circumstances in order organizations to achieve a competitive advance and provide business value they must focus on what they really do. Business Capability describes the following and according to that recently a new approach has been proposed, the Capability Driven Development (CDD). This approach is going to be the foundation for the development of software for the digital enterprises of tomorrow and has as centric idea the Business Capability modeling. Although the essential tools and methodologies have being given for the CDD approach, lacks from empirical experience of application.

Thereafter we have taken a case study from the domain field of a maritime company and the purpose of this dissertation is the development of a Database Management System (DBMS) for Business Capability. By creating this Database Management System we intend to help modern organizations to gain a competitive advance and therefor to achieve growth. That's because when the information about Business Capability are stored, related and viewed in a database, managers have an overall view of what the organization does and thereafter can increase control, achieve better planning and taking decisions more efficient. Finally in order to capture the business complexity and to remain easy to change the modeling language and querying in DBMD must be clear and efficient communicating. Thus we have chosen the ORM data modeling technique for the development of the DBMS.

Keywords: Business Capability, Data Modeling, Capability Driven Development, Object – Role Modeling (ORM), Conceptual modeling.

1.3 Approach

Describing a Business Capability in a Database is something new. Thus the research approach taken in this dissertation is firstly to create a collection of papers and articles from the bibliography in order to achieve the goals deals with understanding of what the Capability of an organization is in general, why is important for an organization to focus in Business Capability, why Business Capability must be used for the development of software in digital enterprises of tomorrow and how.

According to [Merson \(2009\)](#) data modeling is a common activity in the software development process of information systems, which usually use a Database Management System (DBMS) to store information. Also a data model is commonly created to describe the structure of the data handle in information systems and persisted in DBMS. Thus the research approach of this dissertation is secondly to review in bibliography in order to present the data modeling techniques that exists to describe the structures of data.

The implementation approach for describing the structures of the DBMS, that has being chosen in this dissertation is **a fact-oriented data modeling technique the ORM** (Object Role Modeling). This data modeling technique began in the early 1970s as a semantic modeling approach which views the word as objects playing roles ([Halpin T. , 1995-A](#)). As discussed by [Halpin, Evans, Hallock, & Maclean \(2003\)](#), this technique deals with four levels for working with data: **external, conceptual, logical and physical** ([Figure 1](#)); that consist of different database modeling tasks. As they discussed in the Business Analysis task we create a semantically accurate conceptual model of an application domain in terms easily understood. In the Logical data design we create a normalized data model that accurately represents the conceptual model with tables and columns uniquely named, key (primary, foreign) relationships, constraints and derivation rules. In the physical database design we create a SQL schema for a specific database management system, including physical data types and indexes. In the performance database design we tune a physical model for optimum performance on the specific software and hardware platform.

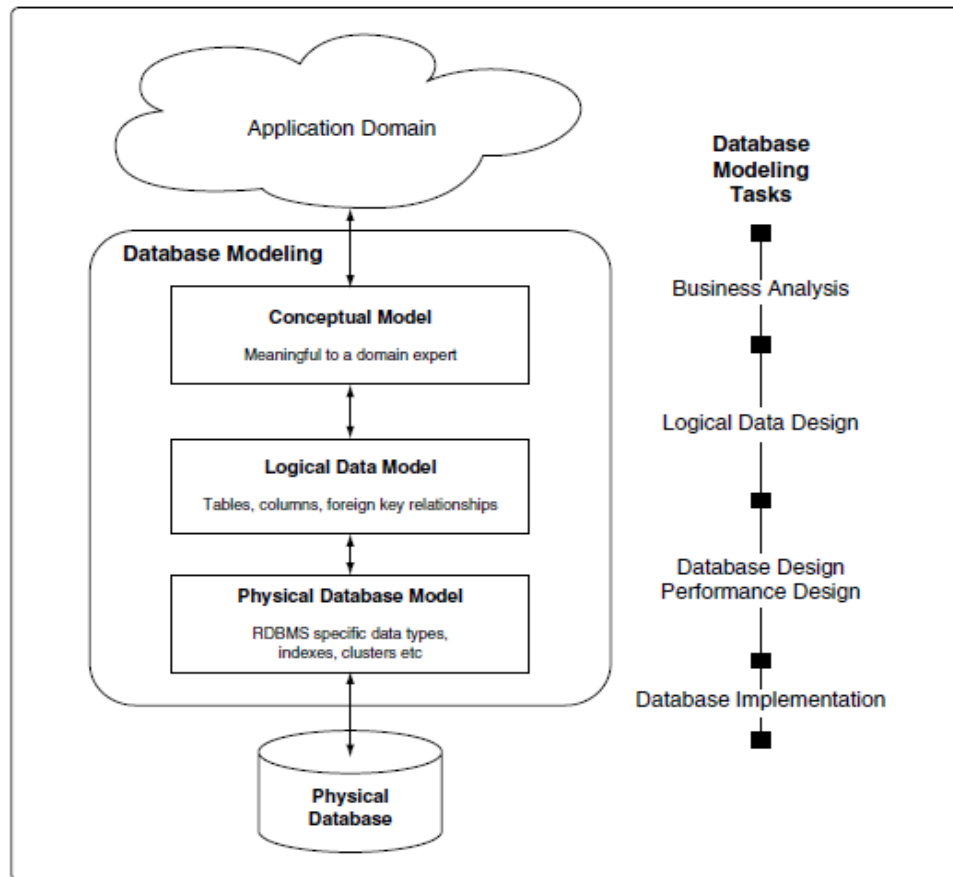


Figure 1: Database modeling in context (Halpin, Evans, Hallock, & Maclean, 2003)

For the purpose of this dissertation we have taken a conceptual model of Business Capability definition by previous work of Loucopoulos et al (2013) and we have adapted to a new conceptual model that can be used for describing in detail the ontologies (e.g. object types, relationships etc.) of the database. Next we create the logical schema (logical data model) for our DBMS by using Halpin (1995-A) mapping procedure that extents and refines an older mapping procedure known as the ONF (Optimal Normal Form) algorithm. Then the logical schema is used with a specific relational database management system (RDBMS), the Oracle DBMS, which in that period is the most popular commercial RDBMS. For this system we use the Oracle Database 11 g as the physical database (as a server) and the Oracle PL/SQL (as a client) an IDE (Integrated Development Environment) in order to store, relate (queried) and determine which kind of data will be accessible to which user groups in the database. Finally we create an external design, which is implemented with the Oracle Forms 6i, and involves the designing appropriate interface for the users.

1.4 Describing the Maritime Application

As we have already stated in a previous section, the result of this dissertation will be the implementation of a maritime application for Business Capability. This application will be used to describe all the related information about Business Capability in a case study of a company from the maritime domain field, the Danaos Management Consultant, who is a part of Danaos Corporation.

Danaos Maritime Consultant (DMC) specializes in software services in the maritime industry and is a leader in this field over the past 30 years. This company in order to be able to provide his services to other shipping companies has developed specific Capabilities which are owned by her (internal capabilities) and related with:

1. The provision of a specific integrated software solution, the Danaos Enterprise Maritime Solution (DEMS), which contains different platforms, that automates all the types of operations or functions of a shipping company (e.g. financial operations, HR management operations etc).
2. The provision of the previous software solution in a web-enable version, maintaining at the same time an ERP, in which the company is able to retain control and security on transactions and communications, and also to advertise its clientele and their actions. This version also provides social-networking services such as web-conferences and a forum.
3. The provision of information and storage management in their client's data, by offering a high level of security and privacy in the previous web-enable version of software and also by keeping a single unified database.
4. The provision of an application, the Port of Calls, in which the essential conforms to all required regulations and rules in each port are achieved.

Also this company has developed some other Capabilities that are owned by some other companies (external capability) and related with:

1. The provision of web-conference management in the web-enable version that offers, which owned by Microsorg Lync.
2. The provision of technical assistance in the web-conference services that provides, which owned by ComSys.
3. The provision of long-term business process outsourcing solutions to owners and managers, who are searching for cost-effective methods for updating and maintaining back-end software, which offered in India by Danaos Services Company.

These Capabilities are interrelated with some other elements of the company, which concerns the high level Strategic and low level Operational goals that must be achieved for each of them, the context in which they exist, the kind of collaborations that exist between them, the ability they use that is made of specific skills, the capacity they use that is made of a specific set of resources and the Business Process they followed for these services, which leads to specific tasks.

From the above we can understand that the related information about Business Capabilities of this company is complex enough in order to facilitate the managers of the company to be able to handle this kind of information in a proper way. On the other hand managers of this company consider that it is important to be able informed at any time about what Capabilities exist in the environment of this company and which elements are interrelated with those. That's because by this way they are able to know at any time what the company actually does, and thereafter they are able to analyze their weakness and strengths related to the operational activity of the company.

The previous concerns of managers can be achieved by implementing the Maritime Application of DBMS for Business Capability, since this application provides different functions according to the previous described. Thus by using this Maritime Application we are able to:

- A. Manage all the related information about Business Capabilities: In this function different sub-functions may exist. In more detail a user of this application is able to:
 1. Create a New Capability. This sub-function concerns the creation of a New Business Capability either internal or external, and also to management of the hierarchies of them, meaning the management of relationship between the main and sub capabilities.
 2. View Total Capability. This concerns the ability of having the total picture for those Business Capabilities or the information about a specific Business Capability. In more detail a user is able to see a general information about those capabilities (e.g code, description, type etc), the kind of ownership, the kind of hierarchies about them, the outputs that is delivered by them, the goals that achieved by them, the context in which they exist, the collaborations between them, the ability they use for delivering a service, the capacity they use for delivering a service, the ability in relation with capacity and service they use, in order to be able to operate.

3. Manage Owners: This concern managing the information about the owners that exist in the company.
 4. Manage Context: This concerns the management of all the available information about the relation between the Business Capabilities and the context in which they exists, and also the management separately of all the available information about context.
 5. Manage Outputs: This concerns the management of all the available information about the outputs the company delivers to other companies, the relation of this information with a specific Business Capability and finally the management of the hierarchies about those outputs.
 6. Manage Collaborations Between Capabilities: This concerns the management of all the kind of collaborations that exists between capabilities and also the management of the kind of collaborator connector that is used in orders these collaborations to take place.
 7. Manage Goals: This concerns the management of all the available information about the relation between Business Capabilities and goals, also the management of the information about goals and the management of the hierarchies about them.
- B. Manage all the related information about Ability: This concerns the management of the information about either Internal Ability or External Ability that the company uses for creating a service and is defined by a specific skills.
- C. Manage all the related information about Capacity: This concerns the management of the information about either Internal Capacity or External Capacity that the company uses for creating a service and is defined by a specific resources.
- D. Manage all the related information about Services: This concerns the management of the information about the services that the company delivers and also the management of the information about the business process follows for those services. Thus this function is divided into two different sub-functions, who include:
1. Manage Services: Here a user may insert all the available information about the services and also relate this service with a specific Business Process.
 2. Manage Business Process: This concerns the management of the information about Business Process.

1.5 Dissertation Structure

We have already discussed the dissertation goals, the description of the problem and the approach that has been used. The rest of this dissertation is structured as follows.

In Chapter 2 first section discusses the related work in bibliography about Capabilities, starting with the beginning of Capabilities in Strategic Management theories and then the adoption of the concept of Business Capability in the Business Informatics and in more especially in Enterprise Architecture, in Service – Orientation, in Business – IT alignment and in software development. Finally it discusses the Capability Driven Development as the foundation of the Capability as Service project. The second section of this chapter deals with Conceptual Modeling and more especially discuss the Database Design Process, the Data Modeling approaches and techniques, and concluding by proposing the Object Role Modeling as the most suitable method for creating the Database Management System for Business Capability.

In Chapter 3 a previous work of an ORM model for Business Capability definition is issued as a background to meta – modeling, and then the Relational Mapping Procedure according to ORM are presented. This procedure will be the guide for designing the logical data model for our Database Management System. More especially for this procedure are discussed the main definition and notation, the rules and strategies of mapping, and the main steps of mapping.

In Chapter 4 a new version of Capability Conceptual Model is presented, which is an extension of a previous definition Business Capability model. In this model we have added the essential reference modes and value types that are needed for describing object types, also the necessary subtype constraints for describing subtypes and in some cases some value constraints and some ring constraints.

In Chapter 5 we describe in detail the relational mapping procedure has followed, in order to design the relational schema (logical schema) for our Database Management System, according to the conceptual meta-model. Finally this Relational Schema is given.

Chapter 6 deals with physical database. In more detail the DBMS architecture is discussed and also the reasons for adopting this kind of system. Then a view of physical tables are presented, including a description of the implementation of constraints where needed. Also a created View is given and the sequences, by which we have generated for some tables unique primary keys. Finally a Database Testing is discussed for a specific Functional Group of tables.

Chapter 7 deals with User Interface design and implementation. The user interface Design Process and quality characteristics are presented. Then a Use Case Diagram, a description of the main windows of the Application, a hierarchy of forms, a basic flow chart for data entry and the application screens are given.

Chapter 8 deals with a Case Study from the Maritime domain field, the Danaos Management Consultant. In more detail firstly a data description of the Case Study is presented. Then we are discussing the way in cases of inserting data, in cases of querying and finally in cases of removing specific records in our Application.

Finally in the end of this dissertation the conclusions of the total work are given.

1.6 Chapter Summary

In this Chapter we have concerned with the introductory concepts of this dissertation. Thus we have presented the basic goals of this dissertation, a description of the problem, the followed approach, the description of the application and finally how this dissertation is organized for the rest of the Chapters.

In more detail by identifying the Business Capability of an organization, and aligning this with the Information Technology (IT), is the key for success and growth in modern organizations, since by this way those organizations are able to manage their complexity (Holman, 2006; Ulrich & Rosen, 2011). During the goal specification of this dissertation one first concern was presented, and related with answering the questions of: What is the Capability of an organization in general; Why is important for an organization to focus in Business Capability; Why Business Capability must be used for the development of software in digital enterprises of tomorrow; How Capability must be used for the development of software for digital enterprises of tomorrow. Then the specific goal of this dissertation was presented, which referred in providing in practice a real case of alignment between Business and IT, by creating a Maritime Database Management System for Business Capability. This DBMS will be used for managing the information about Business Capability and answering queries about that. Since the designing process of this application requires the knowledge of Data Modeling principles, another concern of this dissertation was stated as the presentation of the data modeling methodologies, techniques and language, that exists, and the proposition of the most suitable for Business Capability Maritime Database Management System. Finally some sub goals were presented and related with how we engaged with the implementation of the application.

Thereafter a description of a problem is presented, by which the creation of the DBMS will be a solution not only in order to facilitate the work of managers of an organization, but also by providing a real case of example for feeding up the Capability Driven Development, which has being used for the development of software for the digital enterprises of tomorrow.

Then a researching and also an implementing approaches was presented. In the researching approach one firstly concern was the collection of papers and articles from the bibliography, in order to be able answer the previous queries about capabilities and to present the data modeling methods that exists. For the implementations approach was choosen that of (Halpin, Evans, Hallock, & Maclean, 2003), in which the under development system is examined under four levels of analysis: external, conceptual, logical and physical. Finally the software components of this DBMS was presented and referred in the Oracle 11G for database, in the Oracle PL/SQL for IDE and in the Oracle Forms & Reports 6i for the interface development.

Continuing a brief description of the maritime was presented. In this part the main functionality of the application was presented in relation with a case study from the maritime domain field, the Danaos Management Consultant Company.

Finally the structure for the rest of the Chapters was presented in order the reader to facilitate with the context of this dissertation

CHAPTER 2: State of Art

Structure of this Chapter

2.1 Capability Related Work

2.1.1 The beginning of Capabilities

2.1.2 Business Capability

2.1.3 Capability as Service

2.2 Conceptual Modeling

2.2.1 Database Design

2.2.2 Data Modeling Approaches & Techniques

2.3 Discussion – Result of Research

This chapter briefly reviews the state of art of this dissertation. Thus Section 2.1 deals by presenting the Capability Related Work and it began by presenting the different research areas that Capabilities has being used during the time. Then the Section 2.1.1 presents how Capabilities began from the Management Theories and the Section 2.1.2 how become an important research concern with the notion Business Capabilities in the area of Business Informatics and more especially in Business and IT alignment, in the area of Enterprise Architecture, in Service Orientation and in transformations of software systems. In the section 2.2.3 Business Capabilities has become a centric idea for creating software for the digital Enterprises of tomorrow, by being a part of the CDD method. The second part of this chapter, Section 2.1, deals with Conceptual Modeling. In more detail in Section 2.2.1 the database design procedure is presented and in Section 2.2.2 a brief review of the most known Data Modeling Approaches, Techniques & Languages. Finally in Section 2.3 a brief summary, a discussion and the result of research are presented.

2.1 Capability Related Work

The notion capability has been used in different research areas during the time. In bibliography we meet the term Capabilities in the Management Science (Barton, 1992; Stalk et al, 1992; Long & Vickers-Koch, 1995; Teece et al, 1997; Tallman & Fladmoe-Lindquist, 2002), in Social Sciences and more especially in the field of Sociology (Nussbaum, 2000), in the field of Psychology (Anand, Hunter, & Smith, 2005), in the field of Political (Deneulin & McGregor, 2010) and in the field of Economics (Duhs, 2008). Also it has been used in the Engineering Science and more especially in the interdisciplinary field of the Systems Engineering (Cusick, 1997). Finally we meet the term capability in Computer Science and more especially in the field of Artificial Intelligence (Zhang, Sreedharan, & Kambhampati, 2015), in Software Development (Frey, Hentrich, & Zdun, 2013) and in the field of Business Informatics (Zdravkovic, Pastor, & Loucopoulos, 2014).

For the purpose of this dissertation we will see how the term capability began as a core component in the Strategic Management, then became an important research concern in Business Informatics and especially in the area of Business – IT alignment, in the area of Enterprise Architecture, in Service – Orientation and also in Software development and especially in transformations of software systems, and finally is used as a centric idea for making software for the digital dynamic enterprises of tomorrow.

2.1.1 The beginning of Capabilities

The term capability had been first used in 1965, in the field of Corporate Strategy by Ansoff, who spoke about managerial and functional capabilities, meaning the firm's skills levels in functions such as R&D, purchasing and marketing. However Ansoff did not describe capabilities as components of strategy (Long & Vickers-Koch, 1995).

But it was not until 1980's and 1990's that in the field of Strategic Management managers firstly began talking about resources (Wernerfelt, 1984; Barney, 1991), then competences (Phahalad & Hamel, 1990) and finally capabilities (Barton, 1992; Stalk et al, 1992; Long & Vickers-Koch, 1995; Teece et al, 1997; Tallman & Fladmoe-Lindquist, 2002), as a core component that leads an organization in achieving and maintaining a competitive advance.

First Wernerfelt (1984) attempted to look firms in terms of resources (e.g. brand names, in-house knowledge of technology, employment of skilled personnel, trade contracts, machinery, efficient procedures, capital etc) and issued a **resourced – based view** of a firm. Following this approach Barney (1991) attempted to provide a **resource – based**

framework and mentioned that firm resources include assets, capabilities, organizational processes, firm attributes, information, knowledge etc and finally classified them in three types: a) physical capital resources, b) human capital resources and c) organizational capital resources. [Prahalad & Hamel \(1990\)](#) issued a new strategy, which was an extension of the resource – based view, the **Competence – based view** of a corporation. According to them core competencies derived from the consolidation of corporate wide technologies and production skills and are corporate resources.

Simultaneously in 1990 **Teece, Pisano and Shuen** began talking about capabilities and defined them as *“a set of differentiate skills, complementary assets, and routines that provide the basis for a firm’s competitive capacities and sustainable advantage in a particular business”* ([Barton, 1992](#)).

[Barton \(1992\)](#) taking into account the competence – based view of a firm ([Phahalad & Hamel, 1990](#)) examined the nature of core capabilities of a firm, focusing on their interaction with new product and process development projects. Her research was important in that period because she issues the four dimensions of core capabilities ([Figure 2](#)). Those where a) employee knowledge and skills, embedded in b) technical systems, b) managerial systems and c) the values and norms associated with the various types of embodied and embedded knowledge and with the process of knowledge.

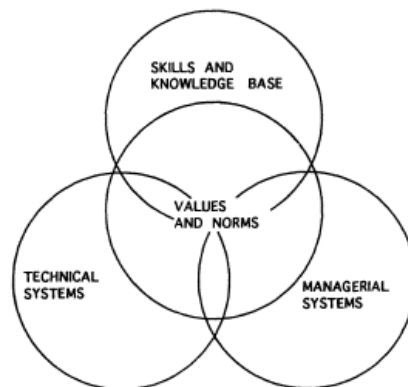


Figure 2: The four dimensions of core capabilities ([Barton, 1992](#))

At the same time **capabilities-based competition** became a new concept in corporation strategy ([Stalk, Evans, & Shulman, 1992](#)). According to this, business processes are the building blocks of the corporate strategy and competitive success depends on transforming a company’s key processes into strategic capabilities that consistently provide superior value to customer. Also companies create these capabilities by making strategic investments in a support infrastructure and finally CEO is responsible for this strategy, since capabilities necessarily cross functions.

Some years after managers started talking about a new type of organizations, the **capability-based organizations** (Long & Vickers-Koch, 1995). These organizations were placing core capabilities at the center of their strategic resources. The main question was “What capabilities do they need to develop and nurture to take full advantage of those changes” and the main components of their leadership agenda were vision, opportunity identification and capability assessment. Also Long & Vickers-Koch (1995) gave a definition of core capabilities illustrated in Figure 3.

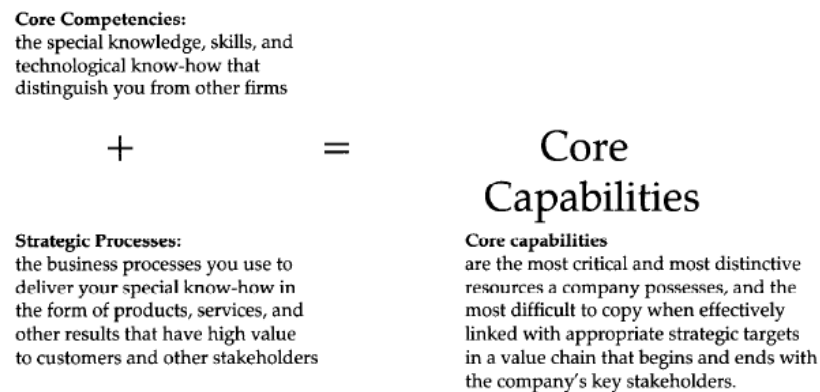


Figure 3: Definition of core capabilities (Long & Vickers-Koch, 1995)

Thereafter the notion of capability has been extended into that of **Dynamic Capabilities**, meaning the firm’s ability to integrate, build, and reconfigure internal and external competences to address rapidly changing environments (Teece, Pisano, & Shuen, 1997).

In the next decades the internationalization and globalization led organizations and researchers to looking for new strategies, in order to gain a sustained competitive advantage in the global marketplace. These strategies adapted models driven by the search of the competitive advantage from the internal knowledge resources and capabilities of an organization (Tallman & Fladmoe-Lindquist, 2002).

Tallman & Fladmoe-Lindquist (2002) presented the **Capability-driven Strategy framework** (Figure 4). This strategy considers the key factors to determine performance levels and the key forces to drive firms into international and global strategies, by the building, protection and exploitation of a set of unique capabilities. The framework suggests that the competitive advantage of an organization results from the possession of unique internal resources and capabilities and his ability to apply them in the marketplace. Also if it can continue develop new capabilities. Those capabilities called “resource-related capabilities” and were in two general types: a) business levels component capabilities and b) corporate level architectural capabilities.

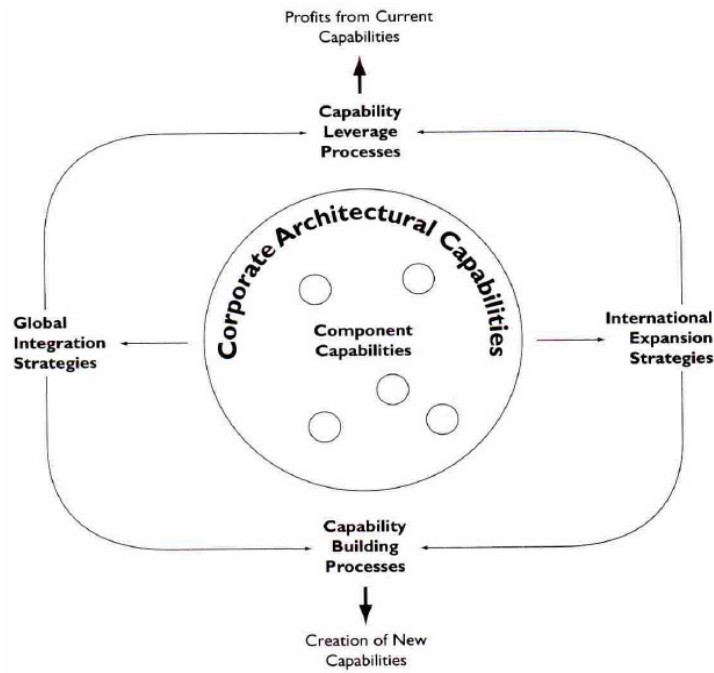


Figure 4: Capability – driven strategy (Tallman & Fladmoe-Lindquist, 2002)

Years after in order to empowered managers to make strategic decisions a **capability-based modeling paradigm** was introduced for representing business functions and processes, from the theories of the resource based view and competence – based view of a firm (Beimborn, Martin, & Holman, 2005). Beimborn et all (2005) talked about the **Capability Map** concept, which were a nested hierarchy of capabilities and a taxonomy diagram that describe the interplay of capabilities while doing business. As they stated “the concept of capability modeling was designed from the need to get a more steady picture of a firm (compared to existing methods of process and organization modeling), which enables managers to evaluate consequences of strategic decisions (which affect synchronously process, data flows and the firm’s size in terms of vertical integration)”.

Taking into account Capability Map concept (Beimborn, Martin, & Holman, 2005), a year after Ulrich Holman (2006) issue the **Business Capability** concept. By this way Capabilities start becoming a concern not only in Strategic Management theories but in Enterprise Architecture, in Service – Orientation, in Business – IT alignment and in software development, as we will discuss in more detail in the next section.

2.1.2 Business Capability

Business Capability is an abstraction that had first been discussed by **Ulrich Holman in 2006** and applied in the field of Enterprise Architecture and Service Oriented Architecture (SOA). [Ulrich Holman \(2006\)](#) in order to prevent from following architecture mistakes of the past, to ensure that the chosen implementation architecture relates to the actual desired state of the business and to prolong the life expectancy of the implementation in ever-changing environment, introduced a more stable foundation focusing on “*what a business actually does that create values for customers and not how it does it*”.

He defined Business Capability as “***the particular ability or capacity that business may possess or exchange to achieve a specific purpose or outcome***”. He mentions that “*Business Capability abstracts and encapsulates the people, process/procedures, technology and information into the essential blocks needed to facilitate performance improvement and redesign analysis*”. He also talked about the use of a taxonomic diagram to describe the network of capabilities used in business, ***the business capability model*** ([Figure 5](#)). This model describes capabilities by a ***Capability map***, which is a hierarchical description, where each level is decomposition of one or more capabilities at a higher level. According to him in the business capability model the Level 1: Foundation Capabilities address the entire ecosystem of the business and represent two categories of capabilities: the operational and the environmental capabilities as shown in [Figure 6](#).

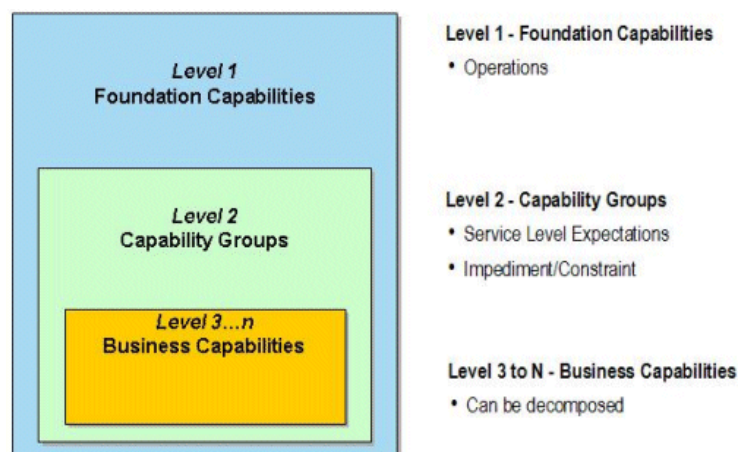


Figure 5: Business capability model taxonomy ([Holman, 2006](#))

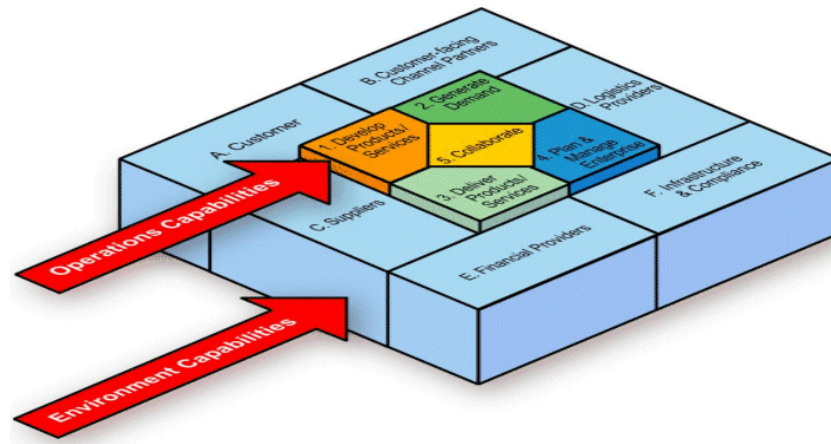


Figure 6: Level 1 Foundation Capability Model – Operational and Environmental Capabilities (Holman, 2006)

By this way Cook (2007) being a part of examination a real case study of a Phone Company, highlight that this new process of modeling a business, **the Business – Capability mapping**, has main goal to model the business on its most stable elements.

Since then there have been numerous publications in the literature, within the field of Information Systems, originating from different resources, with specific research aim/objectives, theoretical perspective/ framework and findings, as shown in [Appendix: Table 1](#).

Some of the publications introduce frameworks and roadmaps for constructing and modeling Business Capabilities in the field of Enterprise Architecture in order to help organizations to achieve a competitive advance (Brits, Botha, & Herselman, 2007), other to make them have profit and to make Enterprise Architectures more effective (Keller, 2009). Brits et al (2007) introduced a conceptual framework for Business Capabilities modeling consisting of a matrix for analysis (Figure 7) and feedback loops for development (Figure 8). Additionally Keller (2009) explained the basic idea of capability – based modeling and provide examples for the use of capabilities in Enterprise Architecture Management. Those were the “Heat Mapping”, the “Footprinting” and the “Mix the Models”. Also he adopted **the Forrester Search definition of Business Capabilities** in which capabilities relating to IT planning and were the building blocks of a business, represent stable business functions, were unique and independent from each other, were abstracted for the organizational model and capture the business interests (Keller, 2009).

The Conceptual Framework for Modeling Business Capabilities						
	<i>Elements of Guidance</i>	<i>Business Processes</i>	<i>Resources</i>	<i>Technology</i>	<i>People</i>	<i>Objects</i>
External Environmental Knowledge						Industry Foresight and Customer Insight
Ends						Vision, Goals and Objectives
Internal Environmental Knowledge						Entity-Relationship Diagrams, Functional Decomposition Diagrams, Flow Charts, Prototypes (Organizational Learning)
Means						Mission, Strategy and Tactics

Figure 7: The conceptual framework for modeling Business Capabilities (Brits, Botha, & Herselman, 2007)

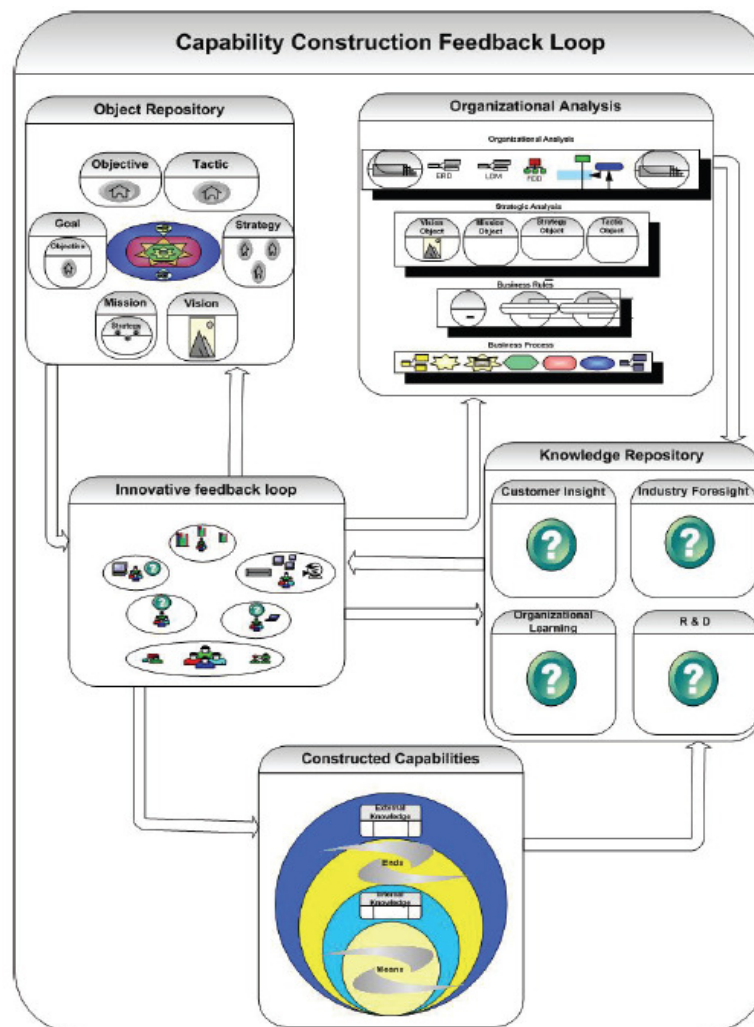


Figure 8: Capability Construction Feedback Loop (Brits, Botha, & Herselman, 2007)

Other publications introduce techniques for constructing Business Capabilities in the fields of Business Strategy and Business Architecture with main issue to help an organization to make decisions (Greski, 2009-A; Greski, 2009-B). Worth mentioning that Greski (2009-B) refers that “business capabilities represent the next level of detail, beneath the business strategy”. He defined them as **“an ability or capacity for a company to deliver a value, either to customers or to shareholders”**. Also he categorized them to customer – facing capabilities and operational capabilities, and referred that the first one deliver directly value to customers (e.g. a network of retail stores, a product or service offering or a transportation service) while the second one deliver value to shareholders (e.g. strategic planning, mergers and acquisitions, and financial planning). He also said that Business Capabilities consists of three major components as shown in Figure 9.

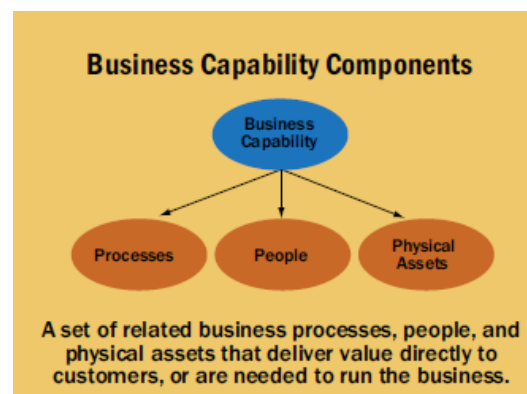


Figure 9: Business Capability Components (Greski, 2009-B)

Then some authors investigated Business Capabilities on the side of its strategic impact. Especially Bakhtiyari & Adel (2012) during their research about the strategic impact of business capabilities, they refer that capabilities endow competitive advance, enable an organization to perform at level that required to success and finally are one of the most strategically relevant artefacts of an organization.

Thereafter some other publications began discussing about **the importance of Business Capabilities in the alignment and communication between Information Technology and Business**. Scott (2009) referred that capability models provide the “Rosetta Stone” through which business needs aligned IT action, provide a focal point for strategic dialogue and they are the core components of the overall business architecture framework. Also he mentions that companies using capability maps to create value and IT architects and planners can take capabilities as the starting point for discussion about IT investments. Rosen (2010) mentioned that business capabilities provide the link between two complex and disparate environments: The Business and IT Architectures. He also mentioned that

analysis of the values streams leads to identification of business capabilities, while Capability Maps link the capabilities up to the strategies, goals, objectives, products and services. Also he support and down to the process, application, systems, services and sourcing that implement them (Figure 10). Finally he categorize hierarchical the level 1 of capabilities as Strategic, Value Added and Commodity. Thereafter Rosen (2012) discussed the difference between Business Capabilities, Value Streams and Processes. He refers that Processes and Values Streams require Business Capabilities and describe how those are used. He then mentions that Processes describe how something is done and Value Streams how value is delivered to a stakeholder, while Business Capability describes what is done.

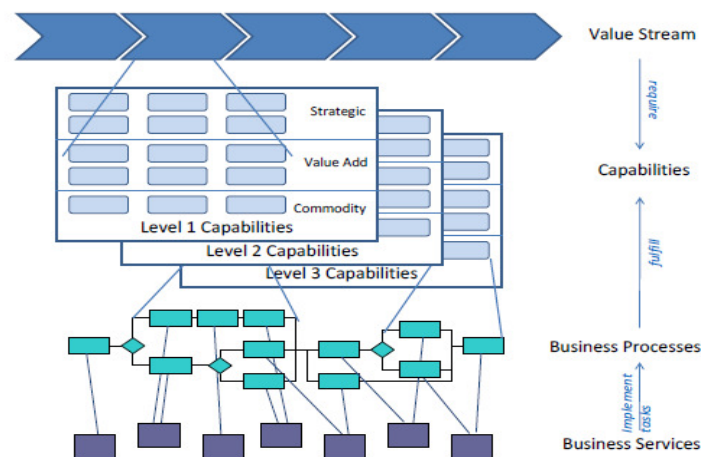


Figure 10: Value Streams leads to identification of Business Capabilities (Rosen, 2010)

Continuing in literature about the Business and IT alignment, Business Capabilities had been considered **as a core component and centric idea in enterprise models** the following years. Especially Barroero et al (2010) provide the Business Capability Centric Extension (BCCE), in the TOFAG core structure, which introduces a Business Component concept, including people, processes and technology (Figure 11). By this extension they achieved the linking between the business strategy and IT strategy, the linking between business component concept and the related information architecture, and the modularization of IT architecture by the Business Component.

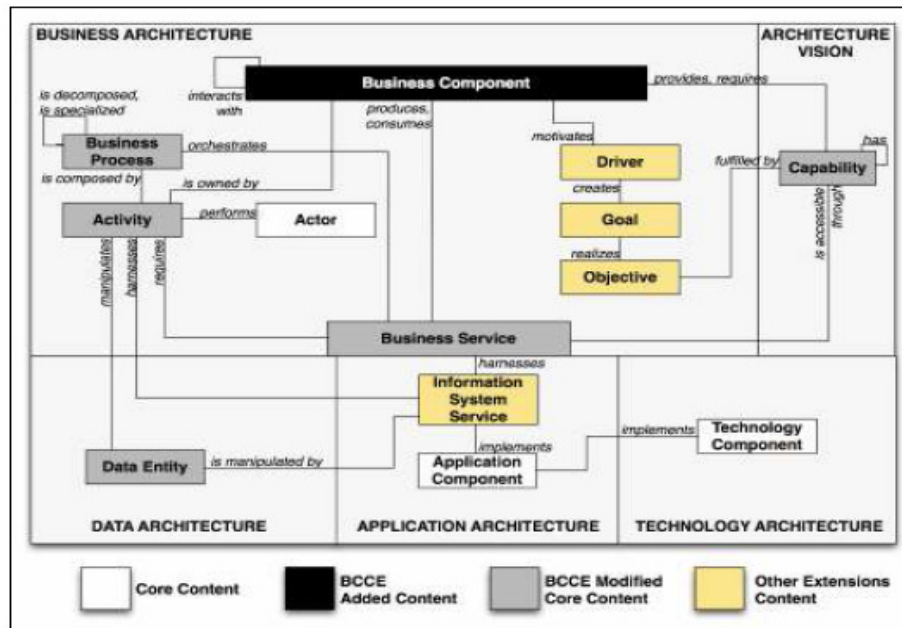


Figure 11: Business Capabilities Centric Extension (BCCE): Changes to Meta-model (Barroero, Motta, & Pign, 2010)

Freitag et al (2011) highlighted the importance of business capabilities as an essential element of the Enterprise Architecture Management (EAM) approach and provide a Capability Dependency Analysis Method (Figure 12) between capabilities and the other elements of the Enterprise Architecture. They defined them as *“a functional building block of the business which supports the business models and the business strategy, i.e. it defines the organization’s capacity to successfully perform a unique business activity”*.

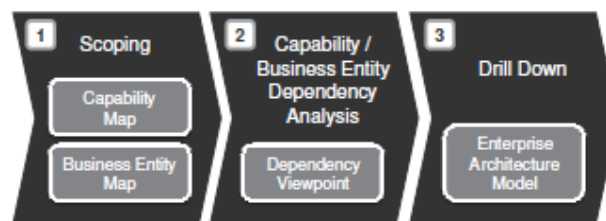


Figure 12: Capability Dependency Analysis Method (Freitag, Matthes, Schu, & Nowobilska, 2011)

At the same time Ulrich & Rosen (2011) introduced a capability mapping framework, a method of incorporating capability into Business Architecture and generally in Enterprise Architecture, a method for Business/IT roadmap development and by that prove that Business Capability provide the high-level foundation for alignment and bridges the Business/IT Chasm. They refer that *“a business capability or a simple capability defines what a business does. It does not communicate or expose where, why or how something is*

done – only what is done. According to them capability relates with other aspect of business as shown in Figure 13.

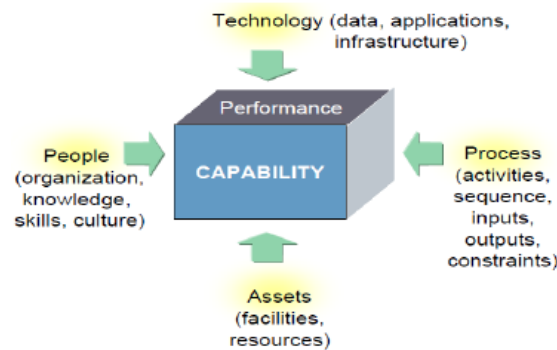


Figure 13: Capability relation of business aspects (Bakhtiyari & Adel, 2012)

In their work they provide a more clear decomposition of the level 1 of Capability Map, by an example as shown in Figure 14. According to that they refer that the “strategic” layer include capabilities that reflect executive properties, the “value-added” tier goes to the heart of what a business does to ensure viability and thrive in the market place, and finally the “Support” layer represents certain abilities that an organization may have to function as business (Ulrich & Rosen, 2011).

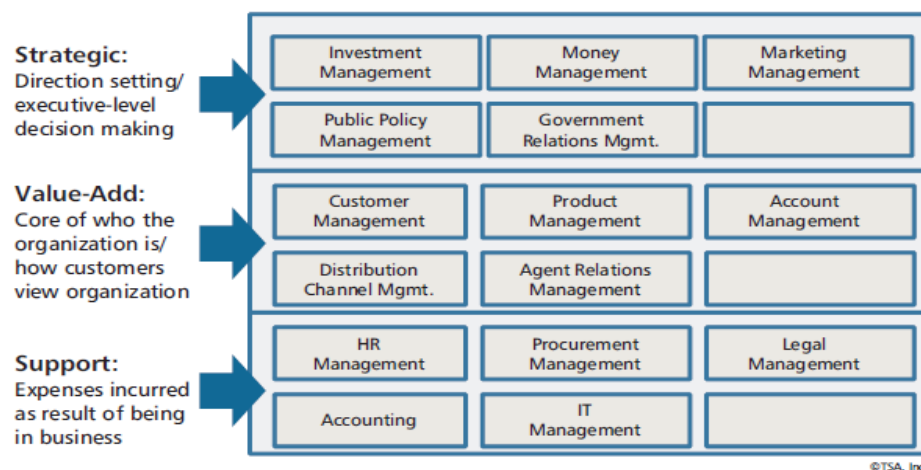


Figure 14: Example Level 1 Capability map (Ulrich & Rosen, 2011)

Also they provide a clear picture of the role of Business Capabilities in Enterprise Architecture (Figure 15), meaning the linking between business requirements and the IT solutions (Ulrich & Rosen, 2011).

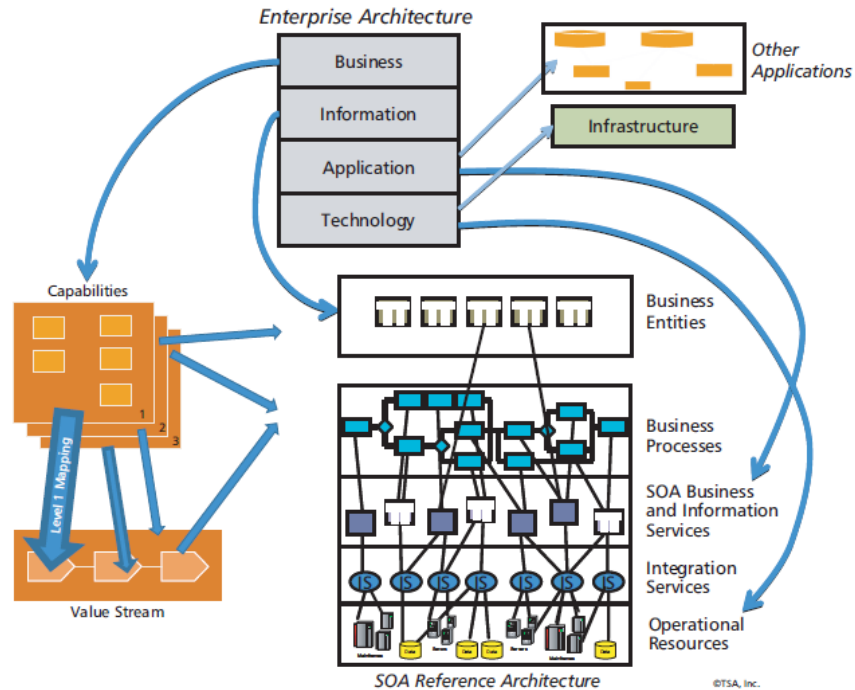


Figure 15: The Role of Business Capabilities in EA (Ulrich & Rosen, 2011)

Finally they talk about the importance of **a Business Architecture Knowledgebase** in which information about business, including organizational structure, capabilities, value streams, information assets, project initiatives, customers and partners, and related IT assets, are stored, related and viewed in a database (Ulrich & Rosen, 2011).

Despite the previous worth mentioning that the term of capability had cause confusion for long time from managerial theories to Informatics. That's because during the time different definitions came into light in different fields. For that purpose Vaughan (2011) held a review of bibliography in order to give a definition of Business Capability, according to business strategy, operations and Informatics. He defined Business as **"the potential of a business resource (or groups of resources to produce customer value by acting on their environment via a process (P) using other tangible (Rt) and intangible Resources (Ri)"**. He also talked about two types of capability that may be **internal of external to business**. External capability occurs where the potential output is of core importance to customer benefit. Internal capability is where the potential output is delivered within the business (Vaughan , 2011). Relevant Tell (2014) in order to make comprehensible the meaning of capability, held a reached with observations of capability definitions, theories and approaches and asked the question what capability is not.

Two years before Tell's work Stirna et all (2012) defined capability as **"the ability to continuously deliver a certain business value in dynamically changing circumstances"** and

introduce a meta – model that integrate organization development with IS development, taking into account changes in the application context of a solution. This was the Capability Driven Development (CDD), which will be discussed in more detail in the next section.

In some cases Business Capabilities has being used in the area of software development during the transformation of systems, as a design and development pattern in Software Architecture. Frey et al (2013) described the **Capability – Based Service Identification pattern** that had been used in the moving from legacy applications to SOA – based Architectures. This pattern identified services and defined the service model based on a model of Business Capabilities. The most important achievement of this pattern is that it facilitates a durable alignment between business and IT in a SOA by using a top-down solution (Figure 16).

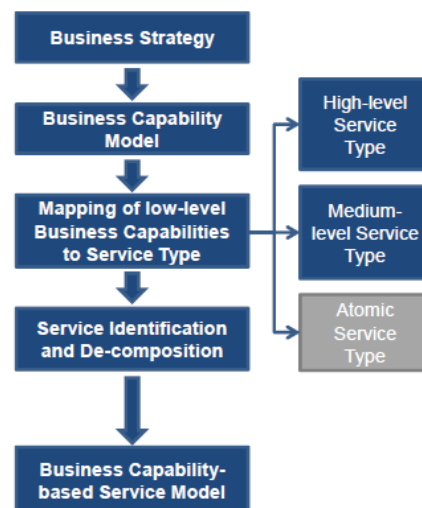


Figure 16: Top – down approach for Capability – Based Service Identification (Frey, Hentrich, & Zdun, 2013)

From so on we have seen that Business Capabilities had been an important research concern in Enterprise Architecture, in Service – Orientation design paradigm of building software (i.e. SOA patterns), in Business – IT alignment, in transformation of software systems and in Business Strategy. Next section deals with a new trend that combines enterprise modeling, context modeling and capability modeling in order to help an organization to deal with changes in the dynamic environment that operates.

2.1.3 Capability as Service

So far we have seen that Business Capabilities has been used during the alignment between Business and Information Technology (Scott, 2009; Rosen, 2010; Barroero, Motta, & Pign, 2010; Freitag, Matthes, Schu, & Nowobilska, 2011; Ulrich & Rosen, 2011; Stirna, Grabis, Henkel, & Zdravkovic, 2012) and sometimes as a centric idea and core component of Enterprise Architecture (Barroero, Motta, & Pign, 2010; Freitag, Matthes, Schu, & Nowobilska, 2011; Ulrich & Rosen, 2011). Business Capability also leads to competitive advance (Brits, Botha, & Herselman, 2007; Bakhtiyari & Adel, 2012) and provide business value (Scott, 2009; Greski, 2009-B). Thus it is clear that enterprises should focus on their Business Capabilities in order to archive growth.

But the extensive use of Internet and its variability has made modern organizations to operate in dynamically changing environments, where the circumstances in which Information Systems (IS) operates and the context of use was not always predictable (Stirna, Grabis, Henkel, & Zdravkovic, 2012). As Stirna et al (2012) states “**context is any information that can be used to characterize the situation**”. It describes circumstances such as geographical location, platforms and devices used, and as well as business conditions and environment“. This definitions shows that context cannot be stable.

From the previous began the need for adopting new solutions in Information Systems and Software development that can help for rapid response to changes in the business context, for the development of new capabilities and for run-time configuration and adjustment of applications (Stirna, Grabis, Henkel, & Zdravkovic, 2012).

Thus Stirna et al (2012) develop a new method named **Capability Driven Development (CDD)** that integrates organizational with IS development taking into account changes in the application context of a solution. The foundation of the CDD approach was a meta-model that consists of goals, key performance indicators, capabilities, context and capability delivery patterns, and make uses of enterprise modeling (EM) techniques as a starting point of the development process (Figure 17). This meta-model has three sections the Enterprise and Capability modeling, the Capability Delivery Context modeling and the Capability Delivery Patterns.

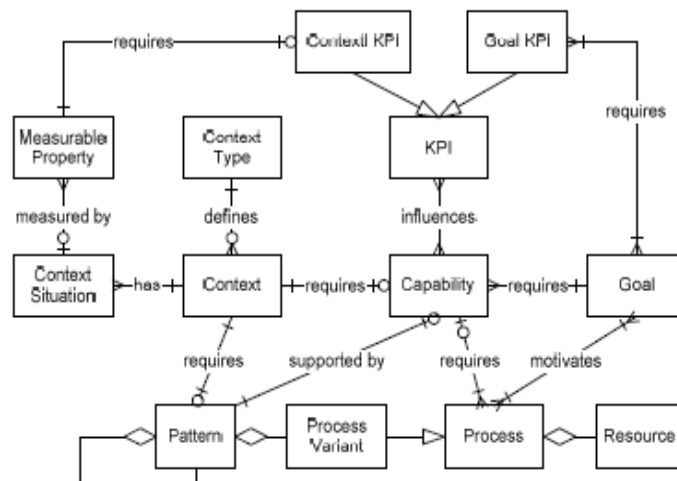


Figure 17: The capability meta-model (Stirna, Grabis, Henkel, & Zdravkovic, 2012)

A year after European Commission announced the project “**Capability as Service in digital enterprises - CaaS**” and aimed to facilitate configuration of business services and development of executable software to monitor the changes arising from the business context. By this project they supported the idea that business capabilities deliver needs to be based on the application context and the main goal were to bring about a shift from the service – oriented paradigm to a capability delivery paradigm (European Commision, 2013).

The result of CaaS project were the delivery of CDD methodology, for making software for the digital enterprises of tomorrow which is situated upon the existing information technologies services (Figure 18), in form of:

- 1) Modeling languages for representing enterprises designs, context models and patterns,
- 2) A methodology for detailing how capabilities may be specified and how these may be used for designing new services,
- 3) Reusable best practices and capability delivery patterns,
- 4) Algorithms for dealing with business context awareness and service re-configuration,
- 5) A tool environment for modeling design and delivery and
- 6) A set of case studies demonstrating the applicability of the CDD (European Commision, 2013).

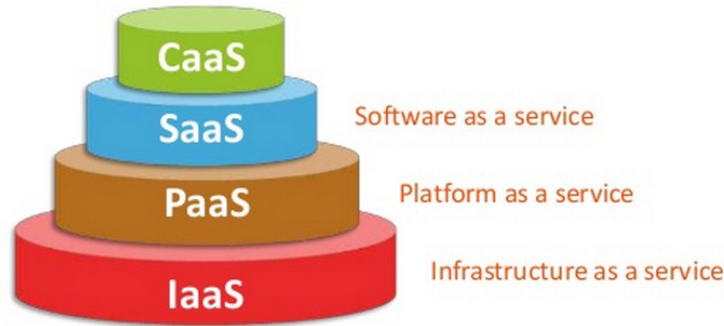


Figure 18: CaaS upon the existing information technologies services (FP7 Collaborative Project with No 611351 , 2014)

Also the project CaaS was driven by three empirical use cases from different business domains namely energy, compliance and e-government (European Commission, 2013). The first one was the **FP7 project EnRiMa** – Energy Efficiency and Risk Management in Public Building, with main scenario the exchange of data between the grid operator and the energy supplier. The second one referred to a case study of provision services for regulatory compliance in the maritime industry by **Flesh TL** Company to Danaos maritime company. Finally the third referred to a case study of improving a Service – Oriented Architecture (SOA) platform for e-government by the **Everis** Company, with emphasis to put on electronic services provided to municipalities and used by citizens and companies.

Within the FP7 project of CaaS a methodology areas relevant to CDD was given by [Berzisa et al \(2013\)](#) in order to give an input to defining the CaaS base methodology. Those were the Capability design and development, the pattern elicitation, the context modeling, the runtime adjustments, the identification of best practices and the Enterprise modeling. Since then several researchers focused in giving the essential tools, methodologies and empirical experience for the application of CDD approach, according to the Caas Project ([Zdravkovic et al, 2013](#); [Espana et al](#); [Bravos, Loucopoulos, Stratigaki & Vavlis, 2014](#); [Bravos, Gonzslez, Grabis, Henkel & Jokster, 2014](#); [Bravos, Grabis, Henkel, Jokste & Kampars, 2014](#); [Stratigaki et al, 2014](#); [Berzisa et al , 2015](#)).

Firstly [Zdravkovic et al \(2013\)](#) extent Stirna' s work by proposing a meta-model for capability design and delivery, with the consideration to delivering as cloud services ([Figure 19](#)). To exemplified his proposed approached he used the EnRiMa use case. The aim of their research as stated by them was to contribute to the business-driven application development and the emergence of new kinds of interoperable cloud-based services, and to set the tools that would support the CDD approach.

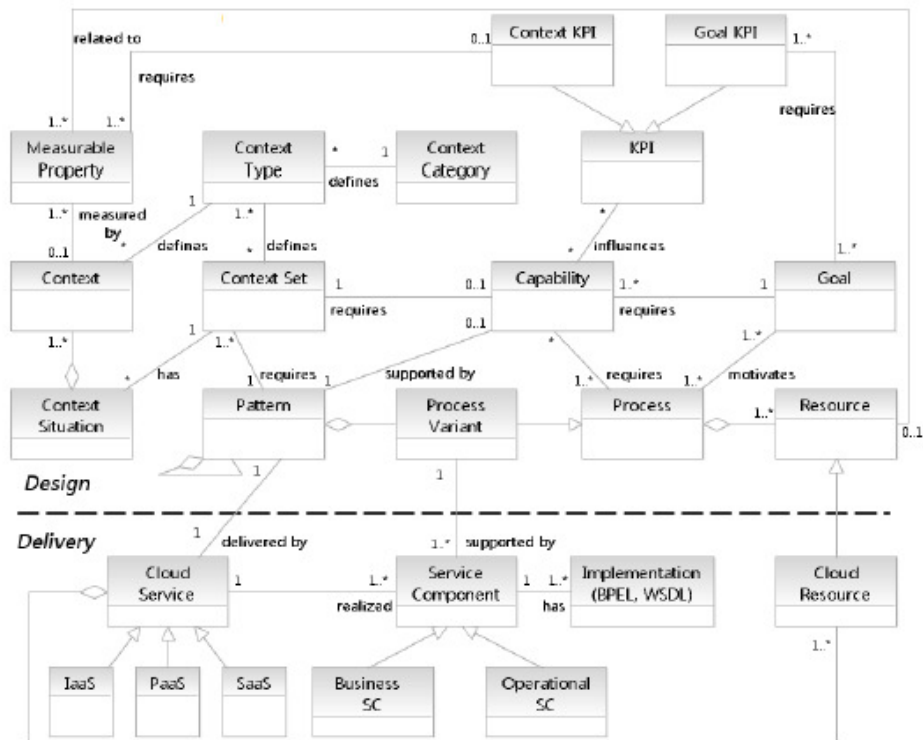


Figure 19: Meta – model for capability design and deployment in cloud (Zdravkovic, Stirna, & Henkel, 2013)

Also Zdravkovic et al (2013) describe the main components of the CDD environment (Figure 20), which were the capability design tool for the capability design, cloud services for the capability delivery and the context platform for capturing context data.

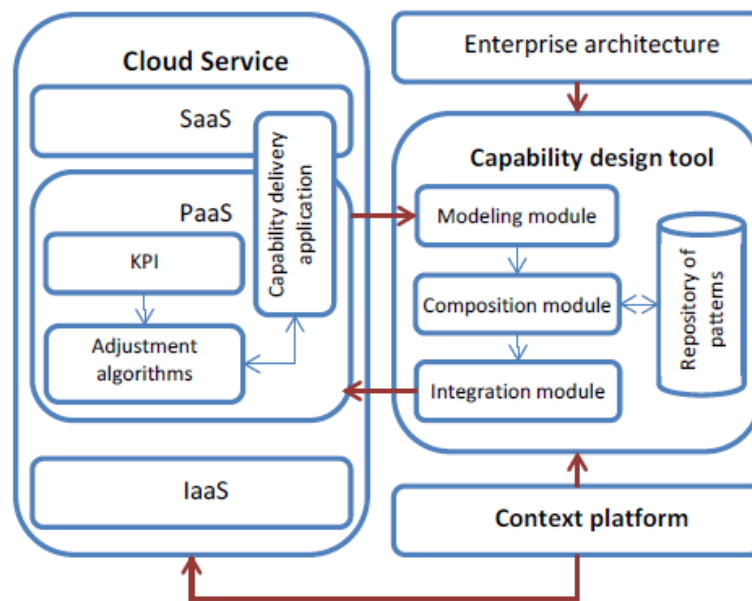


Figure 20: Capability driven development environment (Zdravkovic, Stirna, & Henkel, 2013)

In the meantime [Loucopoulos et al \(2013\)](#) working with the use case of provision services for regulatory compliance in the maritime industry by **Flesh TL** Company to Danaos maritime company, they provide a deliverable of Capability models for Business Compliance Controlling and Auditing, in order to apply the CaaS methodology and its support tool environment. In their work they define Business Capability by providing a conceptual meta-model which will be discussed in more detail in chapter 3.

Then [Espana et al \(2014\)](#) report a case study that focuses on capability modeling within a service oriented architecture development project, in order to mention the lessons learned and the open challenges to feedback the improvement of the CDD approach. They use the case study of EVERIS Company with unit of analysis the project to improve a Service – Oriented Architecture (SOA) platform for e-government. They approached CDD by creating a goal model ([Figure 21](#)) in the first place so that the rest of the models (e.g stakeholders, context) to be reasoned taking this model as input.

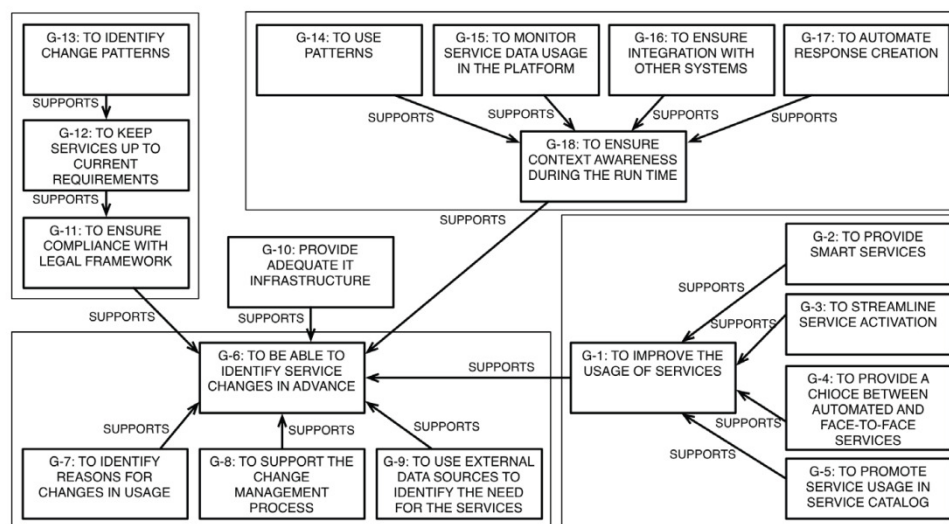


Figure 21: Goal Model of the project ([Espana, Gonzalez, Grabis, Jokste, Juanes, & Valverde, 2014](#))

Thereafter [Bravos, Loucopoulos, Stratigaki & Vavlis \(2014\)](#) being supported by European Commission Project CaaS (611351), investigate the utility of modeling Business Capabilities in the [Zdravkovic et al \(2013\)](#) initial version of CaaS meta-model, by answering whether such a meta-model could provide the sufficient guidance for repeatable design activities by different designs working on the same problem, when using design rationale techniques. They provide two cases for representations of goals by two different modelers and the rationale behind the instantiations, in which resulting in different capability definition and thus differences in the implementation of the same meta-model to the same use case scenario. According to this they propose that future work would include accurate

and thorough definition of all supportive modeling languages required, toward a complete capability meta-model able to support CDD.

Within the same year [Bravos, Gonzslez, Grabis, Henkel & Jokster \(2014\)](#) discusses the initial Capability modeling experiences with main emphasis on the capability design phase of CDD approach. Their main objective is to evaluate the expressiveness of capability meta-model in the three different empirical cases (Everis, Flesh TL and MSCONS). By their research they conclude that capability meta-model is sufficient for modeling business cases because it is flexible enough to represent different business cases and to adjust to the varying needs of the various partners.

Also [Bravos, Grabis, Henkel, Jokste & Kampars \(2014\)](#), defined a set of concrete key goals to be achieved in order to have a methodological support (a goal graph) to CDD approach for creating software systems that can adapt to changes, taking account three different empirical cases (Everis, Flesh TL and MSCONS).

At the same year [Stratigaki et all \(2014\)](#) designed a meta-model as a foundation for compliance capability, in order to support a compliance development methodology that would help in achieving that certain business processes comply with the regulations during the capability modeling of CDD approach.

Finally [Berzisa et all \(2015\)](#) taking into consideration the previous researches that presented in this section, provide a final CDD approach as shown in [Figure 22](#) which include three phases. The Enterprise Modeling, the design and the delivery. Also mentions that the main challenges to be addressed are the availability of patterns and the implementation of algorithms for dynamic adjustment of the capability delivery application.

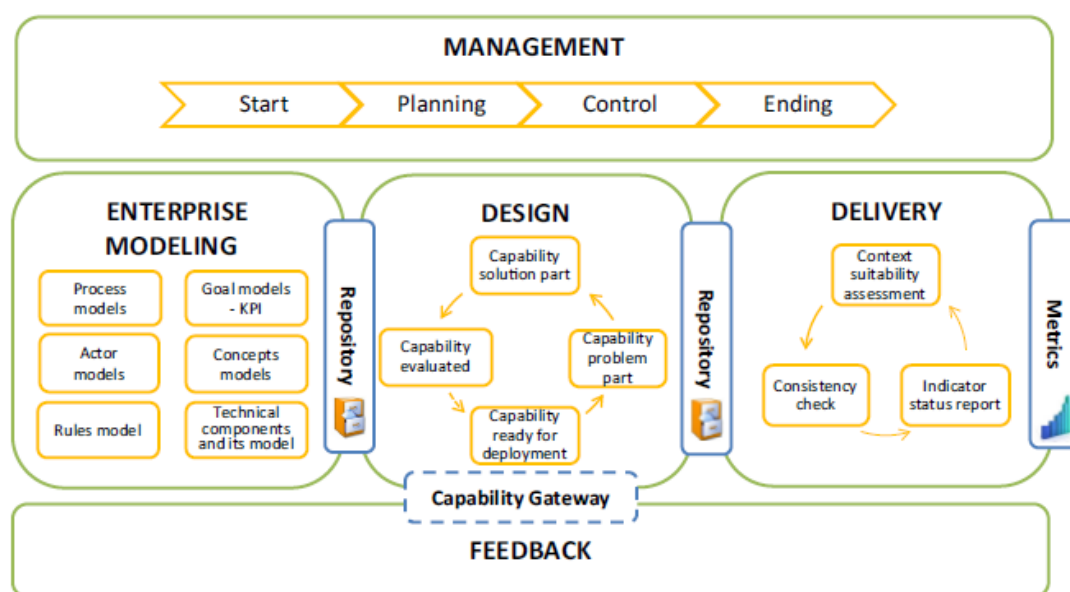


Figure 22: CDD Methodology (Berzisa, et al., 2015)

2.2 Conceptual Modeling

2.2.1 Database Design

Conceptual modeling is a widely applied practice that aims to create an abstract of representation of a situation, using models for that purpose (Thalheim, 2011). These models are means by which we can capture the universe of discourse (UoD) in an abstract way and communication tools among stakeholders (designers, programmers, users, managers etc).

The term of conceptual modeling was consolidated in 1984 by Mylopoulos and Schmidt in Brodie (Roussopoulos & Karagiannis, 2009). As defined by Mylopoulos (1992) “*Conceptual modeling is the activity of formally describing some aspects of the physical and social world around as for purpose of understanding and communication*”. This modeling practice plays an important role in a variety of area in Computer Science. More specifically as mentioned by Roussopoulos & Karagiannis (2009), “*it has found applications in a variety of fields, including information system design, knowledge representation for Artificial Intelligence, modeling of organizational environments, business processes, software development processes, software requirements, or just plain modeling some part of the word for purpose of human communication an understanding*”.

A Database has become the necessary mean for information storage and retrieval and is one of the most important components of an Information System. Thus conceptual modeling has also application during the building of a database. In particular databases the same as Information Systems has a ***specific lifecycle with the following key stages*** (Jackson, 1996):

1. *Requirement Analysis*
2. *Design (comprises conceptual, logical and physical)*
3. *Implementation*
4. *Testing*
5. *Operation*
6. *Maintenance*

During the stage 2 of “Design” a conceptual modeling practice is used *for creating specific data models* in three levels of abstraction (conceptual design, logical design and physical design). Those data models are a conceptual representation of the data structures that are required by a database (Windows Enterprise Support Database Services, 2015). **The ‘Design’ stage in general follows five steps** (Windows Enterprise Support Database Services, 2015):

1. *Planning and analysis*

2. *Conceptual design*
3. *Logical design*
4. *Physical design*
5. *Implementation*

However professionals that have the responsibility to design and implement database systems must have special skills which are related to **data modeling**. In other words they must be well skilled in the methodologies, techniques and practices to data modeling. Next section provides a brief review about them.

2.2.2 Data Modeling Approaches & Techniques

Data Modeling is a technique for exploring the data structures needed to support an organization, and provides a method and means for describing the real-world information requirements in a manner understandable to stakeholders (Ponniah, 2007). The production of data modeling is data models which according to Ponniah (2007):

1. help the users or stakeholders understand clearly the database system that is being implemented based on the information requirements of an organization and
2. enables the database practitioners to implement the database system exactly conforming to the information requirements.

Professionals that design and implement database systems may choose between of different data modeling approaches, each of which may have different data modeling methods, techniques and languages. According to (Ponniah, 2007) those are:

1. *Semantic modeling*
2. *Relational modeling*
3. *Binary modeling*
4. *Entity – Relationship modeling*
5. *Fact – oriented modeling*
6. *Object – oriented modeling*

For the purpose of this dissertation we will limited our research to the most popular approaches for data modelers (Ponniah, 2007) which are the Entity – Relationship Modeling, the Fact – Oriented Modeling and the Object – Oriented Modeling. According to this the approaches, techniques, methods and languages that will be discusses are shown in Figure 23.

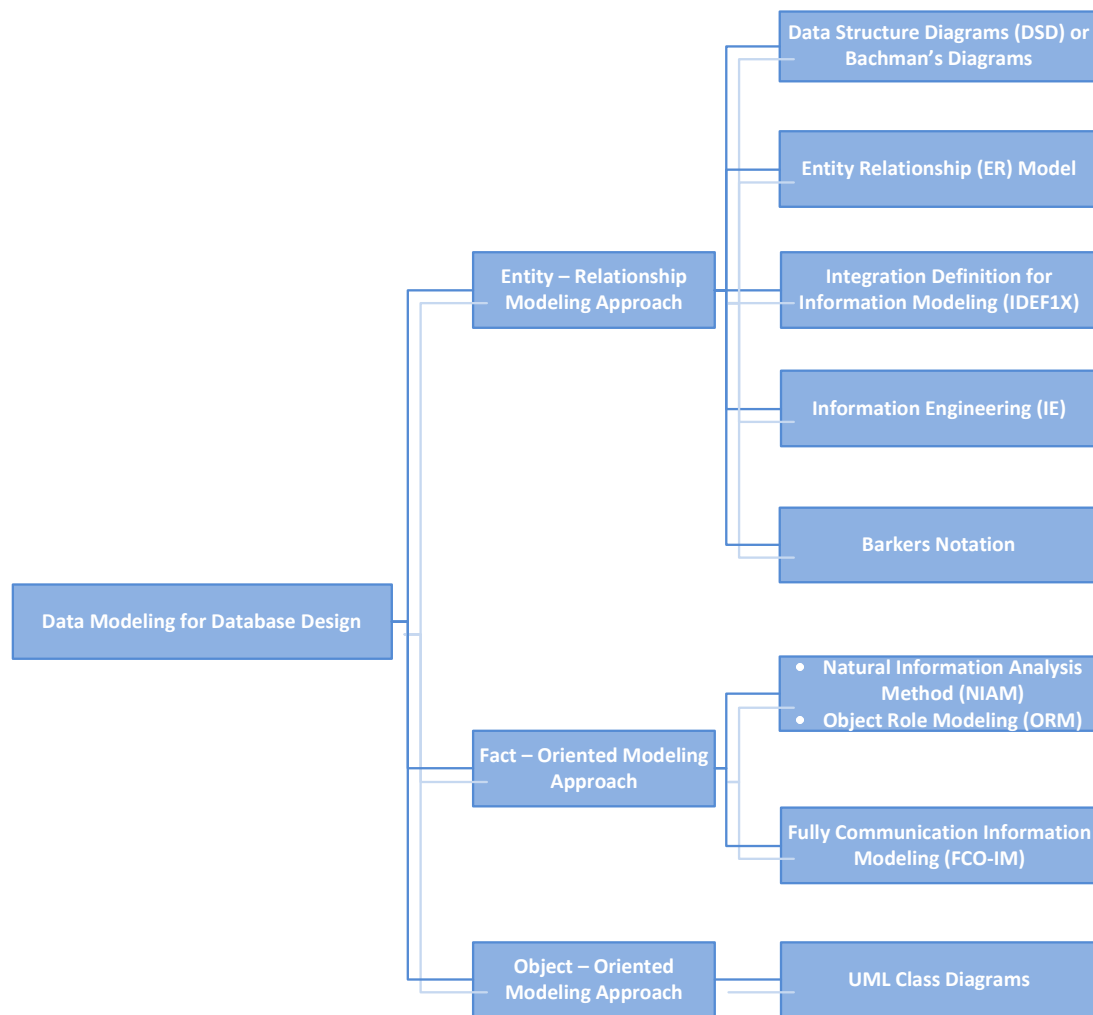


Figure 23: Data Modeling Techniques, Methods & Languages

Entity – Relationship Modeling Approach

Initially databases were designed on the basis the developer’s intuitive understanding of the subject domain, which were usually represented in graphical form (Kogalovsky & Kalinichenko, 2009). The most popular technique for this graphical form was in 1969 the ***Bachman’s Data Structure Diagrams (DSD)***, which were based on a type of notation dealing with classes of entities and the classes of sets that relate them (Bachman, 1969). According to Bachman (1969) the DSD were consisting of two kind of graphic symbols: the block to represent an entity class, and the arrow to represent a set class of and the roles of owner/member established by that set class. An example of two classes of entities in this technique is shown in Figure 24.

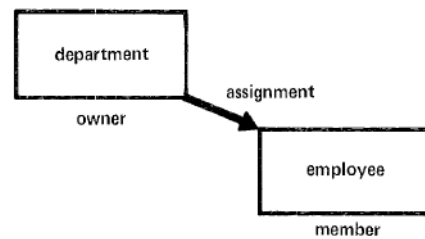


Figure 24: An example of Bachman Diagrams (Bachman, 1969)

The Data Structure Diagrams were a predecessor of **the Entity Relationship (ER) Model** technique for designing database, which were introduced in 1976 by **Peter Chen** (Chen, 1976) for logical design of data. The Entity Relationship Model depicts the data structures in terms of entities, relationships and attributes. A detailed description, understanding and graphical syntax or notation of ER Model can be found in several books such as (Sharron & Evan, 2006; Teorey, Lightstone, & Nadeau, 2006). The ER Model has been the most popular and influential model in the database community and because of that we can find a numerous of publications in bibliography from 1976 as shown (Chen, Song, & Zhu, 2007). By this technique Chen issued an **Entity – Relationship Modeling Approach** for designing databases. An example of ER Model is shown in Figure 25.

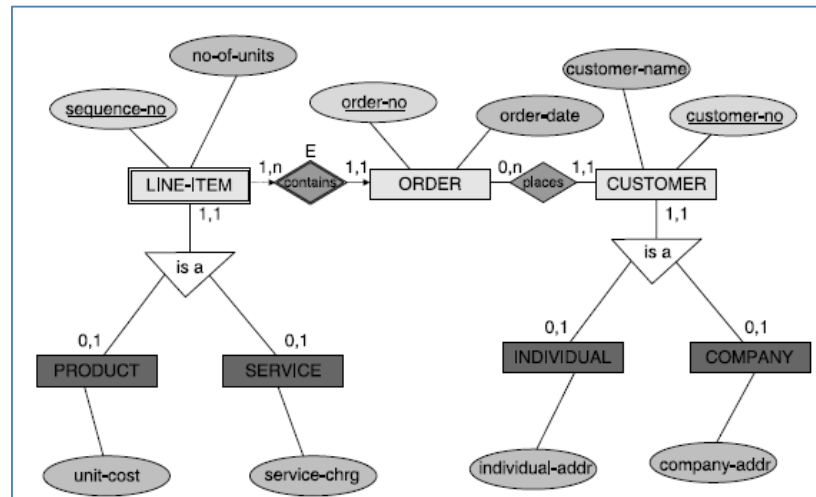


Figure 25: An example of ER Model (Ponniah, 2007)

During the time the ER Model has being found in various versions such as the Extended ER Model or Enhanced ER Model – EER (Teorey, Yang, & Fry, 1986), the E²R Model (Embley & Ling, 1989), the Higher-Order Entity Relationship Model – HERM (Thalheim, 1991), Temporal ER Models for capturing temporal aspects of data (Gregersen & Jensen, 1999) and the Star ER Model for data warehouse design (Tryfona, Busborg, & Borch, 1999). The Extended ER Model issued additional semantics such as ternary relationships, optional

relationships and the generalization abstraction (Teorey, Yang, & Fry, 1986). The E²R Model solved two main limitations and problems that had the ER and EER Models. Those were the distinction between attributes and entities, that can cause downstream redesign and the use of two different types of abstraction which may not support normalization (Embley & Ling, 1989). By the E²R Model Embley & Ling (1989) helped designers not to have distinguished between attributes and entities, and also support the normalization at the model level. The Higher-Order Entity Relationship Model issued the nesting of attributes and the procedure of mapping automatically the model to relational database schemata (Thalheim, 1991).

Since then the Entity Relationship Modeling Approach became the basis for the development of other techniques and languages for designing databases. Those were the **Integration Definition for Information Modeling – IDEF1X** (National Institute of Standards & Technology, 1993), the **Information Engineering – IE** (Finkelstein, 2006) and the **Barker's Notation** (Mamayev, 2013; Ponniah, 2007).

In more detail during the 1970s the U.S. Air Force Program for Integrated Computer Aided Manufacturing (ICAM) developed the IDEF technique, in order to increase manufacturing productivity through the systematic application of computer technology (National Institute of Standards & Technology, 1993). The IDEF technique has a list of methods being developed as shown in Figure 26:

IDEF0	Function Modeling
IDEF1	Information Modeling
IDEF1X	Data Modeling
IDEF3	Process Description Capture
IDEF4	Object Oriented Design
IDEF5	Ontology Description Capture
IDEF6	Design Rationale Capture
IDEF8	User Interface Modeling
IDEF9	Scenario-driven IS Design
IDEF10	Implementation Architecture Modeling
IDEF11	Information Artifact Modeling
IDEF12	Organization Modeling
IDEF13	Three Schema Mapping Design
IDEF14	Network Design

Figure 26: IDEF Methods (Mayer, Painter, & deWitte, 1992)

One of them is the **IDEF1X** which is an extended version of IDEF1 and deals with data modeling. The IDEF1X is most useful for logical database design after the information requirements are known and the decision to implement using a relational database has been made (Mayer, Painter, & deWitte, 1992). The IDEF1X was influenced by various methods, techniques and theories as shown in Figure 27. The IDEF1X uses a graphical form for representing the real world, the terms entities, attributes and relationships between

entities, and has been described in more detail in (National Institute of Standards & Technology, 1993) "Method Report". The IDEF1X uses specific notations as shown in Figure 28. A detailed description, understanding and graphical syntax of IDEF1X can be found in the book (Sharron & Evan, 2005). Finally an example of IDEF1X is shown in Figure 29.

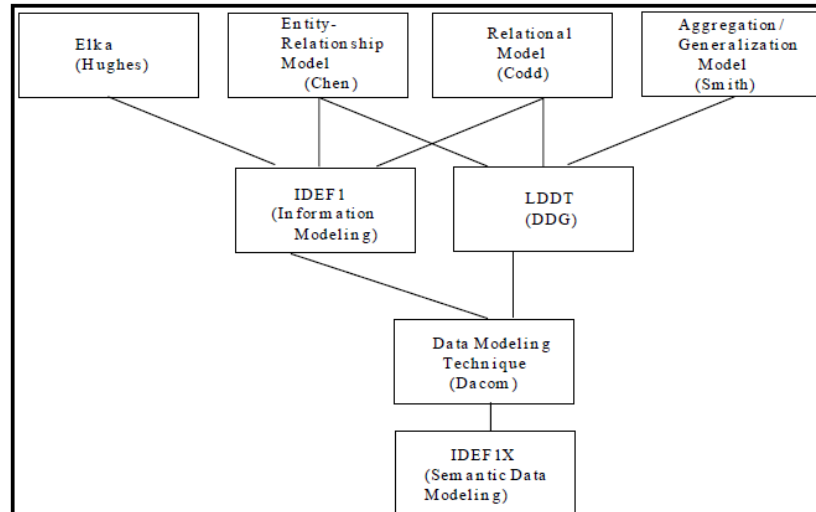


Figure 27: IDEF1X Origins (Mayer, Painter, & deWitte, 1992)

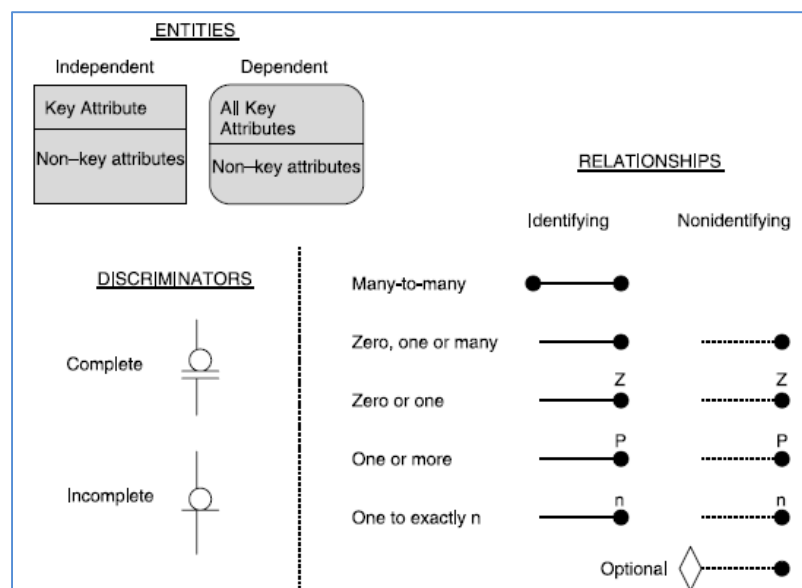


Figure 28: IDEF1X Notation (Ponniah, 2007)

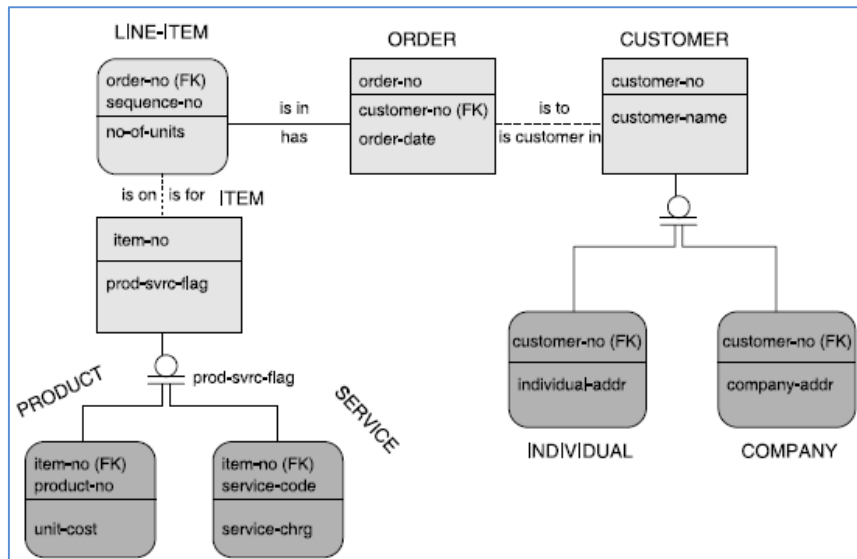


Figure 29: IDEF1X – An example model (Ponniah, 2007)

From 1976 – 1980 Clive Finkelstein and James Martin was working together in order to determine data and information that are required by business users in order to carry out their job responsibilities, and also they try to determine and analyze the processes that are relevant with that data. By this work they developed a series of methods which were: the Data Analysis (the rules of normalization was used in order to interview business users at operational levels), the Data Base Design (was used to identify the data and the information that was needed), the Information Analysis (was used to identify information needed by managers), the Procedure Formation (was used to derive processes from data), and the Distributed Analysis (was used to analyze and design remote distribution of data and processing). Combining all those methods they realized that they had developed a methodology for identification of Information and for the development of Information Systems which they called **Information Engineering (IE)** (Finkelstein, 2006). Information Engineering was first published in 1981 in a series of articles in the magazine Computer World USA (Finkelstein, 1981-A; Finkelstein, 1981-B; Finkelstein, 1981-C; Finkelstein, 1981-D). From 1982 – 1986 Information Engineering began to involve two distinct variants: the DP-driven IE Variant and the Business – driven IE Variant (Finkelstein, 2006). According to Finkelstein (2006) the first one deals with the development phases of an Information System and the second one with separate phases which were the Strategic Business Planning, the Data Modeling, the Process Modeling and the Systems Design & Implementation. For data modeling the Information Engineering methodology is using a specific notation that is shown in Figure 30. A detailed description, understanding and graphical syntax of IE can be found in the book (Sharron & Evan, 2005). An example of this notation is shown in Figure 31.

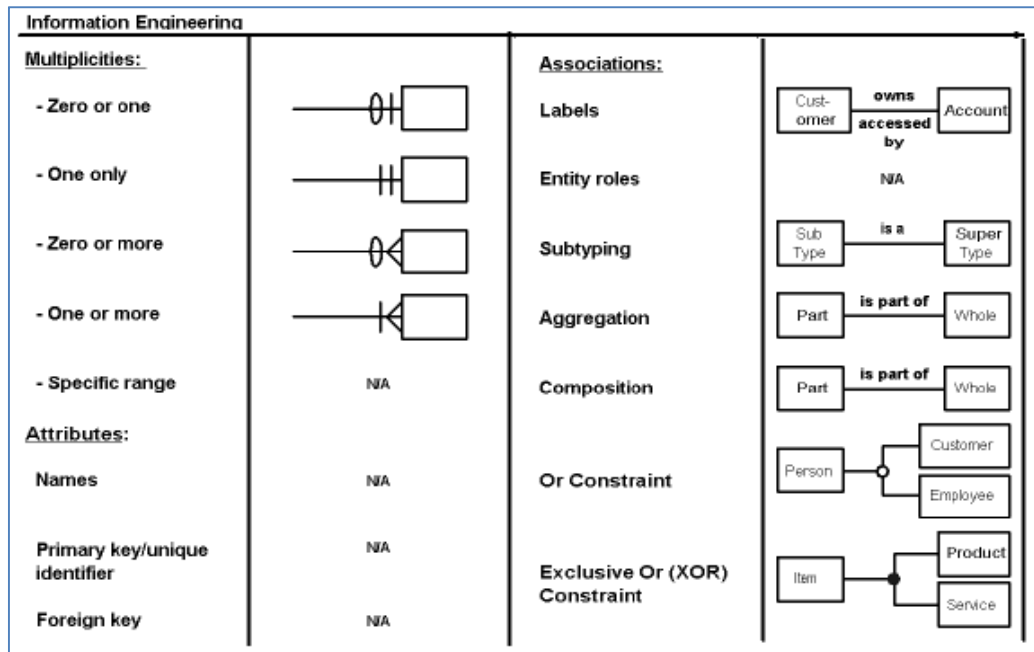


Figure 30: Information Engineering Notation (Wambler, 2015)

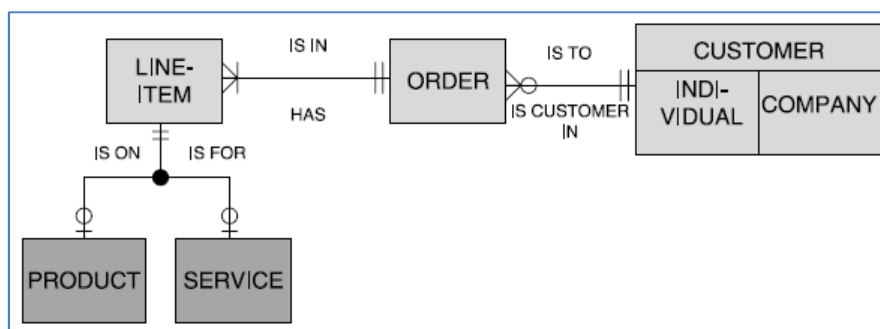


Figure 31: Information Engineering – An example model (Ponniah, 2007)

Finally in 1990 Richard Barker and Harry Ellis developed an **Entity – Relationship Diagram (ERD) notation**, which is described in more detail in the book “Richard Barker, CASE Method: Entity Relationship Modelling, Addison-Wesley Longman, 1990” (Mamayev, 2013). This notation was then developed and extended by Richard Barker as being a part of the Oracle Corporation, and marked as the Oracle Customer Development Method (CDM) (Mamayev, 2013). The ERD notation is a method for describing Entity Relationship Model and so it uses the terms of entities, relationships and attributes. The syntax of Barker’s notation is shown in Figure 32 and an example of Barker’s notation is shown in Figure 33. A detailed description, understanding and graphical syntax of ERD can be found in the book (Sharron & Evan, 2005)

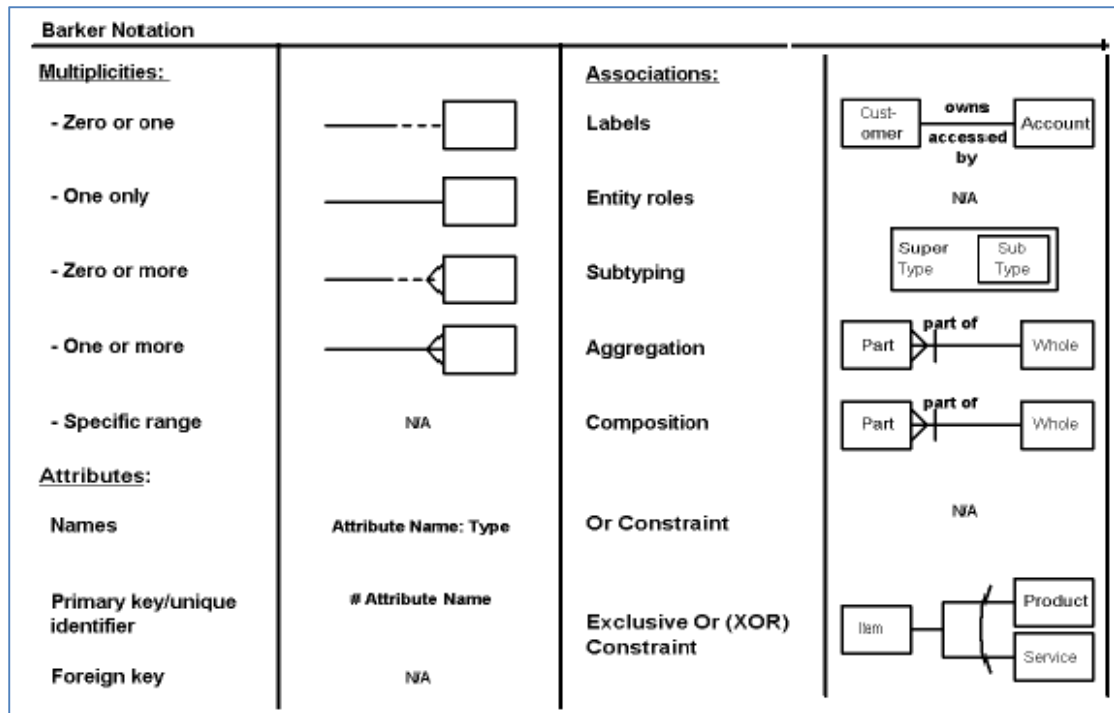


Figure 32: Barker's Notation (Wambler, 2015)

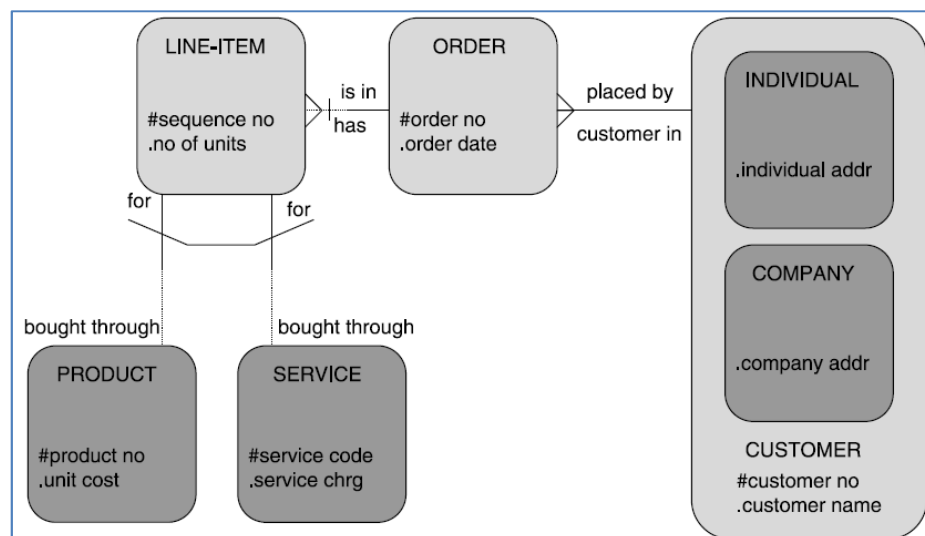


Figure 33: Barker's Notation – An example model (Ponniah, 2007)

Fact – Oriented Modeling Approach

Except from the Entity – Relationship Modeling Approach that described in the previous paragraphs, data modelers may adopt another approach to data modeling for designing a database. This is a **Fact – Oriented Modeling Approach** that began during the 1970s by describing the information domain in terms of objects playing roles (Ponniah, 2007) and the attributes and relationships as elementary facts (Halpin T. , 1991). In fact this approach enables data modelers to model, transform and query information in terms of the underlying facts of interest and has being designed to promote correctness, clarity and adaptability to Information Modeling and Information Systems Engineering (Halpin T. , 2007).

Object Role Modeling (ORM) is a technique of a Fact – Oriented Modeling Approach (Halpin, 1991; Halpin & Orlowska, 1992) that was formulated in 1989 by Terry Halpin in his PhD thesis (Halpin T. , 1989). A brief historical review about the methods, techniques and theories that lead to Object – Role Modeling can be found in chapter 3 of the book (Halpin T. , 2001), in (Halpin T. , 2006), in (Halpin T. , 2007) and in chapter 3 of the book (Halpin & Morgan, 2008).

An Initial version of Object – Role Modeling can be found in (Nijssen & Leunc, 1988; Wintraecken, 1990; Rasdorf & Abudayyeh, 1992; Darke & Shanks, 1995) with the name **NIAM (Natural Information Analysis Method)**. This method was then extended into the version of the **Predicator Model (PM)** (Bommel, Hofstede, & Weide, 1993) which became the **Predicator Set Model (PSM)** (Hofstede & Weide, 1993). Also some others versions of ORM are the **MOON (Normalized Object – Oriented Method)** and the **NORM (Natural Object Relationship Model)** (Halpin & Proper, 1995). One more initial version of ORM can be found in (Shoval & Zohn, 1991) known as **BRM (Binary Relationship Modeling)** (Halpin T. , 1995-A). A new version of ORM is the **Formal ORM (FORM)** which is described in the book (Halpin T. , 1995-A) and is supported by the software of Microsoft Visio for Enterprise Architects (VEA), which is a part of Visual Studio.NET Enterprise Architect. Finally the most recent version of ORM is that of **ORM2** (Halpin T. , 2005-A), which is supported by the NORMA tool (Curland & Halpin, 2010).

An example of ORM is shown in Figure 34. A more detailed description, understanding and graphical syntax or notation can be found in the books (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008). Also a number of technical papers and articles on ORM, as well as a comparison of ORM with other approaches can be found in (Halpin T. , 2015).

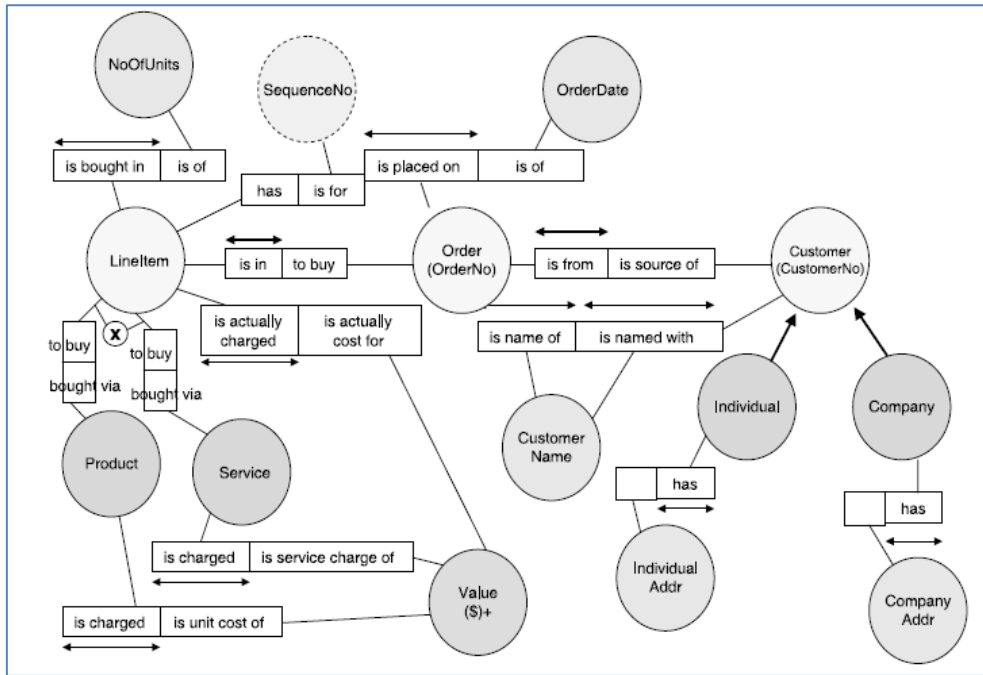


Figure 34: ORM – An example model (Ponniah, 2007)

The Object Role Modeling and more specific his version of Natural Information Analysis Method (NIAM) became the basis for another Fact – Oriented Modeling technique. This is the **Fully Communication Oriented Information Modeling (FCO-IM)** (Bakema, Zwart, & Lek, 2002). The **Fully Communication Oriented Information Modeling (FCO-IM)** is a technique for building conceptual information models that can be automatically transformed into ERM, UML, Relational or Dimensional Models (FCO-IM: Fully Communication Oriented Information Modeling, 2015). This technique uses diagrams that called **Information Grammar Diagrams (IGDs)** that show fact types, label types, object types, fact type expressions and object type expressions in their mutual relationships (Bakema, Zwart, & Lek, 2002). A detailed description, understanding, examples and graphical notation of FCO-IM can be found in the book (Bakema, Zwart, & Lek, 2002). Indicatively some graphical symbols are given in Figure 35. Also a chronological overview of publishing book, articles and papers on FCO-IM can be found in the (FCO-IM: Fully Communication Oriented Information Modeling, 2015). Finally FCO-IM supported by the **CaseTalk** and **Infagon** software tools (FCO-IM: Fully Communication Oriented Information Modeling, 2015). An example of FCO-IM is shown in Figure 36.

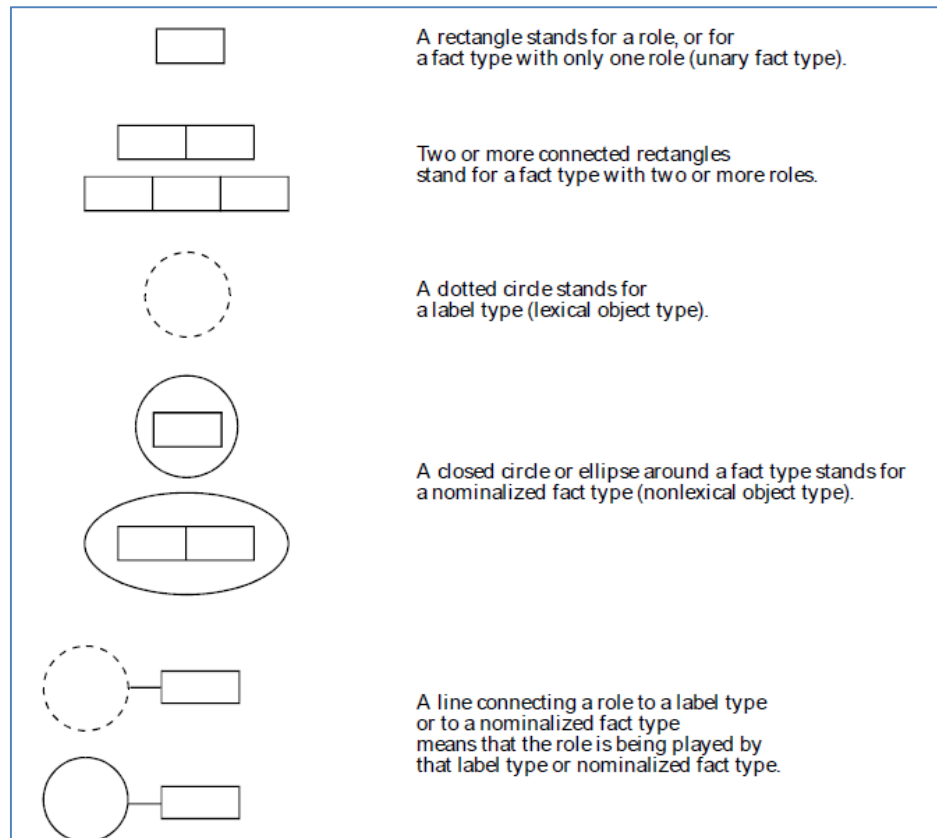


Figure 35: Graphic Symbols of FCO-IM (Bakema, Zwart, & Lek, 2002)

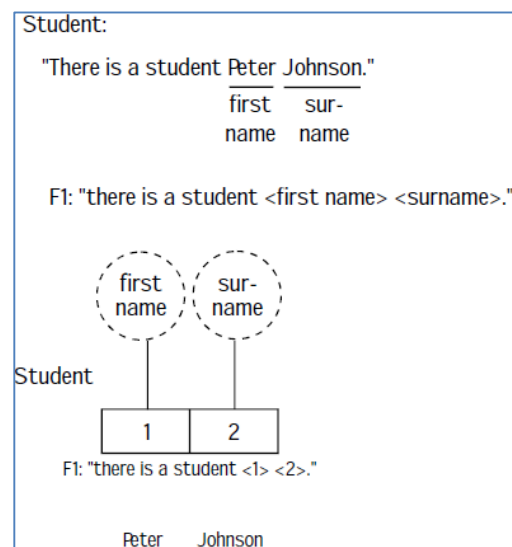


Figure 36: An example of FCO-IM (Bakema, Zwart, & Lek, 2002)

Object – Oriented Modeling Approach

As we referred at the beginning of this section the last most popular approach to data modeling for designing a database is that of **Object – Oriented Approach**. The Object – Oriented Approach focuses on building a model around objects ([Hoffer, Prescott, & McFadden, 2007](#)). More especially the building blocks of that model are object classes, attributes, operation and associations (relationships) ([Lee, 1999](#)).

The main standard that is used for designing a database in this approach is the **Unified Modeling Language (UML)**. The development of UML started in 1994 by Grady Booch and Jim Rumbaugh, in their effort on unifying the Booch and OMT (Object Modeling Technique) methods, and continued in 1995 with her unification in the Object – Oriented Software Method (OOSE) of Ivan Jacobson ([UML Summary: Version 1.1, 1997](#)). Then in 1977 UML was adopted by the Object Management Group (OMG) for object – oriented analysis and design ([Rumbaugh, Jacobson, & Booch, 1999](#)). A more detailed description about the creation of UML can be found in ([Rumbaugh, Jacobson, & Booch, 1999](#)).

According to ([Connolly & Begg, 2005](#); [Elmasri & Navathe, 2004](#)) UML uses a graphical representation and defines a number of diagrams which can be categorized as:

- *Structural Diagrams* which describe the static relationships between components. These include:
 - *Class Diagrams*: They show classes, interfaces, collaborations, dependencies, generalizations, associations and other relationships
 - *Object Diagrams*: They used to test Class Diagrams for accuracy and they show a set of individual objects and their relationships.
 - *Component Diagrams*: They show the organizations and dependencies among software components.
 - *Deployment Diagrams*: They represent the distribution of components (tables, files, libraries, executables) across the hardware topology.
- *Behavioral Diagrams*, which describe the dynamic relationship between Components. These include:
 - *Use Case Diagrams*: They show the functional interactions between users and the system.
 - *Sequence Diagrams*: They show the interactions between various objects over time.
 - *Collaboration Diagrams*: They represent interactions among objects as a series of sequenced messages.

- *Statechart Diagrams*: They describe how an object's state changes in response to external events.
- *Activity Diagrams*: They present a dynamic view of the system by modeling the flow of control from activity to activity.

Elmasri & Navathe (2004) refer that from the above diagrams **the Class Diagrams are the most useful for modeling a conceptual database schema**. An example of UML Class Diagram is shown in Figure 37. A detailed description, understanding, examples and graphical notation of UML Class Diagrams can be found in several books such as (Fowler & Scott, 1999; Elmasri & Navathe, 2004; Connolly & Begg, 2005; Sharron & Evan, 2005; Teorey, Lightstone & Nadeau, 2006). Also in (Object Management Group, 2015) we can find numerous publications and reports on UML different versions.

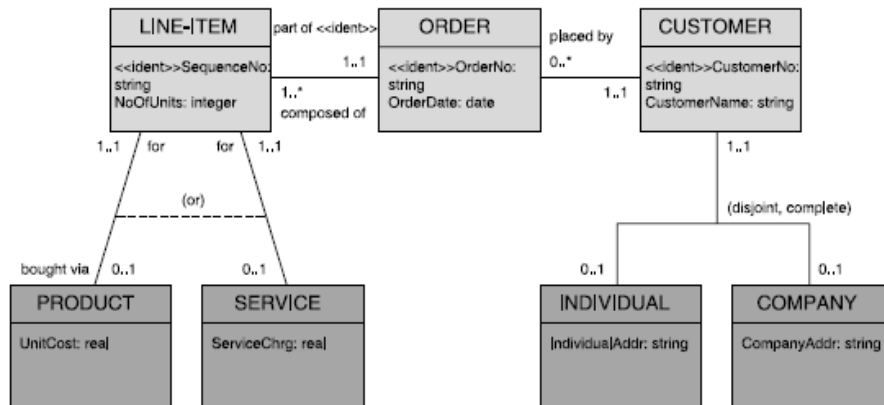


Figure 37: UML Class Diagram (Ponniah, 2007)

From the above data modeling techniques that described, **we propose Object – Role Modeling (ORM) as the most suitable for designing a database for Business Capability**. That's because Object – Role modeling is the most versatile and the most descriptive technique than Entity Relationship (ER) Model, IE, Barker's Notation, IDEF1X and Class Diagrams of UML (Hay, 1999) and is clear and detailed enough to capture the Business Complexity than other data modeling methods (Halpin T. , 1996). In ORM a validated conceptual schema can be easily then mapped to logical/physical/external schema either automated or manually (Cuyler & Halpin, 2003). Also ORM is more stable and provide validation than the others, since it is attribute free because all facts are represented in terms of objects playing roles (Cuyler & Halpin, 2003). In ORM we can find association of any arity (unary, binary, ternary), instead of Entity Relational Approaches (e.g ER) that allows binary associations and Object – Oriented Approaches (e.g UML) that has no unary associations (Cuyler & Halpin, 2003). Also constraints in ORM than the other techniques, methodologies

and languages work properly with n-ary associations (Cuyler & Halpin, 2003). ORM uses a natural language and its schemas can be represented in either diagrammatic or textual form, which mean that can be easily understood and validated by experts (Cuyler & Halpin, 2003). Finally ORM models can be manually or automatically be transformed to the other modeling techniques, methodologies and languages (Cuyler & Halpin, 2003).

2.3 Discussion – Result of Research

We have presented a brief literature review about Capability. We have conducted a thoughtful research in bibliography in order to understand what the Capability of an organization is in general, why is it important for an organization to focus in Business Capability, why Business Capability must be used for the software development in digital enterprises of tomorrow, and how.

The Capability notion began from the Management Science in the field of Corporate Strategy taking the form of managerial and functional capabilities (Long & Vickers-Koch, 1995). Then Capabilities became the main component in the field of Strategic Management in order an organization to achieve and maintain a competitive advance (Wernerfelt, 1984; Phahalad & Hamel, 1990; Barney, 1991; Barton, 1992; Stalk, Evans, & Shulman, 1992; Long & Vickers-Koch, 1995; Teece et all, 1997; Tallman & Fladmoe-Lindquist, 2002;). In this field two main strategies took place during the time the resource – based view of a firm (Wernerfelt, 1984; Barney, 1991) where capabilities or resources classified as physical capital resources, human capital resources and organizational capital resources, and the competence – based view of a firm (Phahalad & Hamel, 1990; Barton, 1992) an extension of the previous where capabilities referred as Core Capabilities with four dimension, which was a) skills and knowledge base, b) values and norms, c) managerial systems and d) technical systems. In the meantime Capabilities – based Competition (Stalk, Evans, & Shulman, 1992) took place in the Corporate Strategy which then led to Capabilities – based Organizations (Long & Vickers-Koch, 1995). The rapidly changing environments led to the extension of capabilities into that of Dynamic Capabilities (Teece, Pisano, & Shuen, 1997) and the Internationalization and Globalization led to a Capability – driven Strategy Framework (Tallman & Fladmoe-Lindquist, 2002). Finally a Capability – based Modeling Paradigm (Beimborn, Martin, & Holman, 2005) introduced for representing business functions and processes, in order to empowered managers to make decisions, by adopting a Capability map concept. From the previous researches it is understandable that capabilities played an important role in Management Science in order an organization to gain a competitive advance. The evolution of Capabilities

into global strategies and into a notion for describing an organization reinforces its importance.

In the meantime by adopting the Capability map concept capabilities start becoming a concern not only in Strategic Management and Corporate Strategy theories but in some areas in Computer Science. More especially enterprise and system architects start talking about Business Capability *as the particular ability or capacity that business may possess or exchange to achieve a specific purpose or outcome* (Holman, 2006) with basic components: Process, People and Physical Assets (Greski, 2009-B). Also Business Capability or a simple capability defines what a business does; it does not communicate or expose where, why or how something is done – only what is done (Ulrich & Rosen, 2011). Thus Business Capability has being used during the alignment between Business and Information Technology (Scott, 2009; Rosen, Business Processes Start with Capabilities, 2010; Barroero, Motta, & Pign, 2010; Freitag, Matthes, Schu, & Nowobilska, 2011; Ulrich & Rosen, 2011; Stirna, Grabis, Henkel, & Zdravkovic, 2012), as a centric idea and core component of Enterprise Architecture (Barroero, Motta, & Pign, 2010; Freitag, Matthes, Schu, & Nowobilska, 2011; Ulrich & Rosen, 2011) and in the moving from legacy applications to SOA – based Architectures (Frey, Hentrich, & Zdun, 2013). Also Business Capability leads to competitive advance (Brits, Botha, & Herselman, 2007; Bakhtiyari & Adel, 2012) and provide business value (Scott, 2009; Greski, 2009-B).

According to the previous different approaches for modeling Business Capabilities took place in the areas of Enterprise Architecture, Service Oriented Architecture and in alignment between Business and IT. The most known were the hierarchical description by a Capability map (Holman, 2006), a matrix analysis with a feedback loop (Brits, Botha, & Herselman, 2007), this of “Heat Mapping”, “Footprinting” and “Mix the Models” (Keller, 2009), the Capability Dependency Analysis Method (Freitag, Matthes, Schu, & Nowobilska, 2011) and the Capability Mapping Framework (Ulrich & Rosen, 2011). From all this approaches the most complete is that of (Ulrich & Rosen, 2011), since they provide a clear decomposition of Capability map hierarchy and a clear picture of the role of Business Capabilities in Enterprise Architecture. By this way they achieve the alignment between Business and IT, and the information about what a Business does is more specific.

In the meantime the extensive use of Internet and its variability led to the development of Capability Driven Development (CDD) method (Stirna, Grabis, Henkel, & Zdravkovic, 2012) that integrates organizational with IS development taking into account changes in the application context of a solution and also uses a Meta – model of Capability

with three sections: Enterprise and Capability modeling, the Capability Delivery Context modeling and the Capability Delivery Patterns. This method has been used as a centric idea for the development of software for the digital enterprises of tomorrow by European Commission, who announced the project “Capability as Service in digital enterprises - CaaS” ([European Commission, 2013](#)). Since now several researchers focused in giving the essential tools, methodologies and empirical experience for the application of CDD approach according to the CaaS Project ([Zdravkovic et al, 2013](#); [Espana et al](#); [Bravos, Loucopoulos, Stratigaki & Vavlis, 2014](#); [Bravos, Gonzslez, Grabis, Henkel & Jokster, 2014](#); [Bravos, Grabis, Henkel, Jokste & Kampars, 2014](#); [Stratigaki et al, 2014](#); [Berzisa et al , 2015](#)). These researches worked in defining the CaaS base methodology ([Berzisa, et al., 2013](#)), in providing a meta-model for capability design and delivery ([Zdravkovic, Pastor, & Loucopoulos, 2014](#)), in providing a deliverable of Capability models for Business Compliance Controlling and Auditing ([Loucopoulos, Bravos, Stratigaki, & Vavlis, 2013](#)), in providing a capability modeling within a service oriented architecture development project ([Espana, Gonzalez, Grabis, Jokste, Juanes, & Valverde, 2014](#)), in investigating the utility of modeling Business Capabilities by using design rational ([Bravos, Loucopoulos, Stratigaki, & Valvis, 2014](#)), in discussing the initial Capability modeling experiences with main emphasis on the capability design phase of CDD approach ([Bravos, Gonzslez, Grabis, Henkel, & Jokste, 2014](#)), in defining a set of concrete key goals to be achieved in order to have a methodological support (a goal graph) to CDD approach ([Bravos, Grabis, Henkel, Jokste, & Kampars, 2014](#)), in designing a meta-model as a foundation for compliance capability ([Stratigaki, Loucopoulos, & Nikolaidou, 2014](#)) and in providing a final CDD approach ([Berzisa, et al., 2015](#)). However the CDD method lacks from empirical experience with practical examples and none of the previous researches focuses in describing the Business Capability in a database in order this kind of information to be stored and classified and thus to be possible for all kind of analysis.

Thus in this dissertation we will respond to the question of how Capability must be used for the development of software for digital enterprises of tomorrow and so the purpose of this is the development of a Database Management System for Business Capability in a case study from the maritime domain field. By creating this Database Management System we intend to help modern organizations to gain a competitive advance and thereafter to achieve growth. That’s because when the information about Business Capability are stored, related and viewed in a database, managers have an overall view of what the organization does and thereafter can increase control, achieve better planning and taking decisions more efficient.

However in order to design this Database Management System we may use different data modeling approaches with the most popular the Entity – Relationship Modeling Approach, the Fact – Oriented Modeling Approach and the Object – Oriented Modeling Approach and each of them has specific methodologies or techniques or languages. In the Entity – Relationship Modeling Approach we can find ([Bachman, 1969](#); [Chen, 1976](#); [National Institute of Standards & Technology, 1993](#); [Finkelstein, 2006](#); [Mamayev, 2013](#)), in the Fact – Oriented Modeling Approach we can find ([Halpin T., 1995-A](#); [Bakema, Zwart, & Lek, 2002](#)), and in the Object – Oriented Modeling Approach we can find ([Rumbaugh, Jacobson, & Booch, 1999](#)).

The DBMS that we will create deals with data and information that describes processes, services, policies, goals, resources etc. This means that we need to use a more versatile, descriptive enough and detailed methodology or language to capture this Business Complexity. Object Role Modeling meets the previous criteria in comparison with other data modeling techniques, methodologies and languages ([Hay, 1999](#)). In ORM a validated conceptual schema can be easily then mapped to logical/physical/external schema either automated or manually. Also ORM is more stable and provide validation than the others, since it is attribute free because all facts are represented in terms of objects playing roles. In ORM we can find association of any arity (unary, binary, ternary), instead of Entity Relational Approaches (e.g ER) that allows binary associations and Object – Oriented Approaches (e.g UML) that has no unary associations. Also constraints in ORM than the other techniques, methodologies and languages work properly with n-ary associations. ORM uses a natural language and its schemas can be represented in either diagrammatic or textual form, which mean that can be easily understood and validated by experts. Finally ORM models can be manually or automatically be transformed to the other modeling techniques, methodologies and languages ([Cuyler & Halpin, 2003](#)).

CHAPTER 3: Object Role Modeling (ORM)

Structure of this Chapter

- 3.1 Background to Meta – Modeling
- 3.2 The Conceptual Schema Design Procedure
- 3.3 The Relational Mapping Procedure
 - 3.3.1 Definitions – Notations
 - 3.3.2 Rules & Strategies of Mapping
 - 3.3.3 Main Steps of Mapping
- 3.4 Chapter Summary

This chapter deals with Object Role Modeling (ORM). Since ORM is a technique that is used for creating specific data models in different levels of abstraction and also in those levels uses a specific syntax, semantic and notation, this Chapter in Section 3.1 discusses a background to Meta-Modeling. Then in Section 3.2 the first level of data abstraction in ORM is presented, meaning the Conceptual Schema Design Procedure (CSDP). In more detail this section discusses the graphical notation is used in this level and the main steps of the CSDP. The Section 3.3 deals with the second level of data abstraction in ORM and more specific with the Relational Mapping Procedure. Here it discusses the definitions & the notation is used, the rules & Strategies of mapping and the main steps of mapping. Finally in Section 3.4 a summary of the chapter is presented.

3.1 Background to Meta – Modeling

As we referred in a previous chapter during the designing stages of a DBMS a conceptual modeling practice is used *for creating specific data models* in four levels of abstraction: conceptual design, logical design and external physical design.

According to [Leppanen \(2006\)](#) a model can be defined as *a think that is used to help or enable the understanding, communication, analysis design and implementation of some other thinks (Teleological Viewpoint)” or a perception and an abstraction of relevant thinks in reality (Semantic Viewpoint)” or in one of three forms, namely as a conceptual construct, as a linguistic expression, or as physical construct (Semiotic Viewpoint)”*.

Also a model can be described by a modeling language, which itself is described by a syntax, semantics and notation ([Karagiannis & Kühn, 2002](#)). According to [Karagiannis & Kuhn \(2002\)](#):

- The **syntax** is described by a grammar and deals with the elements and rules for creating models in forms of graph grammars and meta-models. The syntax is consisting of two parts ([Leppanen, 2006](#)) **abstract syntax** (which leaves out the representational details) and **concrete syntax** (which gives notational elements, called the symbols in the vocabulary of a language, and rules for connecting them with one another and with the concepts).
- The **semantics** describes the meaning of a modeling language and consists of a semantic domain (it describes the meaning by using ontologies, mathematical expressions etc) and the semantic mapping (the procedure for connecting the syntactical constructs with their meaning defined in the semantic domain).
- The **notation** describes the visualization of a modeling language (e.g. symbols for visualizing the syntactical constructs).

Thus to be able to manipulate models, their language needs to be specified as model of these models: Meta – models ([Sprinkle, Rumpe, & Vangheluwe, 2010](#)). A **Meta – model** is a model about the information being expressed during the modeling procedure and basically a Meta model is a model of model ([Geisler, Klar, & Pons, 1998](#)). Thus Meta-modeling is the procedure of modeling models and therefore a Meta – model describes the syntax of models and also helps to define semantics ([Sprinkle, Rumpe, & Vangheluwe, 2010](#)). A more clear definition about Meta – model is given in [Figure 38](#). According to that a model is a simplified representation of reality, while the Meta – model defines a modeling language in which a model can be expressed. This model level hierarchy is according to the Meta Object Facility (MOF) meta – modeling standard of Object Management Group (OMG) which is based in

UML. According to (Hinkelmann, 2011) the **M0 level describes the basic data** while **M1 model level describes the meta – data**, meaning the schemas and interfaces for describing the structure of the data. The **M2 level is the meta – model or the language for specifying the concepts of the modeling language**. Finally the **M3 level is the MOF specification itself** which allows us to draw the model (e.g. boxes, arrows etc).

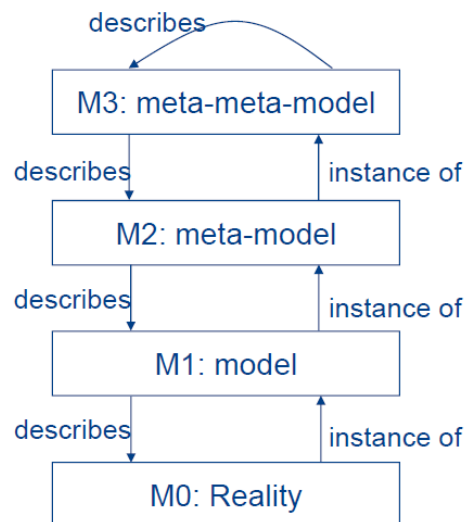


Figure 38: The four levels hierarchy of a model (Hinkelmann, 2015)

Although for other modeling languages there are official standards for Meta – modeling like the previous we describe, according to (Cuyler & Halpin, *Metamodels for Object-Role Modeling*, 2003) for Object Role Modeling (ORM) there is no official standard for meta – model. Thus Cuyler & Halpin (2003) in their work tried to pave a way for a standard ORM Meta – model, in which they discuss how ORM components (object types, roles, predicates) may be metamodeled in ORM including component reuse, examines ways to metamodel business rules in ORM and metamodels instance data in ORM. Also they referred Meta – modeling in ORM is important because:

- It is the better way for conceptual information analysis than ER and UML.
- It is a possible standard for business rules expression and for use in ontology standards.
- It would facilitate the interchange of ORM model data between software tools.
- It aims to specify the grammar of a syntactically valid ORM model, which means that a Meta – model allows models to be checked for syntactic correctness.

3.2 The Conceptual Schema Design Procedure





As we have already discussed in section 1.3 for creating a DBMS according to ORM, it is necessary to working with data for four different levels. The first level of this procedure is the conceptual level, which will be discussed in more detail in this section.

In this level the Universe of Discourse of the application, is described in human concepts and depicted in an **Conceptual Schema**. This schema describes the structure or grammar of the Universe of Discourse, which refers to object types, the roles between objects types and constraints. Also it specifies the structure for all the permitted states and transitions of the conceptual database (Halpin T., 2001).

When designing this schema a modeler may use a specific graphical notation as set out in [Appendix: Table 2](#) (Halpin, 2010-B).

In more detail taking into account (Halpin T., 2001; Halpin T., 2005-B; Halpin T., 2005-A) we could say that:

- **A fact type** consists of relevant object types, associated with roles, with the predicates and the reference schema. Also a fact type can consist of one or more roles, meaning a unary fact type (that depicts one role), a binary fact type (that depicts two roles), a ternary fact type (that depicts three roles) etc. Actually a fact type is a candidate table in the physical database and by which we can derive in a second level of analysis (in relational schema procedure of ORM) all the information about the name of the table, the name of the columns, the primary keys, the foreign keys, the unique columns, the nullable columns and also all the other constraints that this table will have.
- **The object types** or entity types are depicted as a named soft rectangle rounded corners, while **the value types** as ellipse soft rectangle rounded corners.
- **The roles played by entities in a fact type** are depicted as boxes connected to the object types by solid lines. In fact the roles represent the relationships between object types and they show the part played by the object types in this relationship. Also each role is associated with a corresponding column of the fact table.
- **Predicates** are depicted as text upon or under the boxes of roles. For binary fact types, a forward predicate reading is left-to right or top-to-bottom, and an inverse predicate reading is right-to-left or bottom-to-top. For a binary fact type forward and inverse reading may be displayed together, separated by a slash.
- **Objectified Associations** are depicted as a soft rectangle rounded corners, with the name of them out of the rectangle and included in "...".

- **Internal uniqueness constraints** are depicted with a line upon the box of roles, whereas a preferred uniqueness constraint is indicated by a double line (corresponding to one common practice of doubly underlining primary keys when alternate keys exist).
- **External uniqueness constraints** are depicted as circled underline for unique () and as circled double underline for primary (), meaning when the constraint provides the preferred identification scheme.
- Simple **Mandatory constraints** are depicted by a solid line, while disjunctive mandatory constraints (inclusive-or) by placing the solid dot in a cycle () connected by dotted line to the roles it applies to.
- **Subset, Exclusion and Equality Constraints** are depicted by cycle containing  and connected to the associated roles with dashes lines.
- **Frequency and Value Constraints** are enclosed in round brackets, with the value separated in comma and enclosed in '...'.
 - **Ring Constraints** are depicted with a cycle symbol (which differs according the type e.g. reflexive, asymmetric etc) connected with a dashed line with the role associated.
- A relation that deals with sets as a subset of another's is implemented by **Subtyping**. A Subtype is depicted as object type (parent) with an arrow pointing from it to its proper Supertype (child).
- **Textual Constraints** are depicted by a footnote number, with a textual reading of the constraint. This constraints may describe derivation rules.

In order to design a conceptual schema a specific procedure is followed by the modelers. This is the **CSDP procedure (Conceptual Schema Design Procedure)**. In fact this procedure focuses on analyzing and designing of data, and consisting of 7 steps. The first three of these steps deals with identifying the fact types, and the other four by adding constraints and derivation rules to the fact types ([Halpin T. , 1995-A](#)). Those steps are:

1. Transform familiar information examples into elementary facts, and apply quality checks.
2. Draw the fact types, and apply a population check.
3. Check for entity types that should be combined, and note any arithmetic derivations.
4. Add uniqueness constraints, and check arity of fact types.

5. Add mandatory role constraints, and check for logical derivations.
6. Add value, set comparison and subtyping constraints.
7. Add other constraints and perform final checks.

As referred by (Halpin T. , 2007; Halpin T., 2006) the first step is the most critical during this procedure, since examples of required data are verbalized in natural Language. In more detail in this step a modeler takes real use cases of data for this UoD by the organization and clarifies the meaning of their terms. Then he transforms the use cases by using a natural language in terms of elementary facts. Elementary facts assert that a particular object has a property or that one or more objects participate in relationship, where that relationship cannot be expressed as a conjunction of simpler facts without introduction object types. Then for this kind of elementary facts the way of reading it (reading from different directions) is specified. Then the modeler applies a specific quality check. First he insures that the objects are well defined, values are identified by constraints, and entities are real word object that are identified by a definite description. Finally a modeler use familiarity with the UoD to see if some facts should be split or recombined.

In second step of this procedure a modeler uses the notation of ORM that described previously, in order to draw a graph that represents the fact types. During this procedure he also applies a population check, by matching each fact column in this schema with the real data.

In third step the entity types that should be combined are checked and the notes for arithmetic derivations are added. In other words if a modeler has drawn two entity types which have a common instance, he must combine them. Also if the same kind of information is to be recorded by different entity types, he also must combine them. Finally he adds the appropriate derivation rules for the fact types that arithmetically are derivable from other fact types.

Finally all the other steps, meaning 4,5,6 & 7, are dealing with how a modeler may implements all the type of constrains that exists.

As we have already said, the result of the CDSP procedure is a model that will be uses as a background for creating a second model, a Relational Schema. Thus the procedure for that is described in more detail in the next section.

3.3 The Relational Mapping Procedure

In this section we will discuss the relational mapping procedure (Rmap procedure), that refers to the procedure for mapping the ORM meta-model of Business Capability onto a relational schema. More generally we will present the procedure followed, in order to design the logical data model for our Database Management System.

In order to achieve this, firstly there is a need to identify the notion of “relational schema” and then to give the necessary generic notation and terminology that is used due to designing stages of this schema.

Secondly we discuss about the rules of mapping from the ORM meta-model to logical data model, and finally we conclude with the results of mapping, that concerns the constitution of the logical schema.

3.3.1 Definitions – Notation

According to Halpin (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008) a **relational schema** (or relational database schema) is a set of relational table definitions, constraints and perhaps derivation rules.

In order to depict a relational schema, we will use the Formal Object – Role Modeling (FORM) methodology, that extent and refines an older mapping procedure known as the **ONF (Optimal Normal Form) algorithm**. Following this approach a relational schema appears as a schematic form, consists of relational tables each row of them expresses one or more elementary facts. The structures of those tables are called “**table schemes**”. Basically those tables are names of columns (attributes), which draw their values from domains (Halpin, 1995-A; Halpin 2001; Halpin & Morgan, 2008).

There is a use of two main layouts for table schemes. The first one is the **horizontal layout** (abstract), where the table name precedes a parenthesize list of columns separate by commas (Halpin T. , 2001). The second one is the **vertical layout** (Visio-like), where the tables depicts diagrammatically and supplemented by textual rules stored in property sheets of code (Halpin 2001; Halpin & Morgan, 2008). Examples of these layouts are shown in Figure 39. However for the purpose of this dissertation we use the horizontal layout.

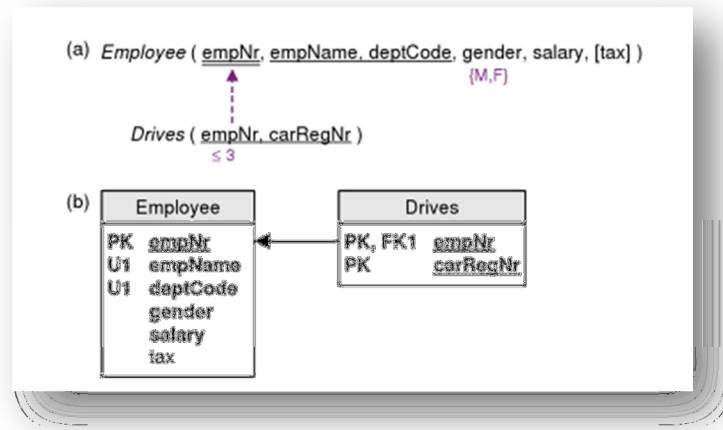


Figure 39: Examples of relational schema in (a) horizontal layout and (b) vertical layout (Halpin & Morgan, 2008)

Before we design the horizontal layout, it is necessary to take into consideration the notation used for specifying names of tables and columns, constraints and derivation rules. Particularly according to Halpin (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008) the main notation we use is:

- **The names of tables are written in *italics*** and starting with capital letters. Also all tables must have meaningful and different names.
- **The names of columns** must be meaningful and different in every table, and **are represented with lower-case letters**, parenthesized and separated by commas. If we desire to display the domain names inside the parenthesis, then we use at starting capital letters and a colon separator after the column names.
- A column that allows null values is said to be **optional**. Optional columns are enclosed in *square brackets* ([]).
- A column that does not allow null values is said to be **mandatory**. A column is mandatory unless it is marked optional.
- The **uniqueness constraints** (internal or external) on relational columns are shown by **underlining**.
- Each unique column or unique column combination provides a **candidate key** for identifying rows in the table. In other words a key is minimal set or uniquely constrained attributes. A **primary key is doubly underlined** if an alternate key or secondary key exists, but if there is only one key, this is automatically the primary key.

- Moreover if columns in a composite key are not listed consecutively, **arrowheads (\leftarrow, \rightarrow)** must be added to the underlines, to show that a single composite uniqueness constraint applies, rather than a multiple single constraint.
- A **referential integrity constraint** between two tables is depicted as a **dotted arrow ($\text{----}\rightarrow$)**. We use this jargon and notation in order to express different facts about the same object, which have to be stored in different tables and referenced each other. The name of the column (attribute) in the first table where the dotted arrow starts is **a foreign key**, that reference to the name of the column on the second table. The last one is the primary key of the second table.
- Relevant a **referential equality constraint** is depicted as a **double dotted arrow ($\text{<----}\rightarrow$)**.
- A **value constraint** is depicted using braces ($\{\}$), separated by commas and quotes. We usually place them up to the name of the column.
- A **frequency constraint** is depicted place the number of frequency down of the column's name.
- All the other types of constrains (**external uniqueness constraints, ring constraints, subset constraints, join subset constraints, exclusion constraints, exclusive-or constraints, inclusive-or constraints, equality constraints etc.**), are depicted as a **dashed or dotted lines** between the attributes, enclosed by the symbols that represent every kind of constraint we want to include.
- Finally **derivation rules** are specified in an appropriate language, and depicted using numeric superscripts on the names of columns that reference to a footnote in the end of the relational scheme. These rules provide a list of functions, operators and rules that may be used to derive information. These may involve mathematical calculations and logical inferences. Any derived fact type should be included on the schema and **marked “*S” with the derivation rule also declared**. Then we map the derivation rule in the table marked “*”.

3.3.2 Rules & Strategies of Mapping

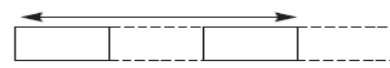
So far, we introduce some basic definitions and notations are used during the modeling stages of the relational schema. This section deals with **the basic rules of mapping**

a conceptual schema onto a relational schema. More generally we will see how to group fact types into table schemes and how to map constraints and rules in more detail.

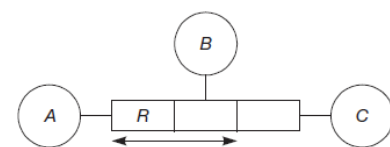
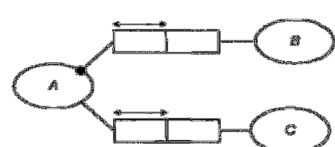
According to Halpin (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008) the main criteria must meet a relational schema should be **correctness, efficiency and clarity**. As he explains a relational schema must be **equivalent to the conceptual schema**, must have **good response times to updates and queries** (with reasonable demands on storage space) and it should **be relatively easy to understand and work with**. He also explains that correctness of data is more important and the only way ensures this is **by avoiding redundancy**. Although the last one may lead to more tables in design, which can slow down queries and updates by the requirement of extra table joins, he refers that it is important **to keep the number of tables down to an acceptable limit**. That's why he insures that the Rmap procedure guarantees a redundancy-free relational design and includes strategies to restrict the number of the tables.

In order to avoid redundancy in tables, Halpin reported in (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008) that we must ensure that each fact type maps to only one table, in such a way that its instances appears only once. To achieve this, he introduces **two basic rules to group fact types into table schemes** as follows:

Rule 1: Fact types with compound uniqueness constraints

 map to separate tables. That means every predicate other than an objectified, which **has a uniqueness constraint spanning two or more of its roles**, meaning m:n binaries and all n-aries ($n \geq 3$), **must map to a separate table**. If there is only one uniqueness constraint on the predicate the primary key of the table is based on this; otherwise one is picked as a primary.

Rule 2: Fact types with functional roles attached to the same object type

 or  are grouped into the same table, keyed on the object type's identifier as primary key. Examples of tables for the first and the second picture are: R (a, b, c) and R (a, b, [c]).

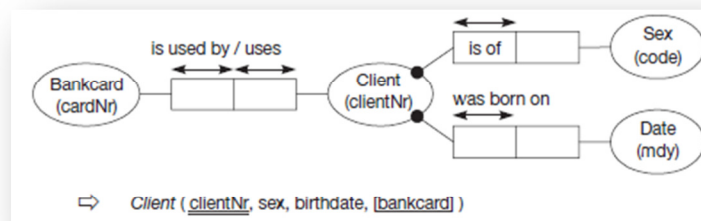
Except the main rules for grouping fact types, he mentions that in a relational model there are **two basic integrity rules**. The first one is the **entity integrity rule** who demands a primary key to contain no null values, which means that all its constitutive columns are mandatory. The second one is the **referential integrity rule** who basically says that every non-null value of a foreign key must match the value of some primary key (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008 ; Montali, 2011-2012).

Thereafter Halpin (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008) **uses particular strategies for grouping fact types into tables** concerning 1:1 associations, external uniqueness constraints, nested object types, independent (lazy) object types and subtypes.

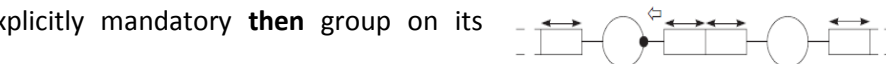
In the case of 1:1 associations it is important to determine in which side (table) this association will be grouped. The choice of grouping depends on criteria whether exists symmetric (both fact types optional or mandatory) or asymmetric situations (one fact type optional and the other mandatory and vice versa) and whether the roles are functional or non-functional. According to those the basic strategy for grouping 1:1 associations is (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008):

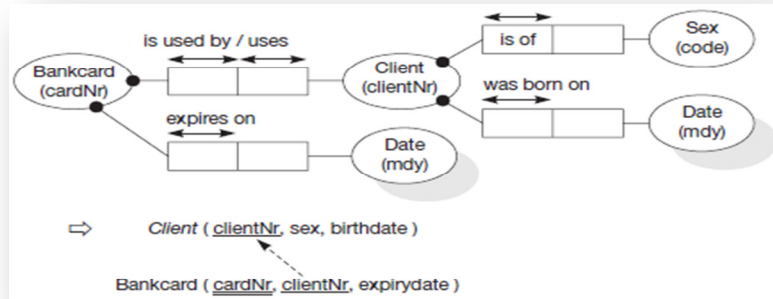
- **If only one object type in the 1:1 predicate has another functional role then group on its side.**

For example:

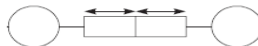


- **Else if both object types have other functional roles and only one role in the 1:1 is explicitly mandatory then group on its side.**

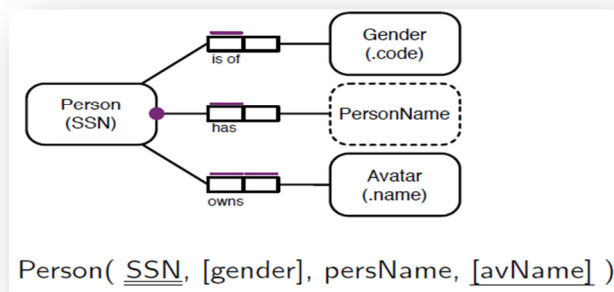




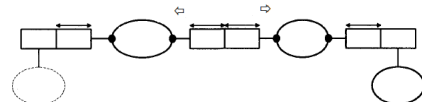
- **Else if** no object type has another functional role, **then** map the 1:1 to a separate table.



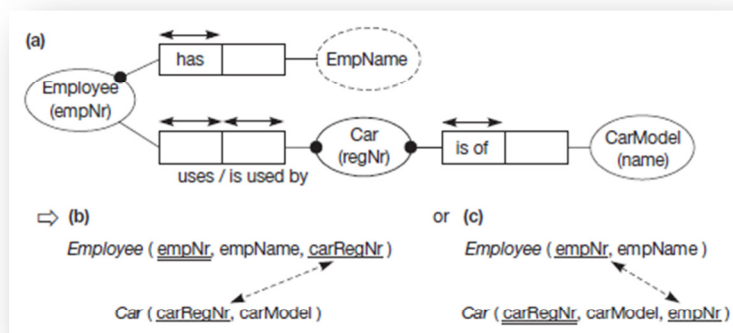
For example:



- **Else** the grouping choice is completely delegated to the modeler. This refers to cases whether both roles of the 1:1 fact types are mandatory or optional (symmetric situations), and each of them is attached to an entity type with another functional role. In such cases we can group on either side or we can try a simple table approach.



For example:

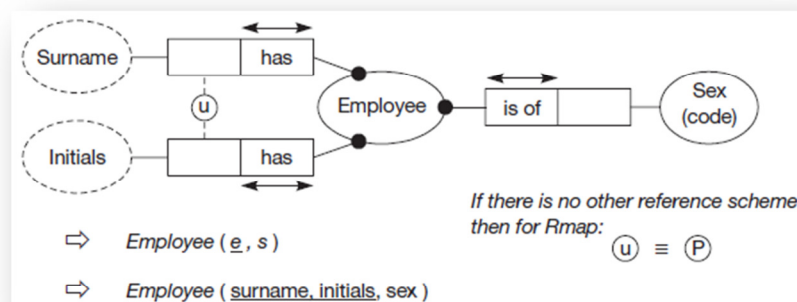


In the case of **external uniqueness constrains the standard mapping strategy is used** to visualize mapping is as follows (Halpin, 1995-A; Halpin, 2001; Halpin & Morgan, 2008; Montali, 2011-2012):

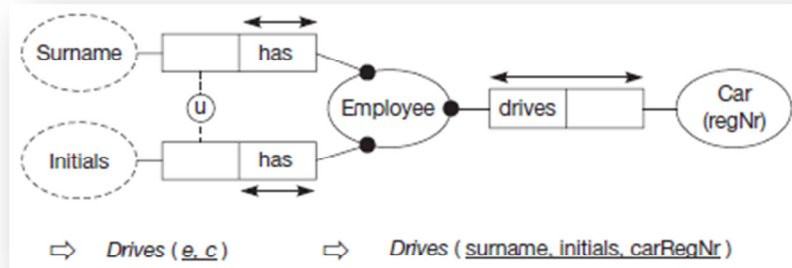
- Firstly, we do not consider the identification scheme of object type.
- Then, we group fact type into tables, using compact surrogates (e.g [e,s]) as columns.
- Finally, we restore the full tables replacing surrogates with the attributes used for preferred identification.

Here we may have four situations in cases like this as follows:

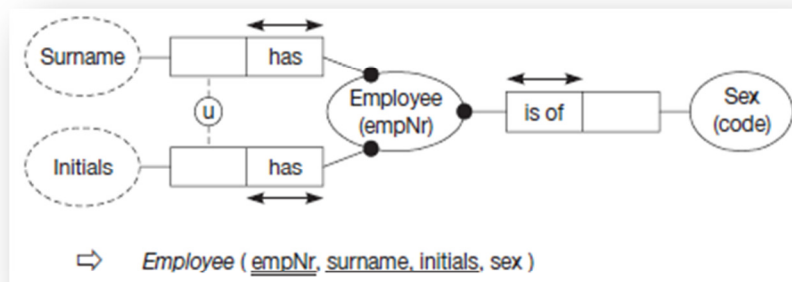
1. Firstly a functional fact type with a composite primary identifier. This means that an external uniqueness constraint is the preferred identification scheme for an object type attached to other functional roles. The strategy we use is grouping on the object type using surrogates, without considering the predicates involved in the external uniqueness constraint. Then expand the primary key using the object types involved in the external uniqueness constraint. For example:



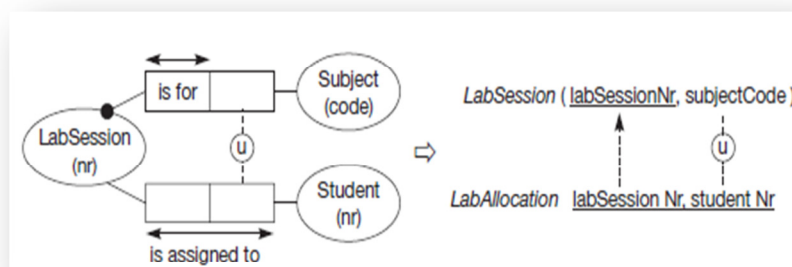
2. Secondly a non-functional fact type with a composite primary identifier. This means that an external uniqueness constraint is the preferred identification scheme for an object type attached to other non-functional roles. The strategy we use is mapping m:n association to a separate table, using surrogates. Then expand the primary key using the object types involved in the external uniqueness constraint. For example:



3. Thirdly a functional fact type with a composite secondary identifier. This means that the external uniqueness constraint is not the preferred identification scheme for an object type attached to other functional roles. The strategy we use is standard mapping and modeling the external uniqueness constraint as a key. For example:



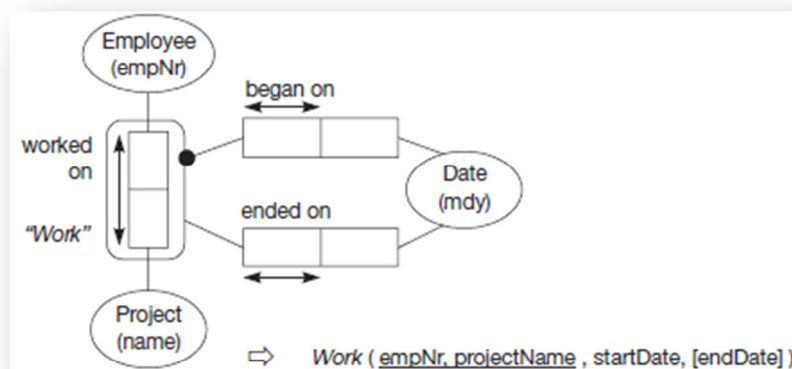
4. Finally a functional fact type with a composite secondary identifier. This means that an external uniqueness constraint is the preferred identification scheme for an object type attached to other non-functional roles. The strategy we use is standard mapping without considering the external uniqueness constraint. Then you add the external uniqueness constraint in the relational schema as an inter-table constraint. For example:



In the case of **nested (objectified) object types the strategy is used** for mapping is as follows (Halpin, 1995-A; Halpin, 2001; Montali, 2011-2012):

- Firstly, we do not consider the identification scheme of the objectified association.
- Then we consider the objectified association as a black box.
- Next we group fact types in the standard way.
- We unpack the black box into its component attributes.
- Finally we deal fine-grained constraints involving component roles of the objectified association if they exist.

For example:



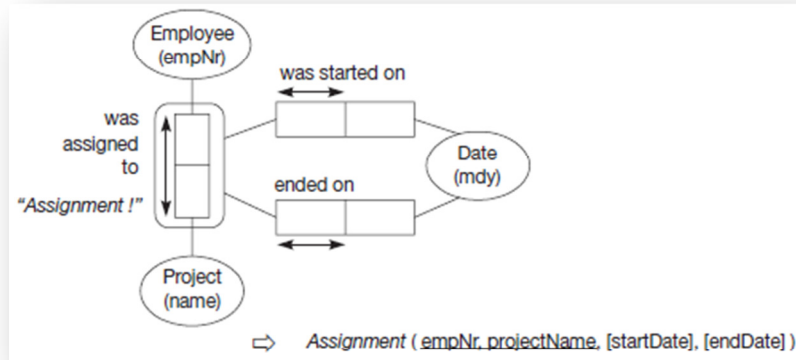
In this example the association "worked on" is objectified as "Work". So

- We do not consider the identification scheme and we initially treat the nested object type "Work" as a "black box": *Work* (■, startdate, [enddate]).
- We unpack the black box into its components attributes and giving: *Work* (empNr, projectName, startdate, [enddate]).
- The key is the combination of "empNr, projectName", because a uniqueness constraint is assumed to span any objectified predicate.

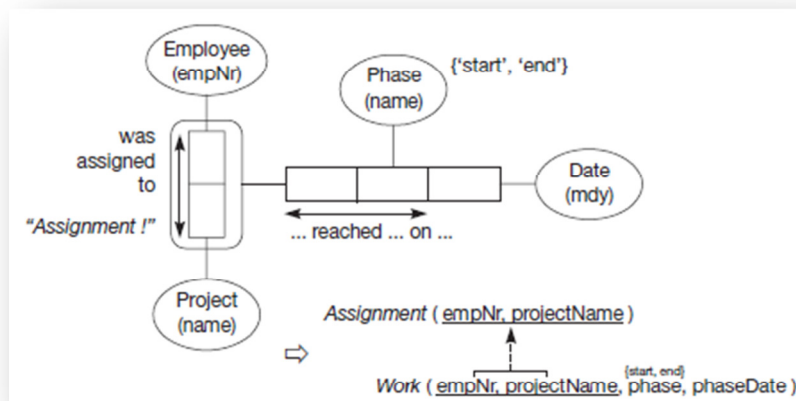
In the case of **independent object types, nested of not, the strategy is used** for mapping is as follows (Halpin, 1995-A; Montali, 2011-2012):

- If it has functional roles attached, then it must mapped to a separate table with its preferred identifier as primary key, together with all fact types in which it plays a functional role (if any) and all other attributes optional.

For example:

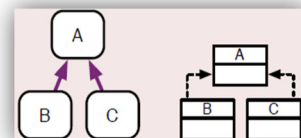


- If it has non-functional roles attached, then it must map to a table by itself and have a foreign key pointing in this table, using a subset constraint.




Finally **in the case of subtyping** there are three main ways in which the fact types may be grouped into tables. Those are *absorption*, *partition* and *separation* as follows. Recall that subtyping constraints are mutually **exclusive** (\otimes): there is a distinction between of them; collectively **exhaustive** (\odot): subtypes equals the union of the supertype; and **the combination of the previous two** (\oplus): where subtypes partition the supertype (Halpin, 1995-A; Halpin, 1995-B; Halpin, 2001; Montali, 2011-2012).

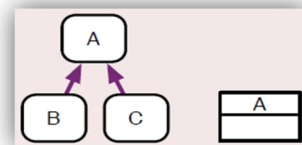
- **Absorption:** In this option subtypes are absorbed back into their top supertype before grouping. Then we group fact types as usual and add the subtyping constraints as textual qualifications. Especially a discriminator column reflects the presence of a classification type to distinguish supertypes, with a fixed domain, whose



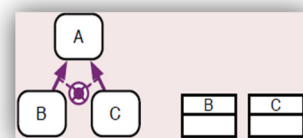
possible values correspond to different subtypes. All fact types attached to subtypes are moved to the supertype making the participation of the supertype optional. When such participation is mandatory, we add the textual constraints “exists only if” and “exist iff”, pointing to the discriminator column. The m:n fact types that involves subtypes are mapped to separate tables, with the foreign key pointing to the supertype table and combined with the “only where” constraint, pointing to the discriminator column. **We use this approach when all the roles played by supertype and subtypes are functional**, meaning they

have a simple uniqueness constraint: . **So any non-functional roles of the subtypes maps to separate table anyway.** The advantage of this approach is that it maps all the functional predicates of a subtype family into a simple table. This makes queries and updates more efficient. The main disadvantage is that generates null values. Also it is difficult to pose queries regarding only subtypes. Finally the functional table of the supertype is larger (more columns).

- **Separation:** In this option each object type is mapped to a separate table. The functional roles grouped directly to each object type. Then foreign keys are added from the subtypes to the supertype table, meaning that primary keys of each subtype refer to (FK) the primary key of the supertype. Also in this case we can use a discriminator column in the supertype, but this requires the presence of suitable constraints in the foreign keys. Depending on the existence of a mandatory role in the supertype, constraints used are the “exactly where” and the “only where”. The main advantage of this approach is that it minimizes nulls and queries about each subtype are fast. The main disadvantage is that queries are slower, because joins are needed. Also subtype constraints are specified as qualified subset constraints, so access to a supertable is required to enforce them. This cause slower insertions subtype tables.



- **Partition:** Here supertype is removed, replicating the attached information for each subtype. Especially roles attached to the



supertype are virtually replicated and pushed down to each subtype. Also we use unions to reconstructive the supertype. Then an exclusion constraint between the primary keys of the subtype tables ensures that each supertype individual is maintained only in one table. Actually ***this approach is used when subtypes form a partition of their supertype, which means that subtypes must be exclusive and exhaustive*** (Halpin, 1995-B). The exclusive means that they cannot have a common instance ($B \cap C = \{\}$) and the exhaustive means that they equals the union of the subtypes ($B \cup C = A$). However, If subtypes do not exhaust supertype, then is needed a separate table ($A - (B \cup C)$), which is not recommended (Montali, 2011-2012). The main advantage of this approach is that minimizes nulls and queries about all properties of the subtypes are fast. The main disadvantage is that union needed to for querying the superclass.

3.3.3 Main Steps of Mapping

Further to the two previous sections, we are now ready to discuss the basic steps we follow, in order to design a relational schema. This procedure has actually **six main steps** (Halpin, 1995-A; Halpin, 2001), which are the following:

1. **Step 0:** This is a preparatory step where we absorb subtypes into their supertype and erase mentally all explicit primary identification schemes, in which we treat compositely identified object types as “black boxes”. A compositely identified object type is either a nested object type or a co-referenced object type (one that is primarily identified with an external uniqueness constraint). For steps 1-3 we treat compositely identified object types just like any other object type. In more detail this step has **seven substeps** as follows:
 - 0.1 *Mentally binarize any unaries, and cater for any relative closure.*
 - 0.2 *Mentally erase all and treat compositely identified object types as “black boxes”.*
 - 0.3 *Indicate any non-absorption choices for subtypes.*
 - 0.4 *Identity any derived fact types that must be stored.*
 - 0.5 *Indicate mapping choices for symmetric 1:1 cases.*
 - 0.6 *Consider replacing any disjunctive reference scheme (meaning identification by a mandatory disjunction of two or more roles, at least*

one of which is optional), by using an artificial or concatenated identifier or mandatory defaults.

0.7 Indicate mapping choices where required for any objectified predicate without a spanning uniqueness constraint.

2. **Step 1:** in this step we follow the Rule 1 of grouping fact types into a table scheme. This is mapping each fact type with a compound uniqueness constraint to a separate table.
3. **Step 2:** Here firstly we follow the Rule 2 of grouping fact types into a table scheme. This is grouping into the same table, fact types with functional roles attached to the same object type, keying on the object type's identifier. Secondly we map 1:1 cases to a single table, generally favoring fewer nulls.
4. **Step 3:** In this step we map independent object types (lazy object types) with no functional roles to a separate table.
5. **Step 4:** Here we unpack each "black box column" into its component attributes.
6. **Step 5:** In this finally step we map all other constraints and derivation rules. Also subtype constraints on functional roles map to qualified optional columns, and on non-functional roles map to qualified subset constraints.

3.4 Chapter Summary

The main scope of this Chapter was to provide a part of theory for Object Role Modeling (ORM). Thus have already discussed that during the designing stages of a DBMS according to ORM specific models are created in four different levels of abstraction: conceptual design, logical design, external design and physical design (Halpin, Evans, Hallock, & Maclean, 2003).

We have also referred that the modeling language for those models is described by syntax, semantics and notation (Karagiannis & Kühn, 2002). In order to be able to manipulate models, this language needs to be specified as model of these models, meaning Meta-models (Sprinkle, Rumpe, & Vangheluwe, 2010). Thus a Meta-model is basically a model about the information being expressed during the modeling procedure, a meta-model is a model of a model, and also describes syntax and defines semantics (Geisler, Klar, & Pons, 1998; Sprinkle, Rumpe, & Vangheluwe, 2010). Meta Object Facility (MOF) standard of Object Management Group (OMG) provides a four hierarchy for meta-modeling in which the M0 level describes the basic data, the M1 level describes the meta-data, the M2 level is the meta-model and the M3 level is the MOF specification itself (Hinkelmann, 2011). ORM has no official standard for meta-modeling, thus an effort for this has taken place by Cuyler & Halpin (2003), in which they discuss how ORM components (object types, roles, predicates) may be metamodeled in ORM including component reuse, and also examines ways to metamodel business rules in ORM and metamodels instance data in ORM. According to them meta-modeling in ORM is important since is a better way for conceptual information analysis than ER and UML, is a possible standard for business rules expression and for use in ontology standards, it would facilitate the interchange of ORM model data between software tools and finally it aims to specify the grammar of a syntactically valid ORM model, which means that a Meta – model allows models to be checked for syntactic correctness.

Then we have described in detail the first level of analysis in ORM, meaning the Conceptual Schema Design Procedure. In this level we have described the concrete syntax of that is used in this modeling language, meaning how the fact types, object types, the roles played by entities in fact types, predicates, objectified associations, constraints (internal uniqueness constraints, external uniqueness, mandatory role constraints, subset constraints, exclusion constraints, equality constraints, frequency constraints, value constraints, ring constraints, subtyping constraints and textual constraints) are depicted, and we are also referred to the main steps are followed in this procedure (Halpin T. , Conceptual Schema & Relational Database Design, 1995-A).

After that we concerned with the Relational Mapping Procedure, in which as previous we describe the different layout options, the definitions & notation, the Rules of mapping and the strategies of mapping. Finally the main steps of mapping were presented.

Thus taking into account the previous, although there is no official standard for meta-modeling in ORM, in my opinion it uses a meta-modeling standard like this of OMG. In more detail as we have already described in this chapter the first two levels of this technique refers to the production of specific models, which are implemented taking into account a specific notation, rules, strategies and steps.

In the first level of ORM technique, meaning the Conceptual Schema Design Procedure, the information about the real word (UoD), can be considered as the M0 level in meta-modeling. Then this information, is depicted in a Conceptual Schema (in a drawing, in a conceptual model), which in turn can be considered as the M1 level in meta-modeling. Thereafter the building blocks or the meta-data that can be used to make this model are defined, meaning the abstract syntax of this modeling language. These blocks concerns the object types that can be used to present the model, the relations between the object types, the identifiers of the object types, the meaning of the object types (semantics) and the rules to combine the object types. Thus the previous level can be considered are the M2 level in meta-modeling. Finally the ORM is using a specific graphical notation, meaning the concrete syntax of this modeling language, which can be considered to be the M3 level. The same analysis as previous can be considered for the second level of analysis in ORM, meaning the Relational Schema Procedure, in which a second model is produced, which can be though as a meta-model since is a model that has being produced according to some other model.

CHAPTER 4: The Capability Meta – Model

Structure of this Chapter

- 4.1 An Initial Version of the Business Capability Meta – Model
- 4.2 Need for Change towards a New Business Capability Meta – Model
- 4.3 Chapter Summary

This chapter deals with working in the first level of our Approach according to ORM, by using an early version or redesigning a specific Conceptual Schema for Business Capability. Thus Section 4.1 discusses the initial version of this Business Capability meta-model. Then in Section 4.2 we redesign the previous meta-model by following the CSDP procedure of ORM and by this way we intend to use an accurate and correctly enough model for the development of the maritime application of Danaos Management Consultant Company. Finally in Section 4.3 a summary of this chapter is presented.

4.1 An Initial Version of the Business Capability Meta – Model

As we referred in a previous chapter the first stage in the designing process of a DBMS includes a conceptual schema design. This schema describes the structure or grammar of a specific Universe of Discourse, meaning ontologies such as object types, the roles between objects types and constraints.

For describing through a specific conceptual schema, the Universe of Discourse of the maritime application for Business Capability, we have taken into account an early work of Loucopoulos et al (2013). This work has set the scene for the definition of a meta-model that focuses on Business Capability as a concept, according to ORM2 (Halpin T. , 2005-A; Halpin T. , 2005-B). An initial version of this Capability meta-model is graphically shown in Figure 23.

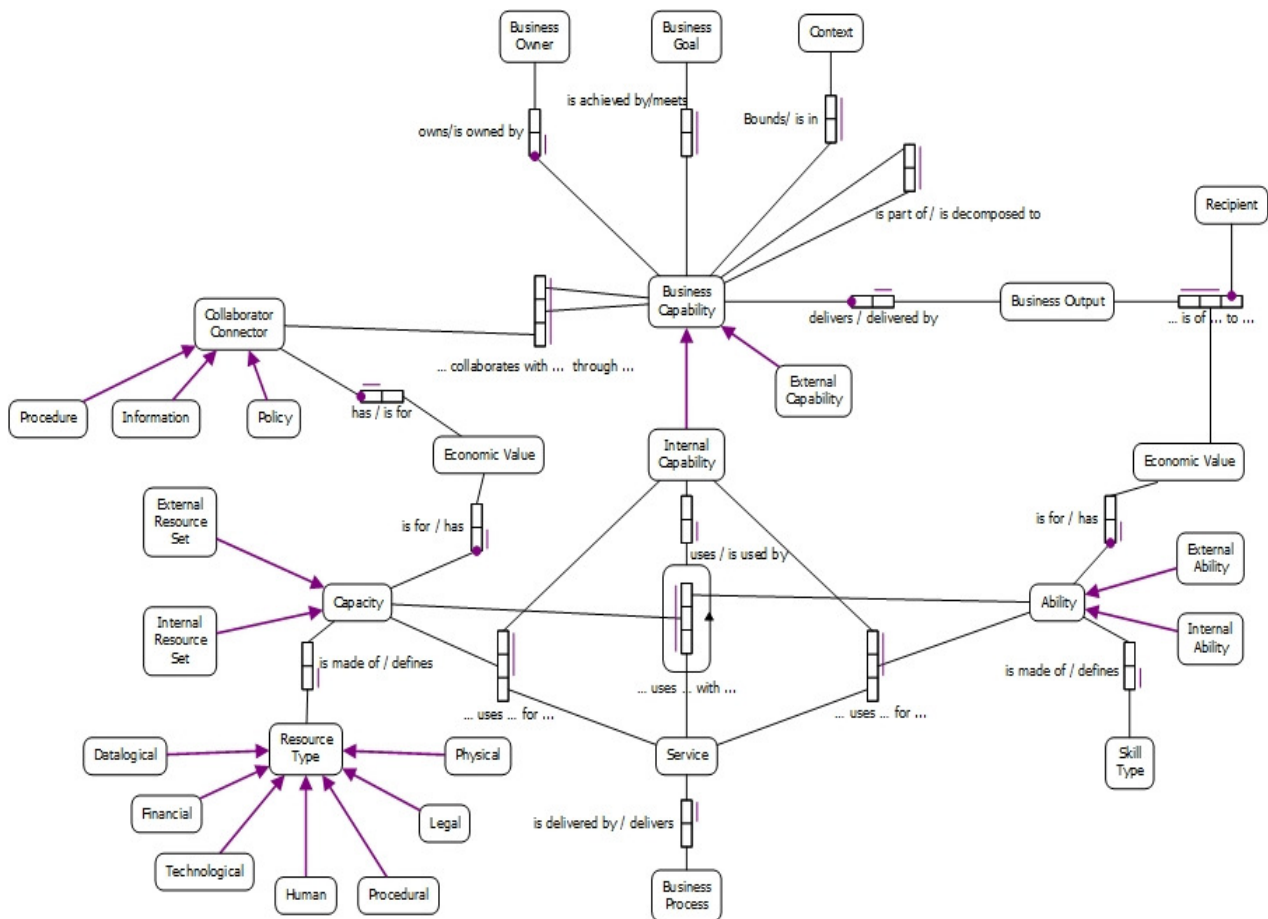


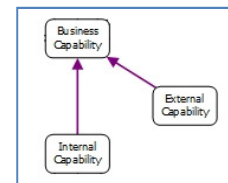
Figure 40: Business Capability Meta-Model (Loucopoulos, Bravos, Stratigaki, & Vavlis, 2013)

In general the Capability meta-model focuses on describing what a business can do and depicts the main components that characterize it from different viewpoints. Those are a Teleological View (goals, rules), an Operational View (Information, processes, transactions and roles), a Service View (service processes, atomic services and software services), a Contextual View (user context, business context, situation and variation) and a Capability

View (capacity, ability, ownership and value) (Loucopoulos, Bravos, Stratigaki, & Vavlis, 2013).

In fact if we use this Capability meta-model for creating the database of the maritime application, by reading it, we can understand in a first level what kind of information about Business Capability is required, and thereafter must be stored in our database. Also we can understand though the implemented constraints, the way this information must be stored in the database (e.g. relations, mandatory fields of tables, unique fields of tables etc).

In more detail, in this Universe of Discourse, a Business Capability is a notion that is used to describe the essential functions of an enterprise. However some of these functions may either owned by the enterprise or by some other enterprises. Thus there is a need for distinction the Business Capability into Internal Capability or External Capability. In that case the meta-model

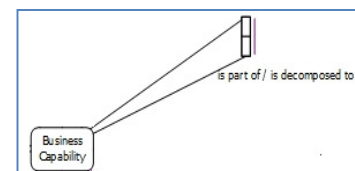


depicts a rule that says that there is a differentiation about the information that is stored in the database and this rule is related with the kind of Business Capability. If we verbalize this rule with the help of NORMA Tool of Visual Studio 2013 for Enterprise Architects we can say that:

Each Internal Capability is an instance of Business Capability.

Each External Capability is an instance of Business Capability.

Continuing with the description about the Capability meta-model, a Business Capability is part of or is decomposed to a Business Capability. In that case the meta-model depicts the information about the hierarchies of Business Capabilities, meaning the information about a relationship similar to a parent and child. Here the constraint indicates that the information that is stored for the combination of parent and child of Business Capability must be unique, meaning that it is not allowed duplicates for each instance of this combination. Also this kind of relationship is said to be many to many (m:n). If we verbalize this rule we can say that:

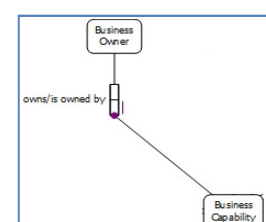


It is possible that some Business Capability is part of more than one Business Capability

and that some Business Capability is decomposed to more than one Business Capability.

In each population of Business Capability is part of Business Capability, each Business Capability, Business Capability combination occurs at most once.

Considering that a Business Capability has a Business Owner, from the meta-model we can understand for this UoD, the way that this information about ownership is maintained. Here the Business Capability column must be unique (meaning that no duplicates allowed) and also



mandatory (which means that this kind of data cannot be null). If we verbalize the constraints, we can say that:

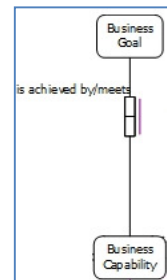
Each Business Capability is owned by exactly one Owner.

It is possible that some Owner owns more than one Business Capability.

For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

Since Business Capability describes what a business does, then maintaining the information about her relation with the Business Goals is important for this UoD.

Business Goals is a part of the planning process and describes what an organization expects to accomplish over a specific period of time. Thus for an organization it is important to establish goals for a specific Business Capability and have available the information about it, in order to be able to measure success and performance. Thereafter by reading the Capability meta-model we can understand the way this information will be maintained in the database of



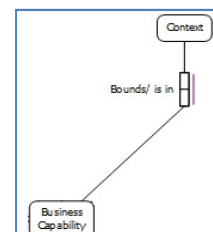
our application. Here the constraint indicates that the information that is stored for the combination of Business Capability and Business Goal must be unique, meaning that it is not allowed duplicates for each instance of this combination. Also this kind of relationship is said to be many to many (m:n). Finally if we verbalize the constraints of this meta-model, we can say that:

It is possible that some Business Capability meets more than one Business Goal

and that some Business Goal is achieved by more than one Business Capability.

In each population of Business Capability meets Business Goal, each Business Capability, Business Goal combination occurs at most once.

Also for this kind of UoD it is important to maintain the information about the relation between a Business Capability and the Context, meaning the environment within an organization operates. Changes in the Context may be affecting in a catastrophic way the operation of an organization. Thereafter maintaining this kind of information is crucial for the survival of an organization. Thus by reading the Capability meta-model we can



understand how this kind of information can be maintained in the database of our application. Here the constraint indicates that the information that is stored for the combination of Business Capability and Context must be unique, meaning that it is not allowed duplicates for each instance of this combination. Also this kind of relationship is said to be many to many (m:n). Finally if we verbalize the constraints, we can say that:

It is possible that some Business Capability is in more than one Context

and that some Context bounds more than one Business Capability.

In each population of Business Capability is in Context, each Business Capability, Context combination occurs at most once

Moreover in this UoD it is important to be described the information related to the Business Output that a Business Capability produces. When we are talking about Business Output we referred to the produced services that an organization delivers in order to increase his incomes.

Thereafter maintaining this information is also important for an organization. Thus by reading the meta-model we can see that the information about Business Capability is mandatory, which means that this kind of data cannot be null, and also the information about Business Capability must be unique (meaning that no duplicates allowed). If we verbalize the constraints, we can say that:

Each Business Capability delivers some Business Output.

Each Business Output delivered by at most one Business Capability.

It is possible that some Business Capability delivers more than one Business Output.

For this kind of information in the third case of verbalization we have a one to many (1:n) relationship and for the second case a many to one (n:1) relationship.

Since Business Output referred to the produced services that an organization delivers, then these services are having an economic value and received by specific Recipients, meaning that this actions results in a financial transaction. Thus in the meta-model we can see how this kind of information is maintained. Here the constraints says that for combination of Business Output and Economic Value is not allowed duplicates, and also that it is not allowed null records for the information about Recipient. Also we have a ternary relationship that is said to be many to many to one (m:m:1). If we verbalize the previous constraints we can say that:

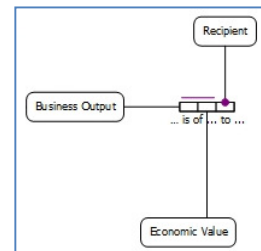
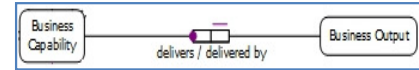
For each Business Output and Economic Value,

that Business Output is of that Economic Value to at most one Recipient.

For each Recipient,

some Business Output is of some Economic Value to that Recipient.

In this under research UoD, maintaining the information about collaborations between Business Capabilities, is the same important as previous. That's because these collaborations affect the economic outputs of an organization. When reading the meta-model we can see that a Business Capability collaborates with a Business Capability through a Collaborator Connector. Here the constraint says that for the combination of the three of them (Business Capability, Business Capability, Collaborator Connector) the information must be unique.



Also we have a ternary relationship which is said to be many to many to many (m:m:n). If we verbalize the constraints we can say that:

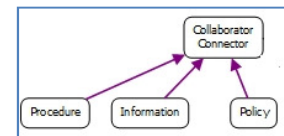
It is possible that for some Business Capability₁ **and** Business Capability₂, **that** Business Capability₁ collaborates with **that** Business Capability₂ through **more than one** Collaborator Connector.

and that for some Business Capability₁ **and** Collaborator Connector, **that** Business Capability₁ collaborates with **more than one** Business Capability₂ through **that** Collaborator Connector.

and that for some Business Capability₁ **and** Collaborator Connector, **more than one** Business Capability₂ collaborates with **that** Business Capability₁ through **that** Collaborator Connector.

In each population of Business Capability collaborates with Business Capability through Collaborator Connector,
each Business Capability, Business Capability, Collaborator Connector **combination occurs at most once.**

For this kind of collaborations a further analysis for the Collaborator Connector is depicted in the meta-model. A Collaborator Connector may be either Procedure or Information or Policy. This means that there is a distinction about the information that is stored according to the type of Collaborator Connector. If we verbalize the previous we can say that:

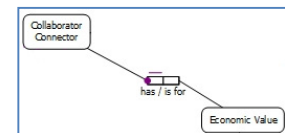


Each Procedure **is an instance of** Collaborator Connector.

Each Information **is an instance of** Collaborator Connector.

Each Policy **is an instance of** Collaborator Connector.

Also a Collaborator Connector has an Economic Value and via versa meaning an Economic Value is for a Collaborator Connector. This describes the involving of a financial transaction in that case. Here the constraints say that the Collaborator Connector information must be unique and also mandatory. If we verbalize the previous we can say that:



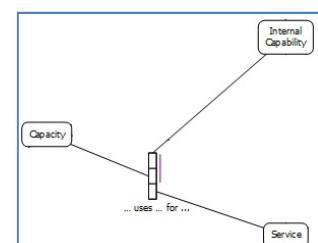
Each Collaborator Connector **has exactly one** Economic Value.

It is possible that some Economic Value **is for more than one** Collaborator Connector.

For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

For the rest of the UoD, the meta-model focuses in describing in more detail all the facts that related with Internal Business Capability.

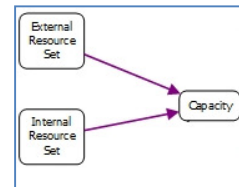
Thus an Internal Capability in order to produce its services it uses a Capacity. In fact the notion Capacity refers to the competence



of an organization in having the essential resources. Here the constraint says that for combination of Internal Capability and Capacity is not allowed duplicates. Also we have a ternary relationship that is said to be many to many to one (m:m:1). If we verbalize the constraints we can say that:

For each Internal Capability and Capacity,
that Internal Capability uses that Capacity for at most one Service.

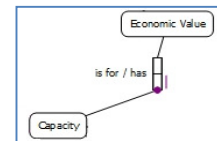
Then a further analysis to Capacity is depicted in the meta-model. More especially a Capacity may be either an External Resource Set or an Internal Resource Set, meaning the resources that owned by the organization or by other organizations. Here we have a rule that says that there is a distinction about the information that is stored according to the type of Capacity. If we verbalize the previous we can say that:



Each External Resource Set is an instance of Capacity.
Each Internal Resource Set is an instance of Capacity.

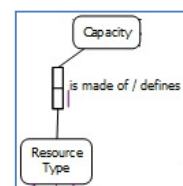
Also a Capacity has an Economic Value and via versa meaning an Economic Value is for a Capacity. This describes the involving of a financial transaction in that case. Here the constraints say that the Capacity information must be unique and also mandatory. If we verbalize the previous we can say that:

Each Capacity has exactly one Economic Value.
It is possible that some Economic Value is for more than one Capacity.



For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

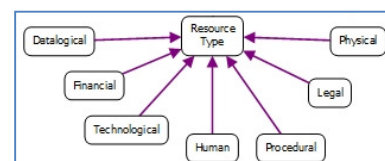
Finally Capacity is made of a Resource Type and via versa meaning a Resource Type defines Capacity. Here the constraints say that the information about the Resource Type must be unique. If we verbalize the previous we can say that:



Each Resources defines at most one Capacity.
It is possible that some Capacity is made of more than one Resource Type.

For this kind of information in the first case of verbalization we have a one to many (1:m) relationship and for the second case a many to one (n:1) relationship.

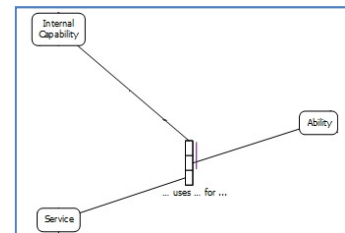
Different kind of Resources may be used by an organization in order to produce a service. Since Capacity uses a Resource Type, there was a need for depicting this distinction of Resources in the meta-model. In more detail a Resource Type may be either Datalogical or Financial or



Technological or Human or Procedural or Legal or Physical. If we verbalize the previous we can say that:

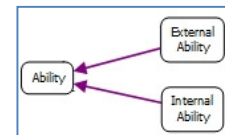
- Each Datalogical is an instance of Resource Type.
- Each Financial is an instance of Resource Type.
- Each Technological is an instance of Resource Type.
- Each Human is an instance of Resource Type.
- Each Procedural is an instance of Resource Type.
- Each Legal is an instance of Resource Type.
- Each Physical is an instance of Resource Type.

Except from Capacity an Internal Capability in order to produce its services it uses Ability. The notion Ability refers to the efficiency of an organization in having the knowledge of how to produce a Service. Here the constraint says that for combination of Internal Capability and Ability is not allowed duplicates. Also the ternary relationship is said to be many to many to one (m:m:1). If we verbalize the constraints we can say that:



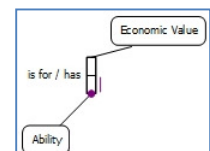
- For each Internal Capability and Ability,
- that Internal Capability uses that Ability for at most one Service.

Then a further analysis to Ability is depicted in the meta-model. More especially an Ability may be either External Ability or Internal Ability. That's because an organization may use the knowledge that has or may use a knowledge that some other organization has. Here we have a rule that says that there is a distinction about the information that is stored according to the type of Ability. If we verbalize the previous we can say that:



- Each External Ability is an instance of Ability.
- Each Internal Ability is an instance of Ability.

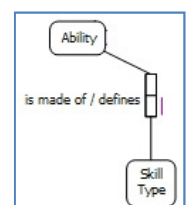
Also Ability has an Economic Value and via versa meaning an Economic Value is for Ability. This describes the involving of a financial transaction in that case. Here the constraints say that the Ability information must be unique and also mandatory. If we verbalize the previous we can say that:



- Each Ability has exactly one Economic Value.
- It is possible that some Economic Value is for more than one Ability.

For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

Finally Ability is made of a Skill Type and via versa meaning a Skill Type defines Ability. Here the constraints say that the information about the Skill Type must be unique. If we verbalize the previous we can say that:

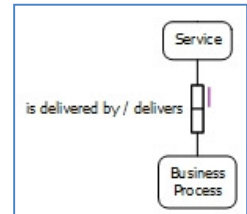


Each Skill Type defines at most one Ability,

It is possible that some Ability is made of more than one Skill Type.

For this kind of information in the first case of verbalization we have a one to many (1:m) relationship and for the second case a many to one (n:1) relationship.

As we have already said an Internal Capability uses a Capacity or an Ability to produce a Service. However an important think is the way this Service is delivered by the organization. Thus an organization in order to deliver this Service may use a specific Business Process. Here the constraints say that the Service information must be unique. If we verbalize the previous we can say that:

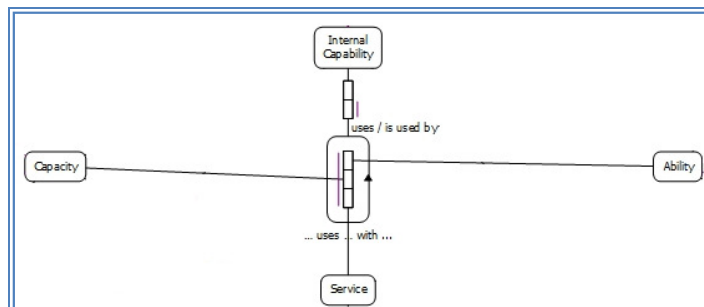


Each Service is delivered by at most one Business Process.

It is possible that some Business Process delivers more than one Service.

For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

Finally as we see in the meta-model an Internal Capability uses the combination of Capacity-Ability-Service in order to be able to operate.



If we want to verbalize the relation between Internal Capability and Capacity-Ability-Service we can say that:

Each AbilityUsesCapacityWithservice is used by at most one Internal Capability.

It is possible that some Internal Capability uses more than one AbilityUsesCapacityWithService.

For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship. Also in that case the combination of Capacity-Ability-Service must be unique.

On the other hand if we verbalize only the combination Capacity-Ability-Service, then we can say that:

It is possible that for some Ability and Capacity, that Ability uses that Capacity with more than one Service

and that for some Ability and Service, that Ability uses more than one Capacity with that Service

and that for some Capacity and Service, more than one Ability uses that Capacity with that Service.

In each population of Ability uses Capacity with Service, each Ability, Capacity, Service combination occurs at most once.

For this kind of information the relationship is said to be many to many to many (m:m:n) and also the data for each record must be unique.

This meta-model is complete enough and accurate in order to provide a good definition about Business Capability and describe all the concepts for this UoD. However in order to be used for the first level of the maritime application there is a need for change towards a new meta-model which will be a matter of discussed in the next Section.

4.2 Need for Change towards a New Business Capability Meta – Model

In the previous Section we discussed about a Capability Meta-Model, which was given in order to define the Business Capability ontology considerations of an organization in general. This is an accurate and complete meta-model for the purpose it has being designed; however it is not detailed and analytical enough in order to support the decryption of all the required by the specifications information for the maritime application, that will be used for the use case of data for Danaos Management Consultant company. This data is described in more detail in a next chapter (Chapter 8).

Thus there is a need for extended the [Loucopoulos et al \(2013\)](#) Capability meta – model into a new meta-model, that will be used for working with data in this first level of abstraction, for this UoD. In order to redesign this new conceptual model we will use the graphical notation of ORM 2 ([Halpin T. , 2005-B](#)) and the NORMA software based conceptual modeling tool for Object Role Modeling, which is an open source plug-in to Microsoft Visual Studio of Enterprise Architectures.

One first aspect during this redesigning procedure is to depict the model in a way that it will be clear for the reader where it starts and where it finishes. Thus we will use a top-down description of the ontologies, meaning we will depicted the different facts types in a vertically way. We will begin by following the **CSDP procedure (Conceptual Schema Design Procedure)** and by this way we intent to specify which information is missing from the initial meta-model.

In the first step of this procedure we begin by using the examples of data and expressed it in terms of elementary facts. Then in order to check on the quality of our work we ask ourselves the questions: Are the objects well defined? Can the facts be split into smaller ones without losing information? In this step for facilitate the reader we will use a sample of data that will be described in more detail in chapter 8.

Thus as we observe in the initial meta-model for the fact type that contains Business Capability and Owner (meaning Business Capability and Owner relation), according to the data we may have the elementary facts:

- a. *The Internal Business Capability 'INCAP1' **is owned by** the Owner 'OWN1'*
- b. *The Owner 'OWN1' **owns** the Internal Business Capability 'INCAP1'*
- c. *The Internal Business Capability 'INCAP4' **is owned by** the Owner 'OWN1'*
- d. *The Owner 'OWN1' **owns** the Internal Business Capability 'INCAP4'*
- e. *The External Business Capability 'EXCAP1' **is owned by** the Owner 'OWN2s'*
- f. *The Owner 'OWN2' **owns** the External Business Capability 'EXCAP1'*

Or

- a. *The Internal Business Capability 'Maritime Management Capability' **is owned by** the Owner 'DMC'*
- b. *The Owner 'DMC' **owns** the Internal Business Capability 'Maritime Management Capability'*
- c. *The Internal Business Capability 'Maritime Compliance Capability' **is owned by** the Owner 'DMC'*
- d. *The Owner 'DMC' **owns** the Internal Business Capability 'Maritime Compliance Capability'*
- e. *The External Business Capability 'Technical Assistance Management Capability' **is owned by** the Owner 'ComSys'*
- f. *The Owner 'ComSys' **owns** the External Business Capability 'Technical Assistance Management Capability'*

Or

- a. *'INCAP1: The Internal Business Capability 'Maritime Management Capability' **is owned by** the Owner 'OWN1: DMC'*
- b. *The Owner 'OWN1: DMC' **owns** the Internal Business Capability 'INCAP1: Maritime Management Capability'*
- c. *The Internal Business Capability 'INCAP4: Maritime Compliance Capability' **is owned by** the Owner 'OWN1: DMC'*
- d. *The Owner 'OWN1: DMC' **owns** the Internal Business Capability 'INCAP4: Maritime Compliance Capability'*
- e. *The External Business Capability 'EXCAP1: Technical Assistance Management Capability' **is owned by** the Owner 'OWN2: ComSys'*
- f. *The Owner 'OWN2: ComSys' **owns** the External Business Capability 'EXCAP1: Technical Assistance Management Capability'*

And so on

Thus it is confused which values of data referred on the two of them, meaning the code or the description or the combination of them, and thus must be stored in the

database. That's because are missing the essential reference modes and values types, meaning the manner in which the values referred to the Business Capability and Owner.

A more wright way to describe those elementary facts is by firstly identifying the wright reference modes for Business Capability and Owner. Since both of them have a code, we choose that for referred them. Thus the elementary facts will be:

- a. *The Internal Business Capability with code 'INCAP1' **is owned by** the Owner with code 'OWN1'*
- b. *The Owner with code 'OWN1' **owns** the Internal Business Capability with 'INCAP1'*
- c. *The Internal Business Capability with code 'INCAP4' **is owned by** the Owner with code 'OWN1'*
- d. *The Owner with code 'OWN1' **owns** the Internal Business Capability with code 'INCAP4'*
- e. *The External Business Capability with code 'EXCAP1' **is owned by** the Owner with code 'OWN2s'*
- f. *The Owner with code 'OWN2' **owns** the External Business Capability with code 'EXCAP1'*

Continuing a Business Capability for example 'INCAP4' has a specific description, meaning the 'Maritime Compliance Capability', and also an Owner for example 'OWN1' has a specific name, meaning 'DMC'. Thus for this cases we have a different properties for each object, which in fact identify those object types. Thus for this kind of information we transform it into two different elementary facts which will be:

- a. *The Internal Business Capability with code 'INCAP1' **has** the Capability Description 'Maritime Management Capability'*
 - b. *The Owner with code 'OWN1' **has** the Owner Name 'DMC'*
- and so on..*

For this step of the CSDP procedure, the same as previous exists for the other relations of the Capability meta-model. In more detail by using a 'Code' reference mode, we identify: The Goals, The Context, The Business Outputs, The Collaborator Connector, The Ability, The Skill Type, The Services, The Business Process and The Capacity. As far for Recipient we are using a 'Name' reference mode and for Economic Value a currency 'EUR:'. Finally for the previous information, in some cases an extra identification is needed in a form of 'Name' or 'Description'. If we transform those cases of data in elementary facts we can say:

- a. *The Goal with code 'INCAP4_GOAL1' **has** the Goal Name 'Goal9: To participate in research projects' and so on...*
- b. *The Context with code 'CONT1' **has** the Context Description 'Local Legislations' and so on...*
- c. *The Output with code 'INCAP4_OUTPUT1' **has** the Output Name 'Rule Compliance Service' and so on...*
- d. *The Ability with code 'INCAP4_INAB1' **has** the Ability Description 'The Ability to ease the transmission procedures of required compliance documents for the Port of Calls Application' and so on...*
- e. *The Skill Type with code 'SK6' **has** the Skill Name 'Master Degree in Project Management' and so on...*
- f. *The Service with code 'SERV5' **has** the Service Name 'E-Compliance System' and so on...*
- g. *The Business Process with code 'BP1' **has** the Process Name 'Business Process for Service Request and Quality Control' and so on...*

- h. *The Capacity with code 'INCAP4_INRES1' **has** the Capacity Description 'The Capacity to ease the transmission procedures of required compliance documents for the Port of Calls Application' and so on...*

Except from the previous cases described, then we are wondering if the facts type can split into a smaller one without losing the information. For those cases we can say that some extra information about the ontologies must be described, which is not depicted in the initial Capability meta-model and by that we can split some facts into a smaller one, in order this information to be described.

In more detail for Collaborator Connector except from the information about the hierarchies of them and the Economic Value that has, some extra information about the kind of code for the inserted values must be described. This can be specified by creating a new relation of this with a new entity the Connector Type. Thus if we transform this information in elementary facts we can say that:

- a. *The Collaborator Connector with code 'PO1' **has** the Connector Type with code 'POLICY' and so on...*
- b. *The Code with code 'Polity' **has** the Collaborator Connector with code 'PO1' and so on...*

The same exists for Business Process. When a Business Process is used by the company then specific tasks are executed by the administrator of this process. This kind of information is missing from the initial Capability meta-model. For identifying this task a 'name' is used as a reference mode. Thus if we transform this information in elementary facts we can say that:

- a. *The Business Process with code 'BP2' **leads to** the Task with the name 'Tasks (manual and user tasks): a) Collect forms with vessel's status b) Collect forms with cargo status etc.' and so on...*
- b. *The Task with the name 'Tasks (manual and user tasks): a) Collect forms with vessel's status b) Collect forms with cargo status etc.' **is executed for** the Business Process with the code 'BP2' and so on...*

Also for the case of Resources that defines Capacity, in the real examples of data that we have from the DMC Company, a Resource may have a code, a description and also is categorized in different types. The initial Capability meta-model missing a lot of this information, thus in that case we consider that we have the elementary facts:

- a. *The Capacity with code 'INCAP4_INRES1' **is made of** the Resources with code 'HU_INRES4.1' and so on...*
- b. *The Resources with code 'HU_INRES1' **defines** Capacity with the code INCAP4_INRES1" and so on...*
- c. *The Resources with code 'HU_INRES1' **has** a Resource Type with code 'HU' and so on....*
- d. *The Resources with code 'HU_INRES1' **has** the Resources Description '2 Software engineers from the IT department and 1 Project Manager' and so on...*

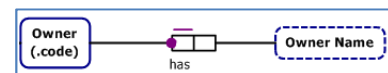
In the second step of the CSDP procedure we draw the additional corrections in a new Capability meta-model and apply a population check. Then ***in the third step of this procedure*** we check for object types that should be combined, and note any arithmetic

derivations. In this last step although there are no arithmetic derivations that must be noted, there is in some cases some object types that can be combined. For example in the Capability meta-model a Collaborator Connector has an Economic Value and the same exists for the Capacity, the Ability and the Business Output. In this model the modeler has chosen to depict twice the ontology of Economic Value. Thus in the new Capability meta-model, we have described the Economic Value in one object type that related with the others.

Next for the other two following steps of the CSCP procedure (step 5 & 6) we implement the changes at the same time. More specific in the new Capability meta-model we will add the uniqueness constraints, we will check the arity of the fact types, we will we add the mandatory roles constraints and if is needed we will check for logical derivations.

In that point for the cases of the initial Capability meta-model that has being kept as is was, the uniqueness constraints, the arity of fact types and the mandatory role constraints maintain the same, since they describes correctively the way the data will be stored in the database for the maritime application. Thus we will only discuss the previous for the fact types that we have added.

In more detail when we describing the information about the Owner of the company, a code is needed to identify him and also a Name. In that case an Owner has exactly one code and also an exactly one name, meaning it cannot be identified by a more than one codes or more than one names or to have none of this identifiers. Thus we will add a uniqueness constraint in the role of Owner, which mean that Owner code in this table must be unique, and also a mandatory role constraint which says that this value it cannot be null. If we verbalize this rule with the help of NORMA Tool of Visual Studio 2013 for Enterprise Architects we can say that:



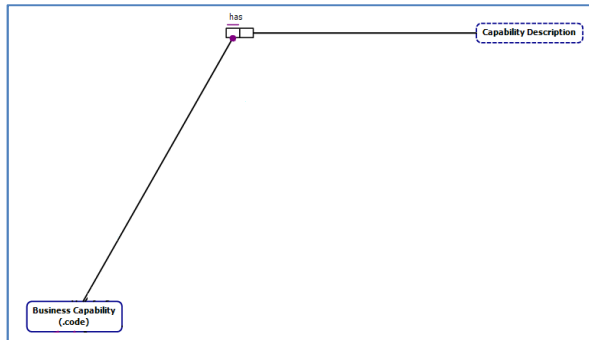
Each Owner has exactly one Owner Name.

It is possible that more than one Owner has the same Owner Name.

For this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

The same as the previous example of Owner exist for:

a. *The identification of Business Capability:*



In this case we say:

Each Business Capability has exactly one Capability Description.

It is possible that more than one Business Capability has the same Capability Description.

b. *The identification of Business Goal:*



In this case we say:

Each Business Goal has exactly one Goal Name.

It is possible that more than one Business Goal has the same Goal Name

c. *The identification of Context:*

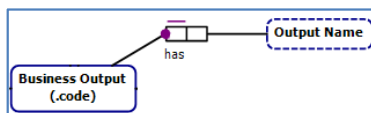


In this case we say:

Each Context has exactly one Context Description.

It is possible that more than one Context has the same Context Description.

d. *The identification of Business Output:*

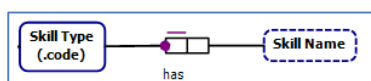


In this case we say:

Each Business Output has exactly one Output Name.

It is possible that more than one Business Output has the same Output Name.

e. *The identification of Skill Type:*

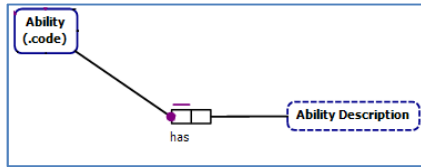


In this case we say:

Each Skill Type has exactly one Skill Name.

It is possible that more than one Skill Type has the same Skill Name

f. The identification of Ability:

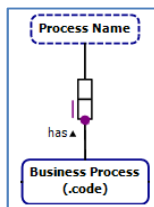


In this case we say:

Each Ability has exactly one Ability Description.

It is possible that more than one Ability has the same Ability Description.

g. The identification of Business Process:

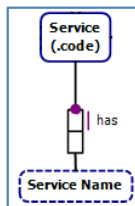


In this case we say:

Each Business Process has exactly one Process Name.

It is possible that more than one Business Process has the same Process Name.

h. The identification of Service:

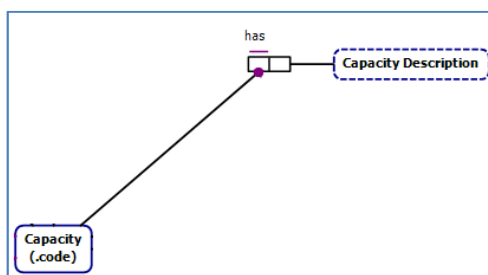


In this case we say:

Each Service has exactly one Service Name.

It is possible that more than one Service has the same Service Name.

i. The identification of Capacity:

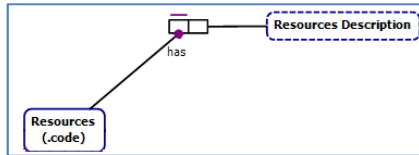


In this case we say:

Each Capacity has exactly one Capacity Description.

It is possible that more than one Capacity has the same Capacity Description.

j. *The identification of Resources:*

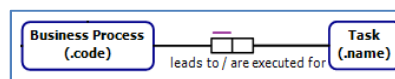


Finally in this case we say:

Each Resources has exactly one Resources Description.

It is possible that more than one Resources has the same Resources Description.

Then we continue with the relation between the Business Process and Task. Here we have a fact that is says that a Business Process with a specific code leads to a Task with specific name and another that says Tasks with a specific name are executed by Business Process with a specific code. For this combination of data the information about Business Process must be unique, which means that it is not allowed duplicates when entering the Business Process Code. Thus in this case we have added a uniqueness constraint in the role that plays the Business Process as follows:



In this case if we verbalize this constraint we can say:

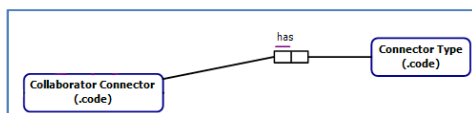
Each Business Process leads to at most one Task.

It is possible that some Task are executed for more than one Business Process.

Also for this kind of information in the first case of verbalization we have a many to one (n:1) relationship and for the second case a one to many (1:n) relationship.

The same as previous exists for:

a. *The relation between the Collaborator Connector and the Connector Type.*

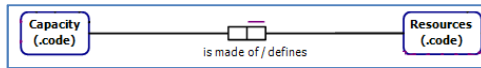


In this case if we verbalize this constraint we can say:

Each Collaborator Connector has at most one Connector Type.

It is possible that more than one Collaborator Connector has more than one Connector Type.

b. *The relation between the Capacity and the Resources.*

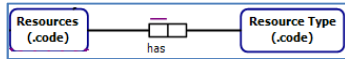


In this case if we verbalize this constraint we can say:

Each Resources defines at most one Capacity.

It is possible that some Capacity is made of more than one Resources.

c. The relation between the Resources and the Resource Type.



In this case if we verbalize this constraint we can say:

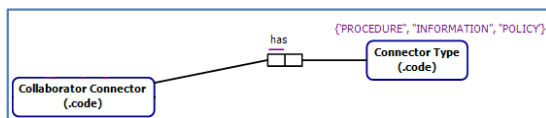
Each Resources has at most one Resource Type.

It is possible that some Resources has more than one Resource Type.

Finally there are no cases for checking for logical derivations in this model.

We then continue with **the Step 7 of the CSCP procedure**. We remind that in this step we add value, set comparison and subtyping constraints. We also mention that cases for the second option of the previous are not implemented in our model.

Thus we will start with the value constraints. Here for the Connector Type, we have already said that is identified by a specific code. This code referred to specific values which are 'Procedure', 'Information' and 'Policy' and nothing else except from the three of them. Thus for this type of values we have added a value constraint in the Connector Type as follows:



If we verbalize this we can say that:

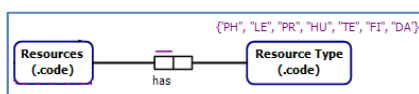
The possible value of Connector Type_code in Connector Type has Connector Type_code is 'PROCEDURE', 'INFORMATION', 'POLICY'.

The adding of this constraint change the way we verbalize the fact type. Now we can say that:

Each Collaborator Connector has at most one Connector Type.

It is possible that more than one Collaborator Connector has the same Connector Type.

The same exists for the Resource Type. Here the code takes specific values as follows:



If we verbalize this we can say that:

The possible value of Resource Type_code in Resource Type has Resource Type_code is 'PH', 'LE', 'PR', 'HU', 'TE', 'FI', 'DA'.

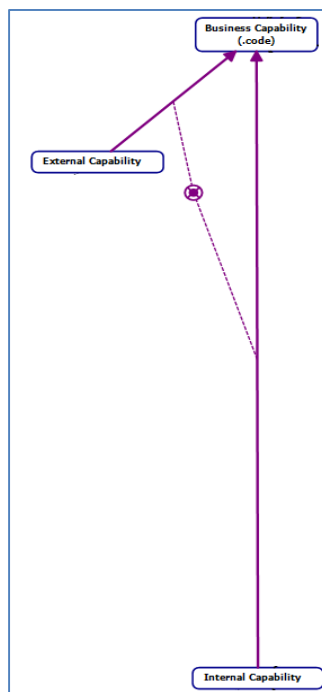
The adding of this constraint change the way we verbalize the fact type. Now we can say that:

Each Resources has at most one Resource Type.

It is possible that some Resources has the same Resource Type

We now continue with the subtype constraints. We reminding that subtyping constraints are mutually **exclusive** (\otimes): there is a distinction between the subtypes; collectively **exhaustive** (\odot): subtypes equals the union of the supertype; and **the combination of the previous two** (\oplus): where subtypes partition the supertype.

As we have already described Business Capability is divided into Internal Capability and External Capability. The last two subtypes equal the union of the supertype Business Capability. That's because Internal Capability is a Business Capability and the same exists for External Capability and thereafter if you take the union of them, then we have the total Business Capability. Also there is a distinction between them since Internal Capability refers to Capabilities that owned by the company, but External Capability describes the Capabilities that owned by some other companies. Thus in that case an exclusive and exhaustive constraint (\oplus) is added between of them as follows:

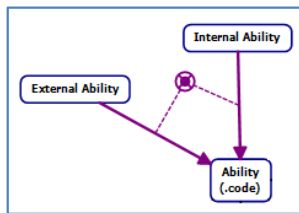


In this case if we verbalize the subtype constraint we can say:

For each Business Capability, exactly one of the following holds:
that Business Capability is some External Capability;
that Business Capability is some Internal Capability.

The same as Business Capability exists for:

- a. The Ability, who is divided into Internal Ability and External Ability.



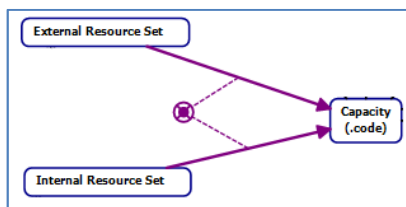
In this case if we verbalize the subtype constraint we can say:

For each Ability, exactly one of the following holds:

that Ability **is some** External Ability;

that Ability **is some** Internal Ability.

- b. The Capacity, who is divided into Internal Resource Set and External Resource Set.



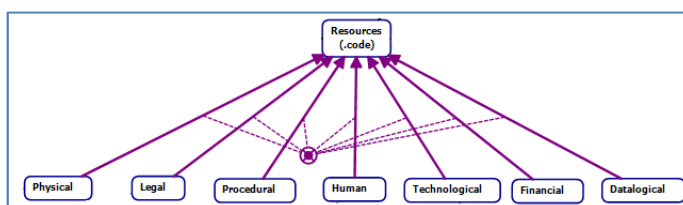
In this case if we verbalize the subtype constraint we can say:

For each Capacity, exactly one of the following holds:

that Capacity **is some** External Resource Set;

that Capacity **is some** Internal Resource Set.

- c. The Resource, who is divided into Phycical, Legal, Procedural, Human, Technological, Financial and Datalogical.



In this case if we verbalize the subtype constraint we can say:

For each Resources, exactly one of the following holds:

that Resources **is some** Phycical;

that Resources **is some** Legal;

that Resources **is some** Procedural;

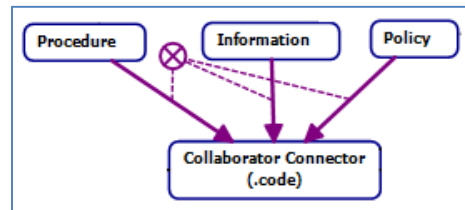
that Resources **is some** Human;

that Resources **is some** Technological;

that Resources **is some** Financial;

that Resources **is some** Datalogical.

As far for Collaborator Connector, there is difference compared with previous cases. Here there is a distinction between subtypes Procedure, Information and Policy, since each of them describes a different kind that is used as a connector for the collaborations between capability, but we are not sure if the total of them equals to this Collaborator Connector. Thus in that case an exclusive constraint (⊗) is added between them as follows:



In this case if we verbalize the subtype constraint we can say:

For each Collaborator Connector, at most one of the following holds:

that Collaborator Connector is some Policy;

that Collaborator Connector is some Information;

that Collaborator Connector is some Procedure.

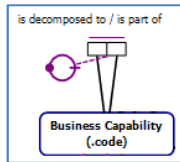
Finally in **the Step 8 of the CSCP procedure** we add other constraints and we perform final checks. In more detail we will add some ring constraints in the ring fact types and also some textual constraints for every derived fact type.

In the initial Capability meta-model a hierarchy of Business Capability is depicted, meaning a parent-child relationship. This relationship indicates a table that contains all the information about a Main Business Capability (parent) and its Sub Business Capabilities (child). For example in the DMC Company, the Main Capability 'INCAP4: Maritime Compliance Capability' is decomposed into the Sub Capabilities 'INCAP4.1: Vessel Monitoring Capability, INCAP4.2: Port Regulation Monitoring Capability and INCAP4.3: Regulation Inconsistencies Reporting Capability. However in this model there is no check for identifying how these two ontologies (Main Capability and its Sub Capability) will be maintained in every instance of this table. This has to do with ring constraints.

In more detail for the previous example we say that the Main Business Capability cannot bear a relationship with itself meaning that we cannot have a table with an instance in this form:

Main Capability	Sub Capability
INCAP4	INCAP4

This type of relationship is said to be irreflexive for each of them and is depicted with the symbol ↻. Thus for Business Capability this constraint is implemented as follows:

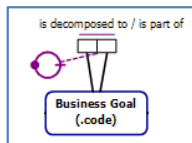


In this case if we verbalize the ring constraint we can say:

No Business Capability is decomposed to **the same Business Capability**.

The same as Business Capability exists for:

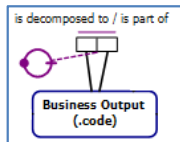
a. *Business Goals*.



In this case if we verbalize the ring constraint we can say:

No Business Goal is decomposed to **the same Business Goal**.

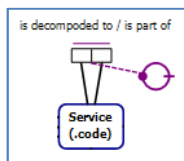
b. *Business Output*.



In this case if we verbalize the ring constraint we can say:

No Business Output is decomposed to **the same Business Output**.

c. *Service*.



In this case if we verbalize the ring constraint we can say:

No Service is decomposed to **the same Service**.

Finally we have added some textual constraints in order to specify every derived fact type exists, meaning note that contains a rules who says how one fact type may be derived from others. For example an External Business Capability is a Business Capability who is decomposed to some Business Capability etc.

All of the previous corrections that described in this section are depicted in the new Capability meta-model, which follows in [Figure 41](#). This meta-models will be used in the next Chapter for designing a Relational Schema.

4.3 Chapter Summary

Summarizing, in this chapter we have worked in a first level of the ORM technique, meaning we have tried to design a specific conceptual schema, which will be used for the maritime application of Business Capability. As we have already stated, this schema describes the structure or grammar of a specific Universe of Discourse, meaning ontologies such as object types, the roles between objects types and constraints.

Thus firstly we have taken an initial version of a Capability meta-model, which defines Business Capability and has being designed according to ORM (Loucopoulos, Bravos, Stratigaki, & Vavlis, 2013), in order to examine if it is a capable model for using it during the designing of the maritime application.

This meta-model has being focused on describing what a business can do and characterize that from a Teleological View (goals, rules), an Operational View (Information, processes, transactions and roles), a Service View (service processes, atomic services and software services), a Contextual View (user context, business context, situation and variation) and a Capability View (capacity, ability, ownership and value) (Loucopoulos, Bravos, Stratigaki, & Vavlis, 2013). By reading this meta-model a modeler can understand in a first level what kind of information about Business Capability is required, and thereafter must be stored in a database, and also though the implemented constraints, he can understand the way this information must be stored in the database (e.g. relations, mandatory fields of tables, unique fields of tables etc). For that reason we have described in natural language all the ontologies of this meta-model, meaning we explain all objects in detail, we verbalized in natural language all the implemented constraints and finally we have provide a description about the kind of relationships (cardinalities) it depicts.

By this description we have concluded that this meta-model is accurate and competed enough for the purpose it has being designed, however it is not detailed and analytical enough in order to support the decryption of all the required by the specifications information for the maritime application, that will be used for the use case of data for Danaos Management Consultant company.

Thus in the second part of this chapter we redesign this meta-model, by using the graphical notation of ORM 2 (Halpin T. , 2005-B) and the NORMA tool of Microsoft Visual Studio of Enterprise Architectures, and by this way we were intended to describe all the missing information from the initial meta-model. To achieve the previous we follow from the beginning the CSDP procedure (Conceptual Schema Design Procedure), and described in detail his steps for our UoD. The result of this work was an accurate and complete

Conceptual Model for Business Capability description of the Maritime Application, which will then be used in the next chapter for creating the Relational Schema for our application.

From the above we can understand that an important factor when designing a Conceptual Schema is the purpose of designing. This purpose usually specifies what kind of information must be depicted and the way is depicted. On the other hand when describing a specific UoD by a Conceptual Schema different patterns may be produced according to the way of thinking of the modeler. For example taking the previous initial of Conceptual Schema for Business Capability was an important help for producing an accurate and complete model for our Application. That's because this model became the basis for the designing of our model, since it has describe correctively the way most of the data required to be stored in the maritime application. On the other hand in the new Business Capability meta-model, some fact types have being depicted in a different way than the initial meta-model. The previous has to do with the way of thinking of the modeling. However, the most important in both cases is not the way of depicting the different models by the modelers, but the prevention of missing important information about the under description of UoD. That' why we are also agree that in the CSPD procedure the most important part is the first step, where examples of data are express in term of elementary facts. This is a step where if the information about the UoD is not expressed in detail, then there is a big possibility to lead into a missing of data. Thus in this step it is important for the modeler to have a full access and permission by the company to the all available information about it.

CHAPTER 5: Mapping the Capability Meta – Model to Relational Schema

Structure of this Chapter


- 5.1 Followed Procedure of Mapping in Detail
- 5.2 Chapter Summary

This chapter deals with working in a second level of our Approach according to ORM, meaning by designing the Relational Schema that will be used for the maritime application of Danaos Management Consultant Company. Thereafter we have taken into account the Business Capability meta-model that was designed in a previous chapter and in this chapter we follow a specific procedure of mapping in order to produce a Relational Schema. Thus Section 5.1 describes this procedure in detail and finally in Section 5.2 a summary of this chapter is presented.

5.1 Followed Procedure of Mapping in Detail

Taking into account the Relational Mapping Procedure that discussed in unit 3.3 of this dissertation, we consider how to implement this on the new Capability Meta-Model discussed in Chapter 4.

Following Step 0 we can see that in the Capability meta-model are no unaries and any relative closure (**sub-step 0.1**). However there are enough reference (primary identification) predicates (eg. Capability Description etc) and one compositely identified object type. Thus, we mentally erase all of them and we are concerning the compositely identified object types as “black boxes” and treat them as simple object types. We have only one nested object type. We will call this nested object type as “*Abilityusescapacitywithservice*” (**sub-step 0.2**). Thus, we generate the table named “*Abilityusescapacitywithservice*” and treated the objectified object type as “black box”. Thus we have the table:

Abilityusescapacitywithservice ()

We then indicate any non-absorption choices for subtypes (**sub-step 0.3**). The main criterion about this is whether the supertype and its subtypes play non-functional roles. If they play functional roles then we choose the absorption option. So, the Capability meta-model has five supertypes. Those are the “Business Capability”, the “Collaboration Connector”, the “Capacity”, the “Resources” and the “Ability”.

Starting with “Business Capability” supertype, we can see that the roles played by that and its supertypes “External Capability” and “Internal Capability”, are not functional. Actually the subtype “Internal Capability” has non-functional roles. In case like this ***the option of absorption is unfeasible***, because any non-functional roles of the subtypes map to separate tables anyway. So, let’s see the option of partition, examining whether supertypes are exclusive and exhaustive. This means that a) “Internal Capability” \cap “External Capability” = { } and also b) “Internal Capability” \cup “External Capability” = Business Capability. The first one is true, as there is distinction between the two of them (meaning that they cannot have common instances). That’s because Internal Capability refers to the capabilities owned by the business and External Capability refers to the capability owned by some other enterprises. The second one is also true because the totally of Internal and External Capability is Business Capability. According to them ***we will choose partition approach for supertypes***. Noted, if supertypes were not exhaustive, then we would choose the separation approach, because in a non-exhaustive situation a separate table is needed anyway. Thus, we generate two tables named “*Incapability*” and “*Excapability*”. The Business Capability

object type has the reference mode “code”, thus ***we generate the primary keys that identify those tables as “incapcode” and “excapcode”***. The union of those primary keys is the totality of Business Capability code named in our schema “capcode”. We are underlying primary keys of each table, since it has to be unique. Also the primary keys cannot be null according to the Entity Integrity Rule. Thus we have the tables:

Incapability (incapcode, ...)

Excapabiliy (excapcode, ...)

Following the same procedure with the previous paragraph, we see that “Collaboration Connector” supertype has no-functional roles, while its subtypes “Procedure”, “Information” and “Policy” has functional roles. In case like this ***the option of absorption is unfeasible***. Also subtypes Procedure”, “Information” and “Policy are exclusive ($\text{“Procedure”} \cap \text{“Information”} \cap \text{“Policy”} = \{ \}$), but are not exhaustive. This is because collaborations with capabilities (internal or external) may exist for now, but a new collaborator connector might be introduced in the future. This means that a capability may provide some other connector to another capability in the future. Whether the subtypes are not both exclusive and exhaustive then ***the option of partition is not chosen***. Thus ***we will choose the option of separation*** for the subtype. This option indicates each object type to be mapped to a separate table. Thus, *we generate a hierarchy of tables named “Collaborator”, “Procedures”, “Information” and “Policy”*. Also in this case the reference mode “code” will be the primary key for the table “Collaborator” and the foreign key ***for the other tables “Procedures”, “Information” and “Policy”***. Thus for the table Collaborator we have the primary key “collabcode”, which will be a foreign key for the other tables. The union of those foreign keys is the totality of the primary key “collabcode”. Also the foreign keys are also and primary keys for the tables “Procedures”, “Information” and “Policy”. Finally we are underlying primary and foreign keys, since it has to be unique. Thus we have the tables:

Collaboratorconnector (collabcode, ...)

Procedures (collabcode)

Information (collabcode)

Policy (collabcode)

Thereafter we see that “Capacity” supertype has non-functional roles, while its subtypes “External Resource Set” and “Internal Resource Set” has functional roles. Same as previous ***the option of absorption is unfeasible***. On the other hand subtypes “External Resource Set” and “Internal Resource Set” are both exclusive and exhaustive. This means that a) “Internal Resource Set” \cap “External Resource Set” = { } and also b) “Internal Resource Set” \cup “External Resource Set” = Capacity. A capacity refers to all kind of resources business has and also to resources that business buys from other enterprises. This mean that there is a distinction between the two subtypes and the total of them equals to Capacity. Considering the previous ***we will choose partition approach for supertypes***. Thus, we generate two tables named “Incapacity” and “Excapacity”. The Capacity supertype has the reference mode “code”, thus ***we generate the primary keys that identify those tables as “incapacode” and “excapacode”***. The union of those primary keys is the totality of Capacity code named in our schema “capacode”. Thus we have the tables:

Incapacity (incapacode, ...)

Excapacity (excapacode, ...)

As far as “Resources” supertype ***we will choose an absorption approach***. That’s because the supertype and its subtypes “Datalogical”, “Financial”, “Technological”, “Human”, “Procedural”, “Legal” and “Physical”, play functional roles. In this case we generate just one table named “Resources”. The other two approaches separation and partition would result us to more tables and this is not efficient. The Resources supertype has the reference mode “code”, thus ***we generate the primary key “rescode” for that table***. Thus we have the table:

Resources (rescode, ...)

Finally the “Ability” supertype has non-functional roles, while its subtypes “External Ability” and “Internal Ability” has functional roles. This means that ***we cannot use the absorption approach***. Both subtypes equals the total ability (Internal Ability” \cup “External Ability” = Capacity), which means that subtypes are exhaustive. Also “Internal Ability” referred to the ability that the business has, but “External Ability” referred to the ability that have some other enterprises. So, both of them are exclusive. According to previous ***we will choose a partition approach***. We then generate two tables named “Inability” and “Exability”. Since there is the reference mode “code” in “Ability” object type, ***we generate the “inabcode” and the “exabcode” as the primary keys that identify those tables***. The

union of those primary keys is the totality of Ability code named in our schema “abcode”.
Thus we have the tables:

Inability (inabcode, ...)

Exability (exabcode, ...)

Next step concerns the indication of the derived fact types that must be stored (**sub-step 0.4**). In our case we have no derived fact types that must be stored.

Also in Capability meta-model there are no symmetric 1:1 cases (**sub-step 0.5**), no disjunctive reference schemes (**sub-step 0.6**) and no cases where an objectified predicate is not spanned by a uniqueness constraint (**sub-step 0.7**). We then continue to the next step.

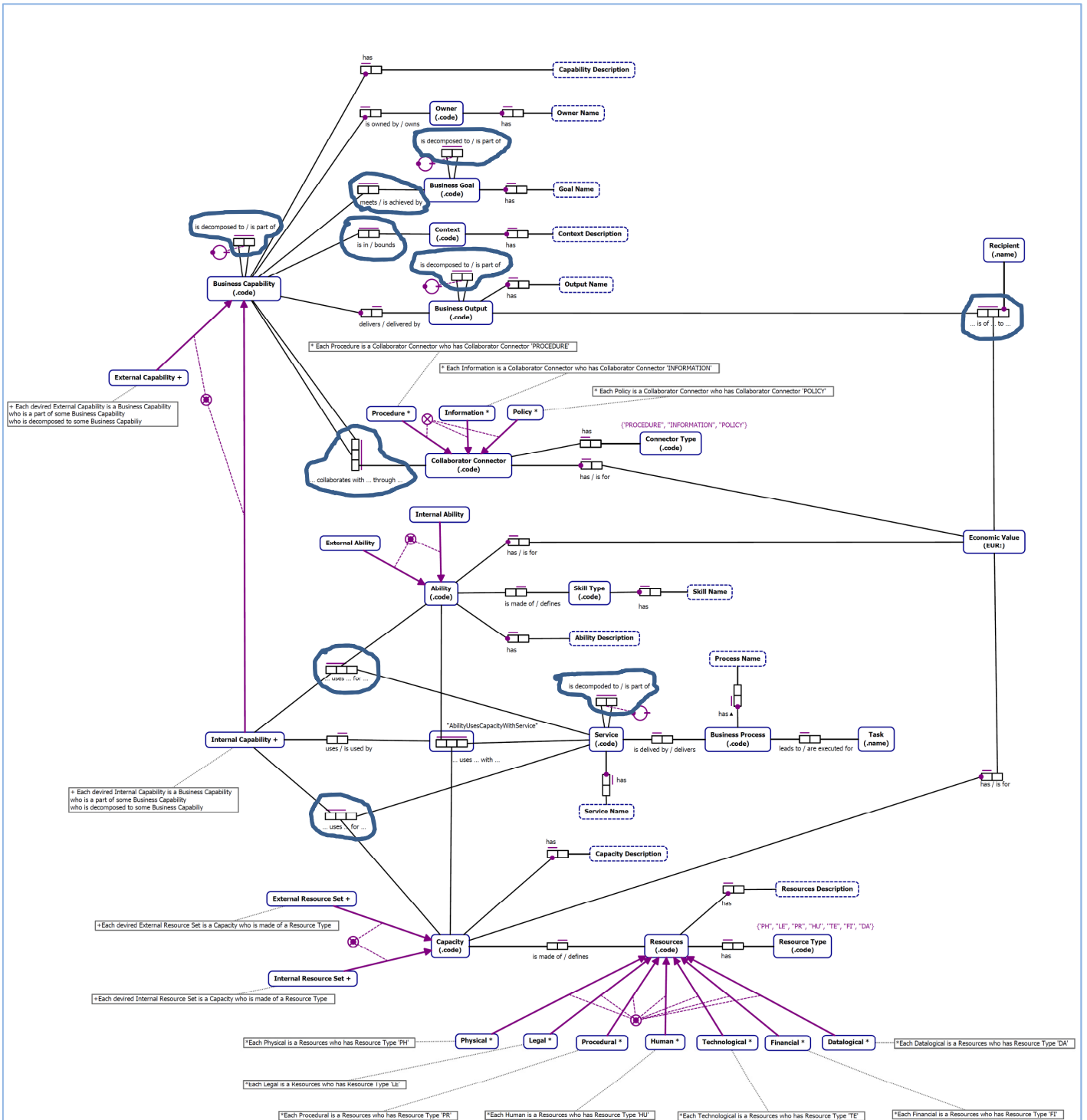
In Step 1 we look around for a predicate with a compound uniqueness constraint.

Those are m:n binaries and n-aries ($n \geq 3$) predicates. Those predicates are:

- a) The m:n binaries:
 - ...meets.../... is achieved by ...
 - ...is in.../...bounds...
 - ...is decomposed to.../...is part of... (that refers to Business Capability)
 - ...is decomposed to.../...is part of... (that refers to Business Goal)
 - ...is decomposed to.../...is part of... (that refers to Business Output)
 - ...is decomposed to.../...is part of... (that refers to Service)

- b) The n-aries predicates (m:m:n & m:m:1):
 - ...collaborates with...through...
 - ...is of...to...
 - ...uses...for... (that refers to Capacity)
 - ...uses...for... (that refers to Ability)

To help visualize we place a lasso to them, indicating that each of them goes to a table by itself as follows:



So we will have ten tables on the relational schema. In more detail for the predicate “...meets.../... is achieved by...”, we generate the table named “Capabilitymeetsgoal”, putting the object types “Business Capability” and “Business Goal” as column names. Since the totality of Business Capability code is the “capcode”, we use this as primary key of that table and also we create the “goalcode” for identify Business Goal, since the object type “Goal” has reference mode “code”. The “goalcode” also will be a primary key for that table. The

uniqueness constraint provides the candidate key for identifying rows and this unique column combination is shown by underling in this table. Thus the combination “capcode-goalcode” is the candidate key for that table. However the “goalcode” will also be a foreign key for that table. Thus we have **the final table**:

Capabilitymeetsgoal (capcode, goalcode)

The same as previous exists for the predicate “...is in.../...bounds”. Here we generate the table named “Capabilityisincontext”. The Business Capability object type is identified by “capcode” and we generate “contcode” to identify the object type Context. Both “capcode” and “contcode” are primary keys for that table and the last one is also a foreign key. Also the combination “capcode-contcode” is the candidate key for that table. Thus we have **the final table**:

Capabilityisincontext (capcode, contcode)

As far for the ring predicate “...is decomposed to.../...is part of...” of Business Capability, we generate the table “Capabilityispartof”. This predicate shows the hierarchy of Business Capability into sub capabilities. In that case we distinguish “capabilitycode” into “mcapcode” for describing the main Business Capability and into “subcapcode” for describing the sub capabilities. Both “mcapcode” and “subcapcode” are primary keys for that table and so the combination “mcapcode-subcapcode” is the candidate key for that table. Thus we have **the final table**:

Capabilityispartof (mcapcode, subcapcode)

The same as previous exists in the ring predicate “...is decomposed to.../...is part of...” of Business Goal, “...is decomposed to.../...is part of...” of Business Output and “is decomposed to.../...is part of...” of Service. In that case we generate the tables “Goalispartof”, “Outputispartof” and “Serviceispartof”. Thus we have **the final tables**:

Goalispartof (mgoalcode, subgoalcode)

Outputispartof (moutputcode, suboutputcode)

Serviceispartof (mservcode, subservcode)

For the ternary predicate “...collaborates with...through...”, we generate the table named “Collaborations”. Since this predicate describe collaborations with external

capabilities and indeed between internal capabilities, which may also be and subcapabilities, we will use as primary keys the “capcode1” and “capcode2” to identify “Business Capability” and their sub capabilities, and the “collabcode” as a primary key to identify the “Collaborator Connector” object type. The “collabcode” will be also and a foreign key for that table. Thus the combination “capcode1-capcode2-collabcode” is the candidate key for that table. Thereafter we have **the final table**:

Collaborations (capcode1, capcode2, collabcode)

For the predicate “...is of...to...”, we generate the table named “Outputisofvalue”. Here we will use the “outputcode” as a primary and foreign key for identifying the object type “Business Output”. The candidate key is the unique column combination of outputcode-evalue. Also the column “recipientname” is mandatory. Thus **we have the final table**:

Outputisofvalue (outputcode, evalue, recipientname)

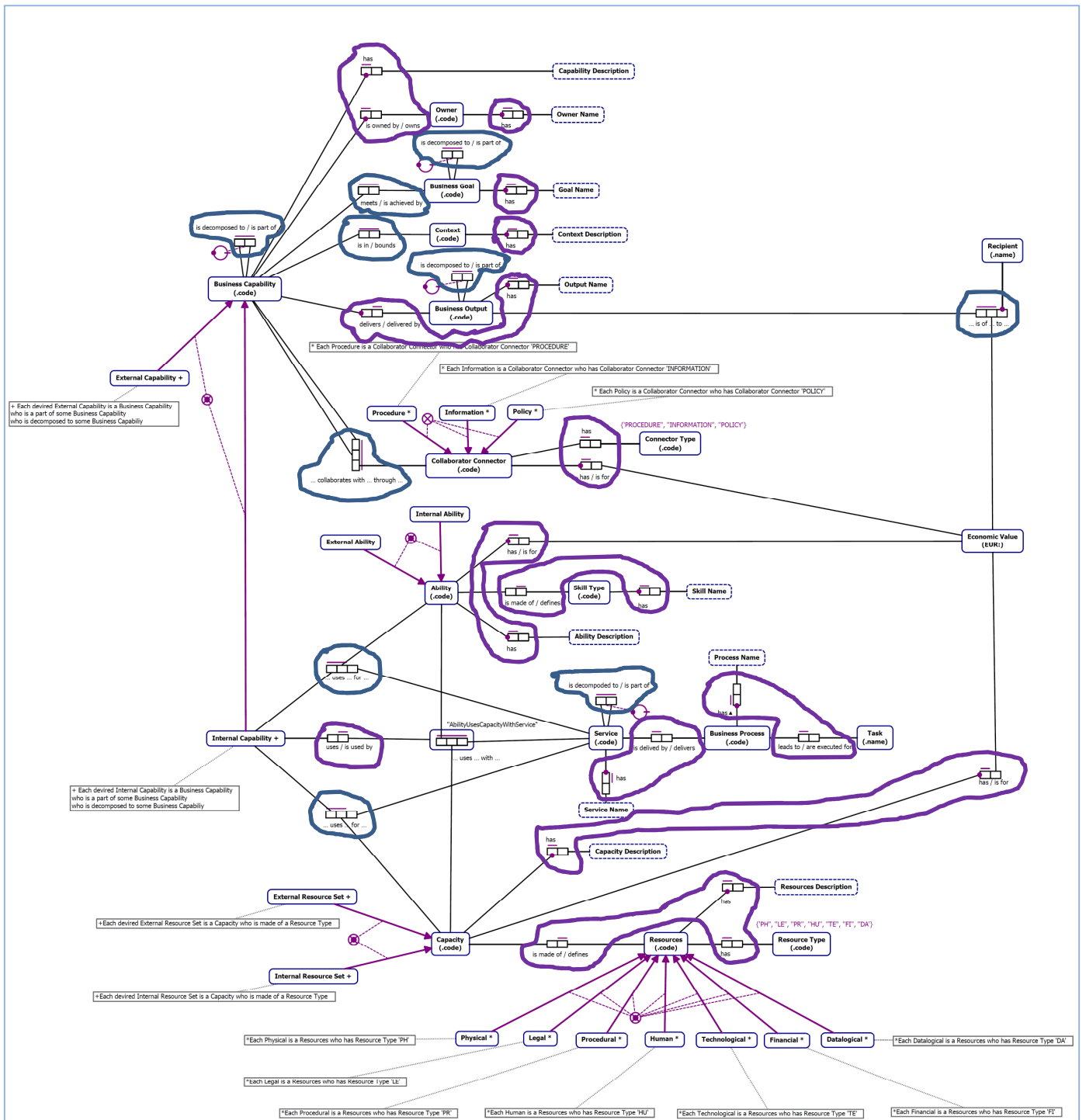
For the predicate “...uses...for...”, that refers to Capacity, we generate the table named “Usescapacityforservice”. Here we will use the “incapcode” to identify “Business Capability” as a primary and a foreign key. As far for “Capacity” we will use the “capaccode” as a primary key. The unique columns combination of capcode-capaccode is the candidate key. Thus we have the table:

Usescapacityforservice (incapcode, capaccode, ...)

For the predicate “...uses...for...”, that refers to Ability, we generate the table named “Usesabilityforservice”. Here we will use the “incapcode” to identify “Business Capability” as a primary and a foreign key. As far for “Ability” we will use the “abcode” as a primary key. The unique columns combination of capcode-abcode is the candidate key. Thus we have the table:

Usesabilityforservice (incapcode, abcode, ...)

In Step 2 we group functional fact types (fact types with functional roles) of the same object type together. In the Capability meta-model there no 1:1 cases so we are limited to the first procedure of step 2. According to that we are looking for object types that have at least one simple uniqueness constraint. To help visualize we place a lasso to them as follows:



Continuing the procedure of mapping, the object type “Owner” has a functional role with the value type “Owner Name”. Thus we generate the table named “Owner”, in which we group the value type “Owner Name” as column named “ownername”, which is mandatory. The object type “Business Owner” has reference mode “code”, so the primary key for this table will be the “ownercode”. Thus we have **the final table**:

Owner (ownercode, ownername)

As we see in the meta-model the object type “Business Capability” has a functional role with the object type “Owner” and the value type “Capability Description”. Thus we group the object type “Owner” as name of column (attribute) named “ownercode” both to tables “Incapability” and “Excapability”, since the last two shares a partition of “Business Capability” object type. The same exists in value type “Capability Description”, where we group it as a name of column named “capdesc” for both tables “Incapability” and “Excapability”. The column “ownercode” is mandatory since there is a foreign key although there is no mandatory constraint applied. Thus we do not enclose it in in *square brackets* ([]). We have already referred to “incapcode” and “excapcode” as primary keys, we just shown uniqueness constraint by doubly underlying, since there is a foreign key in that table. So the **final tables will be as follows**:

Incapability (incapcode, capdesc, ownercode)

Excapabiliy (excapcode, capdesc, ownercode)

The same as previous exists in object types “Business Goal” and “Context”. Thus we generate the tables “Businessgoal” and “Context” which **will be in final forms**:

Businessgoal (goalcode, goalname)

Context (contcode, contdesc)

As well the object type “Business Output” has a functional role with the object type “Business Capability” and with the value type “Output Name”. So we generate a table named “Output” and group the object type “Business Capability” with a name of column “capcode” to that table. Also we will use the reference mode “code” to identify “Business Output” as a primary key named “outputcode”. Here the primary key is doubly underlined because a foreign key exists. So we have **the final table**:

Output (outputcode, outputname, capcode)

The supertype “Collaborator Connector” has a functional role with the object type “Economic Value” and with the object type “Connector Type”. Thus we group the object type “Economic Value” as name of column named “evaluate” and the object type “Connector Type” as a name of column named “connectortype” with the values {“POLICY”, “INFORMATION”, “PROCEDURE”} to table “*Collaborator*”. Thus **the final table will be as follows:**

{POLICY,
INFORMATION,
PROCEDURE}
Collaborator (collabcode, connectortype, evaluate)

The supertype “Ability” has a functional role with the object type “Economic Value” and the value type “Ability Description”. Thus we group the object type “Economic Value” as name of column named “evaluate” and the value type “Ability Description” as name of column named “abdescr” both to tables “*Inability*” and “*Exability*”, since the last two shares a partition of “Ability” object type. Thus **the final tables will be as follows:**

Inability (inabcode, abdescr, evaluate)
Exability (exabcode, abdescr, evaluate)

The object type “Skill Type” has a functional role with the supertype “Ability” and the value type “Skill Name”. Thus we generate a table named “*Skilltype*” and group the supertype “Ability” and the value type “Skill Name” as columns to that table. Since there is the reference mode “code” in the “Skill Type” object type, **we generate the “skillcode” as the primary key that identify it.** Thus we have **the final table:**

Skilltype (skillcode, skillname, abcode)

The object type “Business Process” has a functional role with the object type “Tasks” and the value type “Process Name”. Thus, we generate a new table named “*Process*” and we group on him the object type “Tasks” as taskname and the value type “Process Name” as processname. Since there is the reference mode “code” in “Business Process” object type, **we generate the “processcode” as the primary key that identify it** and shown by underlying uniqueness constraints. The column task name is optional, thus we enclosed it in []. Thus **we have the final table:**

Process (processcode, processname, [taskname])

Also the object type “Service” has a functional role with the object type “Business Process” and the value type “Service Name”. Thus, we generate a new table named “Service” and we group on him as a foreign key the “processcode” of the object type “Business Process” and as attribute name “servname” the value type “Service Name”. Since there is the reference mode “code” in “Service” object type, ***we generate the “servcode” as the primary key that identify it*** and shown by underlying uniqueness constraints. The “servcode” primary key will be a foreign key for the tables “Usescapacityforservice” and “Usesabilityforservice”. Thus ***we have the following final tables:***

Service (servcode, servname, processcode)

Usescapacityforservice (incapcode, capaccode, servcode)

Usesabilityforservice (incapcode, abcode, servcode)

The object type “Capacity” has a functional role with the object type “Economic Value” and the value type “Capacity Description”. Thus we group the object type “Economic Value” as name of column named “evalue” and the value type “Capacity Description” as name of column named “capacdescr” both to tables “Incapacity” and “Excapacity”, since the last two shares a partition of “Capacity” object type. Thus ***the final tables will be as follows:***

Incapacity (incapaccode, capacdescr, evalue)


Excapacity (excapaccode, capacdescr, evalue)

The supertype “Resources” has a functional role with the supertype “Capacity”, with the value type “Resources Description” and the object type “Resource Type”. Thus we group the supertype “Capacity” named “capaccode” to table “Resources”, the value type “Resources Description” as name of column named “resdescription” and the object type “Resource Type” as a name of column named “restype”, which takes the values with the values {“PH”, “LE”, “PR”, “HU”, “TE”, “FI”, “DA”}. Since there is the reference mode “code” in “Resources” supertype, ***we generate the “rescode” as the primary key that identify it*** and shown by underlying uniqueness constraints. Thus ***the final table will be as follows:***

```
{“PH”, “LE”,  
  “PR”, “HU”,  
  “TE”, “FI”,  
  “DA”}
```

Resources (rescode, restype, resdescr, capaccode)

Finally the objectified object type “Abilityusescapacitywithservice” has a functional role with the subtype “Internal Capability”. Thus we group the subtype “Internal Capability” as name of column named “incapcode” to table “Abilityusescapacitywithservice”. Thus we have the table:

Abilityusescapacitywithservice (, incapcode)


Following Step 3 in the Capability meta-model there are no lazy objects types. So we left behind Step 3 and proceed to the next step. **In step 4** we unpack each “black box column” into its component attributes. Thus in the table “Abilityusescapacitywithservice” we do the following:

From  *Abilityusescapacitywithservice* (, incapcode)


To  *Abilityusescapacitywithservice* (abcode, capaccode, servcode, incapcode)

In Step 5 we map all other constraints and derivation rules. Also subtype constraints on functional roles map to qualified optional columns, and on non-functional roles map to qualified subset constraints.

In more detail let’s start with **the referential integrity constraints**. According to the Referential Integrity Rule each non-null vale of a foreign key must match one of the values of the referred primary key. As we referred to a previous section a dotted arrow (---->) is used to depict that starting from the foreign key to the primary key. Thus we depict the referential integrity constraints in our model for every foreign and primary key that we have already describe previously. In some case in our mode the referential integrity constraint followed by a text qualifications. Those text qualifications are shown in the relational schema with numbers 1, 2, 3, 4, 5, 6, 7, 8 and 9 and the “exactly where” clause means that is a mandatory role constraint and not an optional.

Next in cases of partition for subtypes we are reconstructing the supertype using unions and add the subtyping constraints. Thus for supertype “Business Capability” we add the partition constraint  (meaning that subtypes are mutually exclusive and collectively exhaustive), connecting with dotes lines between the primary keys “incapcode” &

“excapcode” that exists in the tables “*Excapability*” and “*Incapability*”. This constraint ensures that each supertype individual is maintained only in one table. We are also using the union “***BusinessCapability (capcode)= InternalCapability (incapcode) union ExternalCapability (excapcode)***”, as textual qualification for reconstructing the supertype “Business Capability”. The same exists to “Capacity” and “Ability” supertypes. We are adding a partition constraint between the primary keys “incapaccode” & “excapaccode” to the tables “*Incapacity*” and “*Exrcapacity*”, and also between the primary keys “inabcode” & “exabcode” to the tables “*Inability*” and “*Exability*”. Here the textual qualifications are “***Capacity (capaccode)= InternalResourceSet (incapaccode) union ExternalResourceSet (excapaccode)***” and “***Ability (abcode)= InternalAbility (inabcode) union ExternalAbility (exabcode)***”.

We also depicting the ring constraints by adding the  symbol connecting it with dots lines between the column names. This ring constraint is irreflexive, which means that the object type cannot bear a relationship with itself.

However in order the relational schema to be accurate and correct will we use the **Normalization** technique for fulfill that schema.

Firstly in the table Incapability, in the column incapcode there is confusion about the data that will be inserted. That’s because in this column we are inserting a hierarchy of Business Capability Codes, which include the Main Internal Business Capability Code and the Sub Internal Business Capability Code. So for that table ***we will create a new column named incaptype***, in which we will separate, by using the values {0,1} the two of them. Thus for the Main Internal Business Capability the incaptype will take the value 1 and for the Sub Internal Business Capability the incaptype will take the value 0.

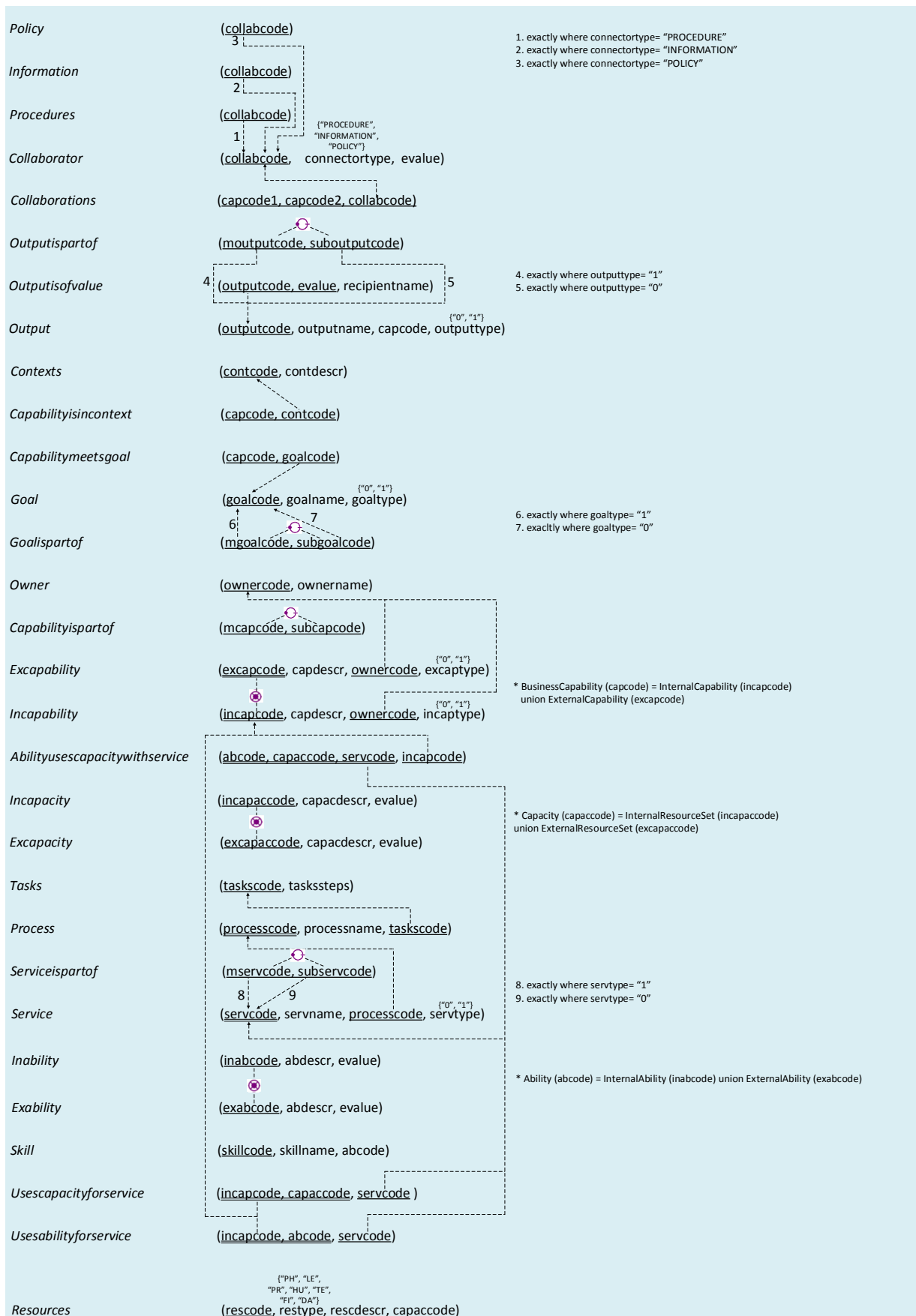
The same as previous exists for the tables Excapability, Goal, Output and Service. For the Table Excapability we in the column excapcode we are inserting a hierarchy of Business Capability Codes, which include the Main External Business Capability Code and the Sub External Business Capability Code. So for that table ***we will create a new column named excaptype***, which we take the values {0,1}. For the Main External Business Capability the excaptype will take the value 1 and for the Sub External Business Capability the excaptype will take the value 0. For the table Goals in the column goalcode we are inserting a hierarchy of Goal Codes, which include the Main Goal Code and the Sub Goal Code. So for that table ***we will create a new column named goaltype***, which we take the values {0,1}. For the Main Goal the goaltype will take the value 1 and for the Sub Goal the goaltype will take the value 0. For the table Output in the column outputcode we are inserting a hierarchy of Output

Codes, which include the Main Output Code and the Sub Output Code. So for that table ***we will create a new column named outputtype***, which we take the values {0,1}. For the Main Output the outputtype will take the value 1 and for the Sub Output the outputtype will take the value 0. Finally for the table Service in the column servcode we are inserting a hierarchy of Output Codes, which include the Main Service Code and the Sub Service Code. So for that table ***we will create a new column named servtype***, which we take the values {0,1}. For the Main Service the servtype will take the value 1 and for the Sub Service the servtype will take the value 0.

Also in order to be more clear the distinction about Internal Resource Set of Capacity and External Resource Set of Capacity, we have created in the table Incapacity ***a column named incapttype*** which will take the value {1} and in the table Excapacity ***a column named excapttype*** which will take the value {0}. We have done the same for the Internal Ability and External Ability. Here in the first case we have created ***a column named inabtype*** which will take the value {1} and in the second case ***a column named exabtype*** which will take the value {0}.

The previous actions are depicted in the following relational schema.

Relational Schema for Business Capability Database Management System



5.2 Chapter Summary

In this Chapter we have taken a Conceptual Schema of Business Capability as a background and then we have worked in a second level of analysis according to ORM, for designing a specific Relational Schema. This Schema is used for the designing of the maritime application, for the case study of the Danaos Management Consultant Company and will also be the basis for the creation of the physical database in the next Chapter.

Thus in this Chapter the specific procedure of mapping into a Relational Schema according ORM is described in detail, for this case of the new Conceptual Model about Business Capability. In more detail we have used a horizontal layout for depicting the schema and the graphical notation that is used according to ORM in this level. Then we have followed the specific procedure of mapping according to ([Halpin, 1995-A](#); [Halpin, 2001](#); [Halpin & Morgan, 2008](#)), in which have taking into account specific rules and strategies. The result of this procedure was the production of a Relational Schema that contains all the tables and the information about them. In other words we have a picture of the under development elements of the database for our application. In more detail in this schema we have depicted the name of the tables, the name of columns that contains, the candidate keys (primary keys, foreign keys) and the constraints that must implemented according to the Conceptual Schema (relationships between columns, unique columns, mandatory column, value constraints, subtyping constraints, ring constraints, textual constraints).

During the mapping procedure we have observed that some kind of information is not depicted accurate and correct for this UoD. This has leaded us to use the Normalization method in order to fulfill the schema. By this method we have inserted specific columns in some of the tables, which specify some extra information that must be stored in the database.

Taking into account all the work that has being done in this Chapter we can say that the procedure of Relational mapping is easily understood and standardized. This means that if we have created an accurate and complete model in the first level of ORM, then the procedure of mapping into a Relational Schema is easily implemented. Otherwise, if from the Conceptual Schema we are missing some of the required information, then the procedure of mapping is flexible enough to express this information in a wright way, by the Normalization method.

CHAPTER 6: Physical Database

Structure of this Chapter

- 6.1 DBMS Architecture
- 6.2 “BC” Physical Tables
- 6.3 Database Testing or Back-End Testing
- 6.4 Chapter Summary

This chapter deals with working in a third level of our Approach according to ORM, meaning the creation of the Physical Database that will be used for the maritime application of Danaos Management Consultant Company. Thus in Section 6.1 we present the reasons for choosing a specific DBMS, the Oracle, and also the Architecture that will be used. In Section 6.2 we presented all the objects that this database will have, meaning tables, sequences, views etc. and in Section 6.3 a database testing is implemented. Finally in Section 6.4 a summary of this chapter is presented.

6.1 DBMS Architecture

Since we have already created the relational schema for our DBMS, according to that in this chapter we will create the physical database design, meaning an SQL schema which includes the physical data types, keys, checks, indexes etc.

SQL (Structure Query Language) is a database computer language designed for managing data in relational database management systems. SQL is consisting of three other programming languages which are (www.zentut.com):

- The *Data Manipulation Language (DML)*, by which we can query and modify data.
- The *Data Definition Language (DDL)*, by which we can manage database objects such as tables, views, indexes etc.
- The *Data Control Language (DCL)*, by which we can grant or revoke privileges to users.

There are a lot of different databases Management Systems that use the SQL language with the most popular according to a DB-Engine ranking (www.db-engines.com) being: Oracle, MySQL, Microsoft SQL Server, MongoDB, PostgreSQL, DB2, Microsoft Access, Cassandra, SQLite, Redis and SAP Adaptive Server.

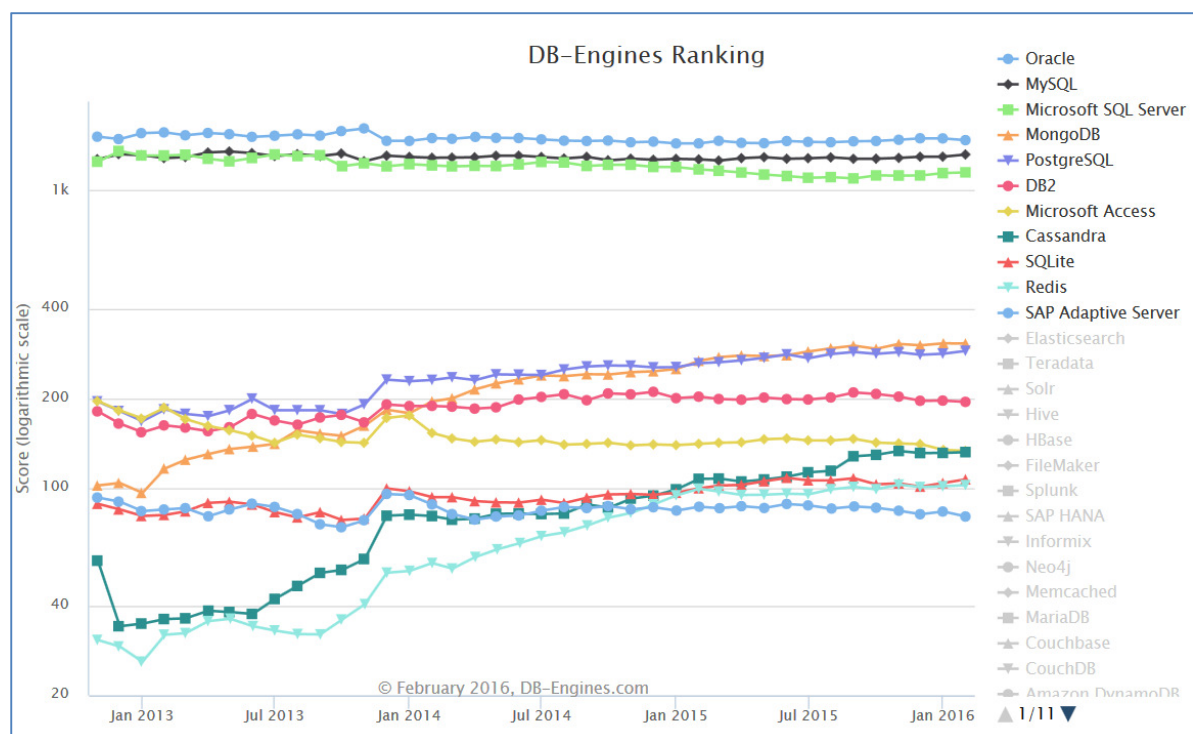


Figure 42: Most popular DBMS according to DB-Engines ranking (www.db-engines.com)

Some of the previous DBMS are commercial (e.g. Oracle, Microsoft SQL Server, DB2, Microsoft Access, SAP Adaptive Server etc.) and some other are open source (e.g MySQL, MongoDB, PostgreSQL, Cassandra, SQLite etc.), meaning that the source code is freely available and can be used and modified according to respective licenses (www.db-engines.com). Historically commercial DBMS are used widely as shown in the following diagram:

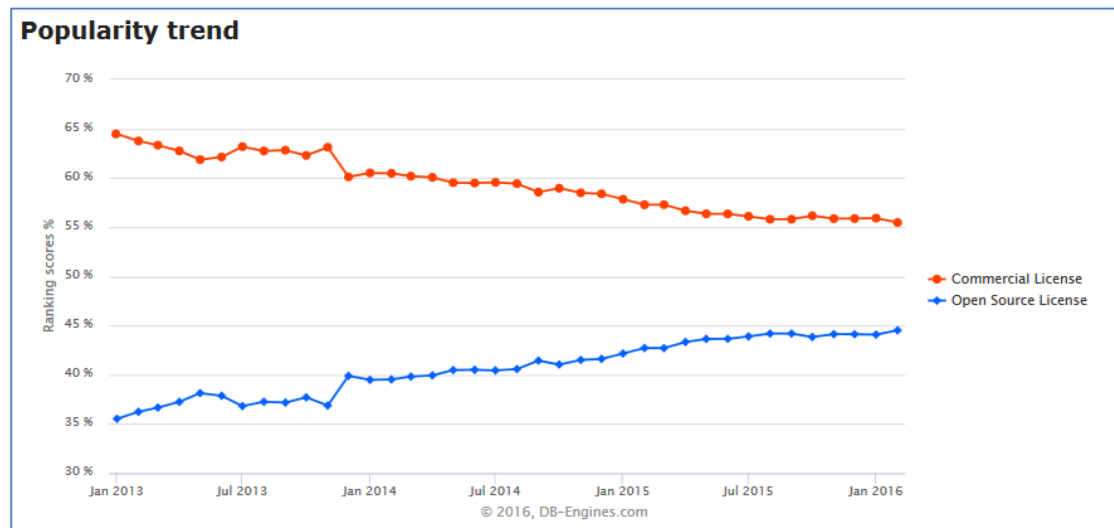


Figure 43: Popularity trend in DBMS (www.db-engines.com)

Since commercial DBMS are widely used instead of open-source DBMS, and also Oracle is the most popular database management system in this period, we will use it in order to create the physical database in this chapter and the interface in the next chapter.

Many oracle applications are built by using a client – server architecture or a multitier architecture. In the client – server application a database and its applications is divided into two parts: front – end or client side and back – end or server side, whereas the multitier is divided into three parts: the client, the application servers and the database servers ([Tickoo & Raina, 2010](#)).

For creating the Oracle database application for Business Capability **we will use a multitier architecture** (three-tier architecture) according to ([Tickoo & Raina, 2010](#)). In more detail we will have a database server, **the Oracle Database 11g**, in which the entire data will be stored. More especially it will contains the oracle data server files that will store tables, indexes and other database objects, and also the processes for request data of the application server for the client. Secondly we will have a client, the **Oracle PL/SQL**, who submit requests for an operation to be processed on the database server and interacts with the database server through one or more application servers. Worth mentioning that PL/SQL

is an Integrated Development Environment (IDE) for developing and storing programs in units in an Oracle Database (Oracle Corporation, April 2011). Finally we will have an Application Server, *the Oracle Forms 6i*, who is responsible for providing data access to the client, and it processes some queries and removes some of the loads from the database server. Also it serves as an interface between the client application and the database server.

6.2 “BC” Physical Tables

Taking into guidance from the relational schema designed in the previous chapter, we have created in the Oracle PL/SQL environment, a database for Business Capability named “BC”, which include the physical tables that depicted in that schema. Those tables contain the physical data types, keys, checks, indexes etc.

For creating the tables in the Oracle PL/SQL environment we have use specific statements of SQL. An SQL statement script of this work in given in [Appendix: SQL Script of BC Tables.](#)

A view of these tables follows thereafter:









▪ **Table INCAPABILITY:**

This table describes the Internal Business Capability of the Company. The incapcode is the primary key for that table. Except from the primary key constraint the table has one value constraint which says that the incaptype takes the values ‘0’ or ‘1’. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Incaptype_Value_Constraint
check (incaptype in ('0', '1'));
```

Also in the incaptype column we have given the default value 1, which means that when inserting data, the system automatically appear that value.

A view of this table follows thereafter:

INCAPABILITY				
Column	Type	Nullable	Default	Comments
 INCAPCODE	VARCHAR2(50)			Internal Capability Code
 CAPDESC	VARCHAR2(250)			Internal Capability Description
 OWNERCODE	VARCHAR2(50)			Internal Capability Owner
 INCAPTYPE	NUMBER(1)		1	Internal Capability Type: (1: Main, 0:Sub)
Key	Column(s)	Type		
 INCAPABILITY_PK	INCAPCODE	P		
 INCAPABILITY_FK	OWNERCODE	R		
 INCAPTYPE_VALUE_CONSTRAINT	INCAPTYPE	C		
Index	Column(s)	Type		
 INCAPABILITY_PK	INCAPCODE	unique		

▪ **Table EXCAPABILITY:**

This table describes the External Business Capability of the Company. Here the excapcode is the primary key for that table. Except from the primary key constraint the table has also one value constraint which says that the excaptype takes the values '0' or '1'. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Excaptype_Value_Constraint
check (excaptype in ('0', '1'));
```

Also in the excaptype column we have given the default value 1, which means that when inserting data, the system automatically appear that value.

A view of this table follows thereafter:






EXCAPABILITY				
Column	Type	Nullable	Default	Comments
EXCAPCODE	VARCHAR2(50)			External Capability Code
CAPDESC	VARCHAR2(250)			External Capability Description
OWNERCODE	VARCHAR2(50)			External Capability Owner
EXCAPTYPE	NUMBER(1)		1	External Capability Type: (1: Main, 0:Sub)
Key	Column(s)	Type		
EXCAPABILITY_PK	EXCAPCODE	P		
EXCAPABILITY_FK	OWNERCODE	R		
EXCAPTYPE_VALUE_CONSTRAINT	EXCAPTYPE	C		
Index	Column(s)	Type		
EXCAPABILITY_PK	EXCAPCODE	unique		

▪ **Table CAPABILITYISPARTOF:**

The table Capabilityispartof describes a hierarchy of capabilities into sub-capabilities. In order the population of data to be described correctly it is necessary to discuss the way we have implemented some of the constraints. In more detail except from the primary key constraint this table has an **irreflexive ring constraint**. As referred in (Halpin T. , UML data models from an ORM perspective: Part 7, 1999) irreflexivity maps to simple check clause which says that an object cannot bear a relationship with itself. Thus the SQL syntax for this kind of constraint will be:





```
constraint Capabilityispartof_RC
check (mcapcode<>subcapcode);
```

A view of this table follows thereafter:

CAPABILITYISPARTOF				
Column	Type	Nullable	Default	Comments
 MCAPCODE	VARCHAR2(50)			Top level Business Capability Code: Main Capabilities
 SUBCAPCODE	VARCHAR2(50)			Low level Business Capability Code: Sub Capabilities
Key	Column(s)	Type		
 CAPABILITYISPARTOF_PK	MCAPCODE, SUBCAPCODE	P		
 CAPABILITYISPARTOF_RC	SUBCAPCODE, MCAPCODE	C		
Index	Column(s)	Type		
 CAPABILITYISPARTOF_PK	MCAPCODE, SUBCAPCODE	unique		

- **Table OWNER:**

This table describes the Owners of the Company. The ownercode is the primary key for that table. Thus a view of this table follows thereafter:

OWNER				
Column	Type	Nullable	Default	Comments
 OWNERCODE	VARCHAR2(50)			Owner Code
 OWNERNAME	VARCHAR2(100)			Owner Name
Key	Column(s)	Type		
 OWNER_PK	OWNERCODE	P		
Index	Column(s)	Type		
 OWNER_PK	OWNERCODE	unique		







- **Table GOAL:**

This table describes the Goals of the Company. The goalcode is the primary key for that table. However except from the primary key constraint the table has one value constraint which says that the goalttype takes the values '0' or '1'. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Goalttype_Value_Constraint
check (goalttype in ('0', '1'));
```

Also in the goalttype column we have given the default value 1, which means that when inserting data, the system automatically appear that value.

A view of this table follows thereafter:

GOAL				
Column	Type	Nullable	Default	Comments
 GOALCODE	VARCHAR2(50)			Goal Code
 GOALNAME	VARCHAR2(250)			Goal Name
 GOALTYPE	NUMBER(1)		1	Goal Type: (1: Main, 0:Sub)
Key	Column(s)	Type		
 GOAL_PK	GOALCODE	P		
 GOALTYPE_VALUE_CONSTRAINT	GOALTYPE	C		
Index	Column(s)	Type		
 GOAL_PK	GOALCODE	unique		

- **Table GOALISPARTOF:**

The table Goalispartof describes a hierarchy of goals into sub-goals. This table except from the primary and foreign keys constraints has also an **irreflexive ring constraint**. Thus the SQL syntax for this kind of constraint will be:

constraint Goalispartof_RC
check (mgoalcode<>subgoalcode);

A view of this table follows thereafter:

GOALISPARTOF				
Column	Type	Nullable	Default	Comments
MGOALCODE	VARCHAR2(50)			Top level Business Goal Code: Strategic Goal Code
SUBGOALCODE	VARCHAR2(50)			Low level Business Goal Code: Operational Goal Code
Key	Column(s)	Type		
GOALISPARTOF_PK	MGOALCODE, SUBGOALCODE	P		
GOALISPARTOF_FK1	MGOALCODE	R		
GOALISPARTOF_FK2	SUBGOALCODE	R		
GOALISPARTOF_RC	SUBGOALCODE, MGOALCODE	C		
Index	Column(s)	Type		
GOALISPARTOF_PK	MGOALCODE, SUBGOALCODE	unique		

▪ **Table CAPABILITYMEETSGOAL:**

This table describes the relationship between Business Capabilities and Goals. The combination of capcode and goalcode is the primary key for that table. Thus a view of this table follows thereafter:

CAPABILITYMEETSGOAL				
Column	Type	Nullable	Default	Comments
CAPCODE	VARCHAR2(50)			Internal and External Capabilitiy Code
GOALCODE	VARCHAR2(50)			Goal Code: Strategic and Operational Goals Codes
Key	Column(s)	Type		
CAPABILITYMEETSGOAL_PK	CAPCODE, GOALCODE	P		
CAPABILITYMEETSGOAL_FK	GOALCODE	R		
Index	Column(s)	Type		
CAPABILITYMEETSGOAL_PK	CAPCODE, GOALCODE	unique		






▪ **Table CONTEXTS:**

This table describes the Context of the Company. The contcode is the primary key for that table. Thus a view of this table follows thereafter:

CONTEXTS				
Column	Type	Nullable	Default	Comments
CONTCODE	VARCHAR2(50)			Context Code
CONTDISC	VARCHAR2(150)			Context Description
Key	Column(s)	Type		
CONTEXTS_PK	CONTCODE	P		
Index	Column(s)	Type		
CONTEXTS_PK	CONTCODE	unique		

▪ **Table CAPABILITYISINCONTEXT:**

This table describes the relationship between Business Capabilities and Context. The combination of capcode and contcode is the primary key for that table. Thus a view of this table follows thereafter:

CAPABILITYISINCONTEXT				
Column	Type	Nullable	Default	Comments
 CAPCODE	VARCHAR2(50)			Internal and External Capabiltiy Code
 CONTCODE	VARCHAR2(50)			Context Code
Key	Column(s)	Type		
 CAPABILITYISINCONTEXT_PK	CAPCODE, CONTCODE	P		
 CAPABILITYISINCONTEXT_FK	CONTCODE	R		
Index	Column(s)	Type		
 CAPABILITYISINCONTEXT_PK	CAPCODE, CONTCODE	unique		








▪ **Table OUTPUT:**

This table describes the Output of the Company. The outputcode is the primary key for that table. However except from the primary key constraint the table has one value constraint which says that the outputtype takes the values '0' or '1'. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Outputtype_Value_Constraint
check (outputtype in ('0', '1'));
```

Also in the outputtype column we have given the default value 1, which means that when inserting data, the system automatically appear that value.

Thus a view of this table follows thereafter:

OUTPUT				
Column	Type	Nullable	Default	Comments
 OUTPUTCODE	VARCHAR2(50)			Output Code
 OUTPUTNAME	VARCHAR2(250)			Output Name
 CAPCODE	VARCHAR2(50)			Internal and External Capability Code
 OUTPUTTYPE	NUMBER(1)		1	Output Type: (1: Main, 0:Sub)
Key	Column(s)	Type		
 OUTPUT_PK	OUTPUTCODE	P		
 OUTPUTTYPE_VALUE_CONSTRAINT	OUTPUTTYPE	C		
Index	Column(s)	Type		
 OUTPUT_PK	OUTPUTCODE	unique		

▪ **Table OUTPUTISPARTOF:**

The table Outputispartof describes a hierarchy of outputs into sub-outputs. This table except from the primary key constraint has also an **irreflexive ring constraint**.

Thus the SQL syntax for this kind of constraint will be:

```
constraint Outputispartof_RC
check (moutputcode<>suboutputcode);
```

Thus a view of this table follows thereafter:

OUTPUTISPARTOF				
Column	Type	Nullable	Default	Comments
MOUTPUTCODE	VARCHAR2(50)			Top level Output Code: Main Output
SUBOUTPUTCODE	VARCHAR2(50)			Low level Output Code: Sub Output
Key	Column(s)	Type		
OUTPUTISPARTOF_PK	MOUTPUTCODE, SUBOUTPUTCODE	P		
OUTPUTISPARTOF_FK1	MOUTPUTCODE	R		
OUTPUTISPARTOF_FK2	SUBOUTPUTCODE	R		
OUTPUTISPARTOF_RC	SUBOUTPUTCODE, MOUTPUTCODE	C		
Index	Column(s)	Type		
OUTPUTISPARTOF_PK	MOUTPUTCODE, SUBOUTPUTCODE	unique		

▪ **Table OUTPUTISOFVALUE:**

This table describes the economic transactions according to the output of the company. The combination of outputcode and evalule is the primary key for that table. Thus a view of this table follows thereafter:

OUTPUTISOFVALUE				
Column	Type	Nullable	Default	Comments
OUTPUTCODE	VARCHAR2(50)			Output Code
EVALUE	NUMBER(20,2)			Output Economic Value
RECIPIENTNAME	VARCHAR2(250)			Recipient Name
Key	Column(s)	Type		
OUTPUTISOFVALUE_PK	OUTPUTCODE, EVALUE	P		
OUTPUTISOFVALUE_FK	OUTPUTCODE	R		
Index	Column(s)	Type		
OUTPUTISOFVALUE_PK	OUTPUTCODE, EVALUE	unique		

▪ **Table COLLABORATOR:**

This table describes the top element (supertype) in a structure of a hierarchy for subtyping. The collabcode is a primary key for that table and also a foreign key for the table Policy, Information and Procedure. Except from the primary key constraint that table has one value constraint and three qualification constraints. The value constraint says that the connectortype has the value 'POLICY' or 'INFORMATION' or 'PROCEDURE'. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Collaborator_Value_Constraint
check (connectortype in ('POLICY', 'INFORMATION', 'PROCEDURE'));
```

On the other hand the three qualifications declare:

- Qualification 1** declares that each value in the collabcode for Policy (policy.collabcode) must be a value of collabcode for Collaborator (collaborator.collabcode) for which the value of connectortype is 'POLICY'.
- Qualification 2** that each value in the collabcode for Information (information.collabcode) must be a value of collabcode for Collaborator

(collaborator.collabcode) for which the value of connectortype is 'INFORMATION'.

- c) **Qualification 3** declares that each value in the collabcode for Procedures (procedures.collabcode) must be a value of collabcode for Collaborator (collaborator.collabcode) for which the value of connectortype is 'PROCEDURE'.

As referred in (Halpin T. , 1995-B; Halpin T. , 2002) those qualification implemented with specific assertions instead of declaring a foreign key. Since assertions is not yet supported by the SQL system that we use, we will implemented those qualifications by generate alternative code. Thus in order the population of data to be inserting and deleting correctly for all those tables we have created the following *trigger named 'collaborator_TRG'*:

```
create or replace trigger collaborator_TRG
after insert or delete on collaborator
for each row
declare

begin
if INSERTING then

    if :new.connectortype = 'POLICY' then
        insert into POLICY (Collabcode) values (:new.collabcode);
    elsif :new.connectortype = 'INFORMATION' then
        insert into INFORMATION (Collabcode) values (:new.collabcode);
    elsif :new.connectortype = 'PROCEDURE' then
        insert into PROCEDURES (Collabcode) values (:new.collabcode);
    end if;

else
    if :old.connectortype = 'POLICY' then
        delete POLICY where Collabcode = :old.collabcode;
    elsif :old.connectortype = 'INFORMATION' then
        delete INFORMATION where Collabcode = :old.collabcode;
    elsif :old.connectortype = 'PROCEDURE' then
        delete PROCEDURES where Collabcode = :old.collabcode;
    end if;

end if;







end collaborator_TRG;
```

The trigger 'collaborator_TRG' is a procedure that runs automatically when a certain event occurs in the DBMS. Here we have the events of inserting and deleting, in a specific time (after the event) and a granularity which says that the event executed for each row. The action says that if inserting the system fill in

some values automatically referenced as new values (:new) and if deleting the system deleting some values automatically referenced as old values (:old). In practice this trigger do the following:





- If the user inserts connectortype 'POLICY' in the table Collaborator, then the system automatically inserts in the table Policy the value that has the collabcode in the table Collaborator. The same exists for tables Information and Procedures.
- If the user deletes connectortype 'POLICY' in the table Collaborator, then the system automatically deletes in the table Policy the value that has the collabcode in the table Collaborator. The same exists for tables Information and Procedures.

Thus a view of this table follows thereafter:

COLLABORATOR					
Column	Type	Nullable	Default	Comments	
 COLLABCODE	VARCHAR2(50)			Collaborator Connector Code	
 CONNECTORTYPE	VARCHAR2(20)			The type of Collaborator Connector	
 EVALUE	NUMBER(20,2)			The economic value of Collaborator Connector	
Key	Column(s)	Type			
 COLLABORATOR_PK	COLLABCODE	P			
 COLLABORATOR_VALUE_CONSTRAINT	CONNECTORTYPE	C			
Index	Column(s)	Type			
 COLLABORATOR_PK	COLLABCODE	unique			





▪ **Table POLICY:**

This table describes the Connector Type Policy. The collabcode is the primary key for that table. Thus a view of this table follows thereafter:

POLICY					
Column	Type	Nullable	Default	Comments	
 COLLABCODE	VARCHAR2(50)			Policy Collaborator Connector Code	
Key	Column(s)	Type			
 POLICY_PK	COLLABCODE	P			
 POLICY_FK	COLLABCODE	R			
Index	Column(s)	Type			
 POLICY_PK	COLLABCODE	unique			







▪ **Table INFORMATION:**

This table describes the Connector Type Information. The collabcode is the primary key for that table. Thus a view of this table follows thereafter:

INFORMATION					
Column	Type	Nullable	Default	Comments	
 COLLABCODE	VARCHAR2(50)			Information Collaborator Connector Code	
Key	Column(s)	Type			
 INFORMATION_PK	COLLABCODE	P			
 INFORMATION_FK	COLLABCODE	R			
Index	Column(s)	Type			
 INFORMATION_PK	COLLABCODE	unique			









▪ **Table PROCEDURES:**

This table describes the Connector Type Procedures. The collabcode is the primary key for that table. Thus a view of this table follows thereafter:

PROCEDURES				
Column	Type	Nullable	Default	Comments
 COLLABCODE	VARCHAR2(50)			Procedure Collaborator Connector Code
 Key	Column(s)	Type		
 PROCEDURES_PK	COLLABCODE	P		
 PROCEDURES_FK	COLLABCODE	R		
 Index	Column(s)	Type		
 PROCEDURES_PK	COLLABCODE	unique		

▪ **Table COLLABORATIONS:**

This table describes the relationship between Business Capabilities according to a Collaborator Connector. The combination of capcode1, capcode2 and collabcode is the primary key for that table. Thus a view of this table follows thereafter:










COLLABORATIONS				
Column	Type	Nullable	Default	Comments
 CAPCODE1	VARCHAR2(50)			Internal and External Capability Code
 CAPCODE2	VARCHAR2(50)			Internal and External Capability Code
 COLLABCODE	VARCHAR2(50)			Collaborator Connector Code
 Key	Column(s)	Type		
 COLLABORATIONS_PK	CAPCODE1, CAPCODE2, COLLABCODE	P		
 COLLABORATIONS_FK	COLLABCODE	R		
 Index	Column(s)	Type		
 COLLABORATIONS_PK	CAPCODE1, CAPCODE2, COLLABCODE	unique		

▪ **Table INABILITY:**

This table describes the Internal Ability of the Company. The inabcode is the primary key for that table. Except from the primary key constraint this table has a value constraint. This value constraint says that the inabtype takes only the value '1'. The SQL syntax for this kind of constraint will be:

```
constraint Inabtype_Value_Constraint
check (inabtype= '1');
```

Also in the inabtype column we have given the default value 1, which means that when inserting data, the system automatically appear that value. Thus a view of this table follows thereafter:








INABILITY				
Column	Type	Nullable	Default	Comments
 INABCODE	VARCHAR2(50)			Internal Ability Code
 ABDESCR	VARCHAR2(250)			Internal Ability Description
 EVALUE	NUMBER(20,2)			Internal Ability Economic Value
 INABTYPE	NUMBER(1)		1	Internal Ability Type: 1
 Key	Column(s)	Type		
 INABILITY_PK	INABCODE	P		
 INABTYPE_VALUE_CONSTRAINT	INABTYPE	C		
 Index	Column(s)	Type		
 INABILITY_PK	INABCODE	unique		

▪ **Table EXABILITY:**

This table describes the External Ability of the Company. The exabcode is the primary key for that table. Except from the primary key constraint this table has a value constraint. This value constraint says that the exabtype takes only the value '0'. The SQL syntax for this kind of constraint will be:






```
constraint Exabtype_Value_Constraint
check (excaptype = '0');
```

Also in the exabtype column we have given the default value 0, which means that when inserting data, the system automatically appear that value. Thus a view of this table follows thereafter:

EXABILITY				
Column	Type	Nullable	Default	Comments
 EXABCODE	VARCHAR2(50)			External Ability Code
 ABDESCR	VARCHAR2(250)			External Ability Description
 EVALUE	NUMBER(20,2)			External Ability Economic Value
 EXABTYPE	NUMBER(1)		0	External Ability Type= 0
Key	Column(s)	Type		
 EXABILITY_PK	EXABCODE	P		
 EXABTYPE_VALUE_CONSTRAINT	EXABTYPE	C		
Index	Column(s)	Type		
 EXABILITY_PK	EXABCODE	unique		

▪ **Table SKILL:**

This table describes the Skills that define External and Internal Ability. The skillcode is the primary key for that table. Thus a view of this table follows thereafter:

SKILL				
Column	Type	Nullable	Default	Comments
 SKILLCODE	VARCHAR2(50)			Skill Code
 SKILLNAME	VARCHAR2(250)			Skill Name
 ABCODE	VARCHAR2(50)	Y		Internal and External Ability Code
Key	Column(s)	Type		
 SKILL_PK	SKILLCODE	P		
Index	Column(s)	Type		
 SKILL_PK	SKILLCODE	unique		

▪ **Table INCAPACITY:**

This table describes the Internal Resource Set for Capacity of the Company. The incapaccode is the primary key for that table. Except from the primary key constraint this table has a value constraint. This value constraint says that the incapactype takes only the value '1'. The SQL syntax for this kind of constraint will be:

```
constraint Incapactype_Value_Constraint
check (incapactype = '1');
```

Also in the incapactype column we have given the default value 1, which means that when inserting data, the system automatically appear that value. Thus a view of this table follows thereafter:

INCAPACITY				
Column	Type	Nullable	Default	Comments
INCAPACCODE	VARCHAR2(50)			Internal Resource Set/Internal Capacity Code
CAPACDESCR	VARCHAR2(250)			Internal Resource Set/Internal Capacity Description
EVALUE	NUMBER(20,2)			Internal Resource Set/Internal Capacity Economic Value
INCAPACTYPE	NUMBER(1)		1	Incapacity Type= 1
Key	Column(s)	Type		
INCAPACITY_PK	INCAPACCODE	P		
INCAPACTYPE_VALUE_CONSTRAINT	INCAPACTYPE	C		
Index	Column(s)	Type		
INCAPACITY_PK	INCAPACCODE	unique		

- **Table EXCAPACITY:**

This table describes the External Resource Set for Capacity of the Company. The excapaccode is the primary key for that table. Except from the primary key constraint this table has a value constraint. This value constraint says that the incapactype takes only the value '0'. The SQL syntax for this kind of constraint will be:

```
constraint Excapactype_Value_Constraint
check (incapactype = '0');
```

Also in the incapactype column we have given the default value 0, which means that when inserting data, the system automatically appear that value. Thus a view of this table follows thereafter:

EXCAPACITY				
Column	Type	Nullable	Default	Comments
EXCAPACCODE	VARCHAR2(50)			External Resource Set/External Capacity Code
CAPACDESCR	VARCHAR2(250)			External Resource Set/External Capacity Description
EVALUE	NUMBER(20,2)			External Resource Set/External Capacity Economic Value
EXCAPACTYPE	NUMBER(1)		0	Excapacity Type= 0
Key	Column(s)	Type		
EXCAPACITY_PK	EXCAPACCODE	P		
EXCAPACTYPE_VALUE_CONSTRAINT	EXCAPACTYPE	C		
Index	Column(s)	Type		
EXCAPACITY_PK	EXCAPACCODE	unique		









- **Table SERVICE:**

This table describes the Services of the Company. The servcode is the primary key for that table. However except from the primary key constraint the table has one value constraint which says that the servtype takes the values '0' or '1'. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Servtype_Value_Constraint
check (servtype in ('0', '1'));
```

Also in the servtype column we have given the default value 1, which means that when inserting data, the system automatically appear that value.

Thus a view of this table follows thereafter:

SERVICE				
Column	Type	Nullable	Default	Comments
 SERVCODE	VARCHAR2(50)			Service Code
 SERVNAME	VARCHAR2(250)			Service Name
 PROCESSCODE	VARCHAR2(50)	Y		Business Process Code
 SERVTYPE	NUMBER(1)		1	Service Type: (1: Main, 0:Sub)
Key	Column(s)	Type		
 SERVICE_PK	SERVCODE	P		
 SERVICE_FK	PROCESSCODE	R		
 SERVTYPE_VALUE_CONSTRAINT	SERVTYPE	C		
Index	Column(s)	Type		
 SERVICE_PK	SERVCODE	unique		








■ **Table SERVICEISPARTOF:**

The table Serviceispartof describes a hierarchy of services into sub-services. This table except from the primary key constraint has also an **irreflexive ring constraint**.

Thus the SQL syntax for this kind of constraint will be:






```
constraint Serviceispartof_RC
check (mservcode<>subservcode);
```

Thus a view of this table follows thereafter:

SERVICEISPARTOF				
Column	Type	Nullable	Default	Comments
 MSERVCODE	VARCHAR2(50)			Top level Service Code: Main Service
 SUBSERVCODE	VARCHAR2(50)			Low level Service Code: Sub Service
Key	Column(s)	Type		
 SERVICEISPARTOF_PK	MSERVCODE, SUBSERVCODE	P		
 SERVICEISPARTOF_FK1	MSERVCODE	R		
 SERVICEISPARTOF_FK2	SUBSERVCODE	R		
 SERVICEISPARTOF_RC	SUBSERVCODE, MSERVCODE	C		
Index	Column(s)	Type		
 SERVICEISPARTOF_PK	MSERVCODE, SUBSERVCODE	unique		

■ **Table PROCESS:**

This table describes the Business Processes of the Company. The processcode is the primary key for that table. Thus a view of this table follows thereafter:

PROCESS				
Column	Type	Nullable	Default	Comments
 PROCESSCODE	VARCHAR2(50)			Business Process Code
 PROCESSNAME	VARCHAR2(250)			Business Process Name
 TASKNAME	VARCHAR2(1000)	Y		Tasks Name of Business Process
Key	Column(s)	Type		
 PROCESS_PK	PROCESSCODE	P		
Index	Column(s)	Type		
 PROCESS_PK	PROCESSCODE	unique		

- **Table RESOURCES:**

Before giving the view of table Resources, it is necessary to discuss the way we have implemented some of the constraints. In more detail except from the primary key constraint this table has one value constraint.

The value constraint says that the resource type code has the value 'PH' or 'LE' or 'PR' or 'HU' or 'TE' or 'FI' or 'DA'. As referred in (Halpin T. , 2001) check clauses are used to declare value constraints. Thus the SQL syntax for this kind of constraint will be:

```
constraint Resources_Value_Constraint
check (restype in ('PH', 'LE', 'PR', 'HU', 'TE', 'FI', 'DA'));
```

Thus a view of this table follows thereafter:

RESOURCES				
Column	Type	Nullable	Default	Comments
RESCODE	VARCHAR2(50)			Resource Code
RESTYPE	VARCHAR2(10)			Resource Type Values
REDESCR	VARCHAR2(1000)			Resources Description
CAPACCODE	VARCHAR2(50)	Y		Internal Resource Set/Internal Capacity and External Resource Set/Internal Capacity Code
Key	Column(s)	Type		
RESOURCES_PK	RESCODE	P		
RESOURCES_VALUE_CONSTRAINT	RESTYPE	C		
Index	Column(s)	Type		
RESOURCES_PK	RESCODE	unique		

- **Table USESCAPACITYFORSERVICE:**

This table describes the relationship between Internal Business Capability and Capacity in order to deliver a Service. The combination of incapcode and capaccocode is the primary key for that table. Thus a view of this table follows thereafter:

USESCAPACITYFORSERVICE				
Column	Type	Nullable	Default	Comments
INCAPCODE	VARCHAR2(50)			Internal Capability Code
CAPACCODE	VARCHAR2(50)			Internal Resource Set/Internal Capacity and External Resource Set/Internal Capacity Code
SERVCODE	VARCHAR2(50)			Service Code
Key	Column(s)	Type		
USESCAPACITYFORSERVICE_PK	INCAPCODE, CAPACCODE	P		
USESCAPACITYFORSERVICE_FK1	INCAPCODE	R		
USESCAPACITYFORSERVICE_FK2	SERVCODE	R		
Index	Column(s)	Type		
USESCAPACITYFORSERVICE_PK	INCAPCODE, CAPACCODE	unique		

- **Table USESABILITYFORSERVICE:**

This table describes the relationship between Internal Business Capability and Ability in order to deliver a Service. The combination of incapcode and abcode is the primary key for that table. Thus a view of this table follows thereafter:

USESABILITYFORSERVICE				
Column	Type	Nullable	Default	Comments
INCAPCODE	VARCHAR2(50)			Internal Capability Code
ABCODE	VARCHAR2(50)			Internal Ability and External Ability Code
SERVCODE	VARCHAR2(50)			Service Code
Key	Column(s)	Type		
USESABILITYFORSERVICE_PK	INCAPCODE, APCODE	P		
USESABILITYFORSERVICE_FK1	INCAPCODE	R		
USESABILITYFORSERVICE_FK2	SERVCODE	R		
Index	Column(s)	Type		
USESABILITYFORSERVICE_PK	INCAPCODE, APCODE	unique		

▪ **Table ABILITYUSESCAPACITYWITHSERVICE:**

This table describes the relationship between Internal Business Capability in combination with Capacity and Ability in order to deliver a Service. The combination of abcode, capacode and servcode is the primary key for that table. Thus a view of this table follows thereafter:

ABILITYUSESCAPACITYWITHSERVICE				
Column	Type	Nullable	Default	Comments
ABCODE	VARCHAR2(50)			Internal Ability and External Ability Code
CAPACODE	VARCHAR2(50)			Internal Resource Set/Internal Capacity and External Resource Set/Internal Capacity Code
SERVCODE	VARCHAR2(50)			Service Code
INCAPCODE	VARCHAR2(50)			Internal Capability Code
Key	Column(s)	Type		
ABUSESCAPWITHSERV_PK	ABCODE, CAPACODE, SERVCODE	P		
ABUSESCAPWITHSERV_FK1	INCAPCODE	R		
ABUSESCAPWITHSERV_FK2	SERVCODE	R		
Index	Column(s)	Type		
ABUSESCAPWITHSERV_PK	ABCODE, CAPACODE, SERVCODE	unique		

▪ **Table MENU:**

Except from the previous tables we have created a table for helping us creating a tree hierarchy, in the Main Page of our Application. This will be the Page for navigating between the different pages of the Application. Thus a view of this table follows thereafter:

MENU				
Column	Type	Nullable	Default	Comments
S_NUM	VARCHAR2(50)			Unique Identifier of Parent and Child Tree Nodes
NAME	VARCHAR2(250)	Y		Parent and Child Tree Nodes Names
MENU_NAME	VARCHAR2(250)	Y		Child Tree Node Values
Key	Column(s)	Type		
MENU_PK	S_NUM	P		
Index	Column(s)	Type		
MENU_PK	S_NUM	unique		

Except from the previous tables we have also **created a VIEW** named "FINAL_VIEW" to be used at the Application Level, in order to be able to execute queries for specific data about Business Capability. An SQL statement script of this VIEW is given in [Appendix: SQL Script of Total View](#).

Finally we have **created some database sequences** in order to help us to generate automatically unique primary keys in some cases of tables. Thus:

- For the table *Owner* we have created the sequence:

```
create sequence OWNER_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

- For the table *Skill* we have created the sequence:

```
create sequence SKILL_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

- For the table *Process* we have created the sequence:

```
create sequence PROCESS_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

- For the table *Contexts* we have created the sequence:

```
create sequence CONTEXTS_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

- For the table *Procedures* we have created the sequence:

```
create sequence PROCEDURE_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

- For the table *Information* we have created the sequence:

```
create sequence INFORMATION_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

- For the table *Policy* we have created the sequence:

```
create sequence SKILL_SEQ  
minvalue 1  
maxvalue 99999  
start with 1  
increment by 1  
cache 20;
```

6.3 Database Testing or Back-End Testing

Now that we have created the physical database it is important to test it. As stated by [Chang & Cheung \(1999\)](#) “testing of database application is of great importance in both the development production phase, since undetected faults in this application may result in incorrect modification or accidental removal of crucial data”.

By Database Testing we are testing the back-end components, which are not visible to users and by that, we intent to insure (www.tutorialspoint.com):

- Data validation
- Data integrity
- Performance check to database
- Testing of Procedures, Triggers and Functions

Also in order to guarantee that the database transactions are processed concurrently, we must satisfy all the ACID properties which are (www.softwaretestinghelp.com):

- **Atomicity:** Describes that a transaction either falls or passes. In other words if a single part of transaction fails then the entire transaction has failed
- **Consistence:** Describes that a transaction will always result in a valid state o database.
- **Isolation:** Describes that if there are multiple transactions and they are executed all at once, the result of database should be the same as if they were executed one after the other.
- **Durability:** Describes that once a transaction is done and committed, no external factor like power loss or crash should be able to chance it.



The Process of Database Testing has specific steps which are shown in the following Figure:

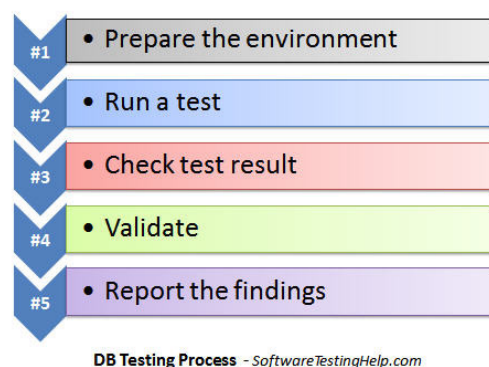


Figure 44: Database Testing Process (www.softwaretestinghelp.com)

In first step we insert in our database a sample of data in order the testing to be achieved. For the purpose of this testing we *will use examples of data by a real company from the maritime domain field, the Danaos Maritime Consultant (DMC)*, which will be described in more detail in chapter 8.

The next step deals with the implementation of a specific category for Database Testing. In general Database Testing can be categorized into three categories (www.tutorialspoint.com):

- **Structural Database Testing:** This category deals with the procedure of testing tables and columns, of testing the schema, of testing stored procedures, views and triggers etc.
- **Functional Testing:** This category involves checking functionality of database from user point of view, with the most common types the White Box and the Black Box testing
- **Nonfunctional Testing:** This final category deals with the performance of the database and involves load-testing, risk testing, stress testing and minimum system requirements.

According to (www.onestopsoftwaretesting.com) the most effective methods are those of Structural testing and Functional testing. Thus taking into account the directions that are given for those methods by (www.onestopsoftwaretesting.com), we will follow the same for our database system.

We will first begin by implementing the **Structural Testing** Method. By this method the tests will verify each and every object in a type of structure. Thus we will make a Database Schema Testing and a Trigger Testing, since there is no stored Procedure in our database.

- i. The **Database Schema Testing** contains:
 - a) Tests in Databases and Devices and more especially in:
 - Database Name
 - Data device, log device and dump device
 - The existence of enough space allocated for the database
 - Database option setting (i.e. trunc option)

From the above four we will test the first and the third. Thus if we take the following view:

We can see that in the option General the Name is written correctly, meaning “BC”. Also there is enough storage allocated for the database. Thus we continue with the next step.

- b) Tests in Tables, Columns, Columns Types, Defaults and Rules. In this step we will try to find out differences between the relational schema and the actual tables. First guiding by the relational schema ***we will check at least once all the Table Names and the Column Names for each table.*** Then we will do the same for the Column Types and we will also test whether the column is null or not. An example of the described test is given for the table Incapability. Thus in the relational schema we have the table:

{'0','1'}

Incapability (incapcode, capdescr, ownercode, incaptype)

Taking a view of this table from the physical database we check as shown in the following picture if there is a difference with the relational schema in the Name.

Since it is correct we then proceed in checking the column names. As we see in the view of Columns, the column names is the same with those of relational

schema. Also there is small enough to describe a query. So we then proceed on checking the value types. In more detail the column name incapcode is a varchar2(50) data type, meaning a variable length string with maximum size 50. This is an efficient length for describing the Internal Capability Code and the appropriate data type since a code may contain number and characters. The same exists for the other column names, meaning capdesc, ownercode and incaptype. Finally we are checking whether the column names are null or not. In the relational schema all columns are mandatory (they are not enclosed in []). This means that the columns must be null. Thus as we seeing in the view none of them is chosen as nullable, meaning that are mandatory.

General Columns Keys Checks Indexes Privileges						
Type owner	Name					
Name	Virtual	Type	Nullable	Default/Expr.	Storage	Comments
▶ INCAPCODE	<input type="checkbox"/>	VARCHAR2(50)	<input type="checkbox"/>	...		Internal Capability Code
CAPDESC	<input type="checkbox"/>	VARCHAR2(250)	<input type="checkbox"/>	...		Internal Capability Description
OWNERCODE	<input type="checkbox"/>	VARCHAR2(50)	<input type="checkbox"/>	...		Internal Capability Owner
INCAPTYPE	<input type="checkbox"/>	NUMBER(1)	<input type="checkbox"/>	1	...	Internal Capability Type: (1: Main, 0:Sub)
*	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	

Thus if we go to the table Incapability, we will see that it contains those four column names: incapcode, capdesc, ownercode and incaptype:

SQL Output Statistics						
<pre>select * from Incapability;</pre>						
	INCAPCODE	CAPDESC	OWNERCODE	INCAPTYPE		
▶ 1	INCAP1.1	Ship Financial Management	OWN1	0		
2	INCAP1.2	Ship Technical Management	OWN1	0		
3	INCAP1.3	Ship Procurement Management	OWN1	0		
4	INCAP1.4	International Safety Management	OWN1	0		
5	INCAP1.5	Human Resource Management	OWN1	0		
6	INCAP1.6	Chartering Management	OWN1	0		
7	INCAP1.7	Operation Management	OWN1	0		
8	INCAP2	Social Networking Capability	OWN1	1		
9	INCAP2.1	Secure Transactions and Communications Capability	OWN1	0		
10	INCAP2.2	Marketing Capability	OWN1	0		
11	INCAP3	Information Store & Management Capability	OWN1	1		
12	INCAP3.1	Information Storing	OWN1	0		
13	INCAP3.2	Information Management	OWN1	0		
14	INCAP1	Maritime Management Capability	OWN1	1		
15	INCAP4	Maritime Compliance Capability	OWN1	1		
16	INCAP4.1	Vessel Monitoring Capability	OWN1	0		
17	INCAP4.2	Port Regulation Monitoring Capability	OWN1	0		
18	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	0		

Thus the check in columns is done. Then the same test is continuing for the other tables.

Now ***we are going to test rules definition and whether a rule is bound to correct table columns.*** This rules deals with testing the checks constraints that has implemented and concerns value constraints and ring constraints.

In our database we have five value constraints in the tables Collaborator, Resources, Incapability, Exc capability, Goal, Output and Service. Thus ***we are going to check these value constraints.***

Firstly we are going to the table Collaborator and we are checking the name of the check, if the condition is written correctly and if it is enabled or not as follows:

General Columns Keys Checks Indexes Privileges						
Name	Condition	Enabled	Deferrable	Deferred	Last change	
COLLABORATOR_VALUE_CONSTRAINT	connectortype in ('POLICY','INFORMATION','PROCEDURE')	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:22 μμ	
*		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

Here the condition says that connectortype column must have the values Policy or Information of Procedure, which is correct. We then go to edit data in that table and during the inserting of a new instance we can see that when choosing the column connectortype the three values appear as follows:

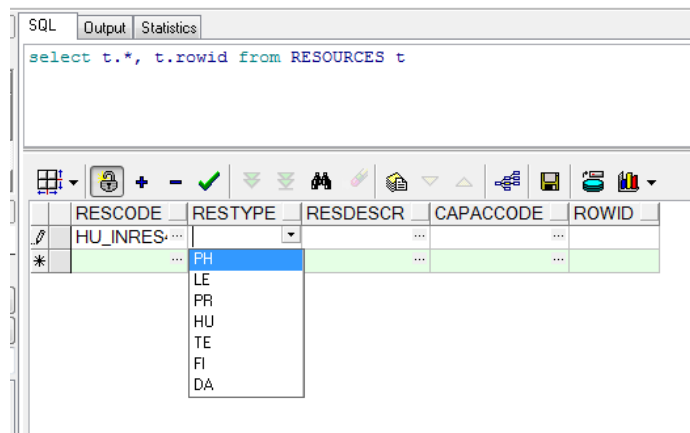
SQL Output Statistics				
select t.*, t.rowid from COLLABORATOR t				
	COLLABCODE	CONNECTORTYPE	EVALUE	ROWID
1	PR1	PROCEDURE	10000,00	AAASNpAAEAAAAKrAAB
2	IN1	INFORMATION	15000,00	AAASNpAAEAAAAKrAAC
	PO1			
		POLICY		
		INFORMATION		
		PROCEDURE		

Thus the value check for that table is done.

We are now doing the same in the table Resources. We are going in the Check Page and checking the name of the check, if the condition is written correctly and if it is enabled or not as follows:

General Columns Keys Checks Indexes Privileges						
Name	Condition	Enabled	Deferrable	Deferred	Last change	
RESOURCES_VALUE_CONSTRAINT	restype in ('PH','LE','PR','HU','TE','FI','DA')	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:24 μμ	

Here the condition says that restype column must have the values PH or LE or PR or HU or TE or FI or DA, which is correct. We then go to edit data in that table during the inserting of a new instance we can we see that when choosing the column restype the seven values appear as follows:



Thus the value check for that table is done.

We continue with the table Incapability. We are going in the Check Page and checking the name of the check, if the condition is written correctly and if it is enabled or not as follows:

Name	Condition	Enabled	Deferrable	Deferred	Last change
INCAPTYPE_VALUE_CONSTRAINT	incaptype in (0,1)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	6/4/2016 3:11:57 μμ

Here the condition says that the incaptype column must have the values 0 or 1, which is correct. We then go to edit data in that table during the inserting of a new instance we can see that when choosing the column incaptype the two values appear as follows:

INCAPCODE	CAPDESC	OWNERCODE	INCAPTYPE	ROWID
1 INCAP1.1	Ship Financial Management	OWN1	0	AAASNRAEAAAAKbAAA
2 INCAP1.2	Ship Technical Management	OWN1	0	AAASNRAEAAAAKbAAB
3 INCAP1.3	Ship Procurement Management	OWN1	0	AAASNRAEAAAAKbAAC
4 INCAP1.4	International Safety Management	OWN1	0	AAASNRAEAAAAKbAAD
5 INCAP1.5	Human Resource Management	OWN1	0	AAASNRAEAAAAKbAAE
6 INCAP1.6	Chartering Management	OWN1	0	AAASNRAEAAAAKbAAF
7 INCAP1.7	Operation Management	OWN1	0	AAASNRAEAAAAKbAAG
8 INCAP2	Social Networking Capability	OWN1	1	AAASNRAEAAAAKbAAH
9 INCAP2.1	Secure Transactions and Communications Capability	OWN1	0	AAASNRAEAAAAKbAAI
10 INCAP2.2	Marketing Capability	OWN1	0	AAASNRAEAAAAKbAAJ
11 INCAP3	Information Store & Management Capability	OWN1	1	AAASNRAEAAAAKbAAK
12 INCAP3.1	Information Storing	OWN1	0	AAASNRAEAAAAKbAAL
13 INCAP3.2	Information Management	OWN1	0	AAASNRAEAAAAKbAAM
14 INCAP1	Maritime Management Capability	OWN1	1	AAASNRAEAAAAKcAAA
15 INCAP4	Maritime Compliance Capability	OWN1	1	AAASNRAEAAAAKtAAA
16 INCAP4.1	Vessel Monitoring Capability	OWN1	0	AAASNRAEAAAAKtAAB
17 INCAP4.2	Port Regulation Monitoring Capability	OWN1	0	AAASNRAEAAAAKtAAC
18 INCAP4.3	Regulation Inconsistencies Reporting Capability	OWN1	0	AAASNRAEAAAAKtAAD
19 INCAP4.5	Regulation Provision Capability	OWN1	0	

Thus the value check for that table is done. The same results of testing exist for the tables Excapability, Goal, Output and Service.

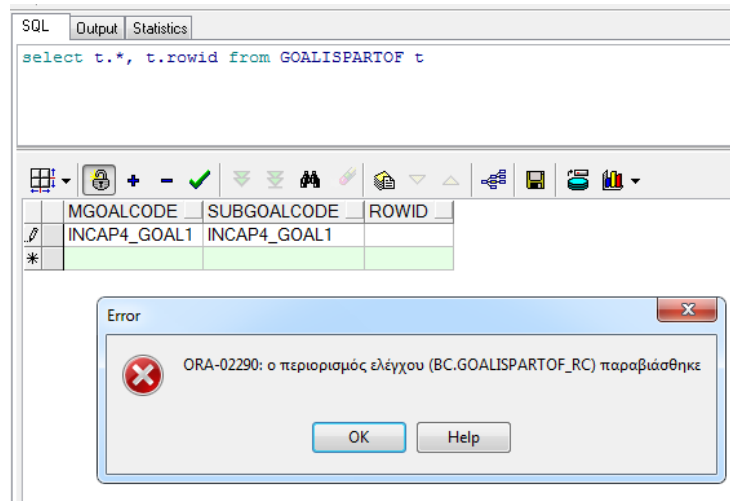
We then continue to test the ring constraints. In our database we have some ring constraints in the tables Capabilityispartof, Goalispartof, Outputispartof and Serviceispartof.

Name	Condition	Enabled	Deferrable	Deferred	Last change
► CAPABILITYSPARTOF_RC	mcapcode<>subcapcode	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:21 μμ

The screenshot shows the SQL Developer interface. The top bar has tabs for 'SQL', 'Output', and 'Statistics'. The SQL editor contains the query: `select t.*, t.rowid from CAPABILITYISPARTOF t`. Below the editor is a toolbar with various icons. A table grid is visible, showing columns 'MCAPCODE', 'SUBCAPCODE', and 'ROWID'. The first row has values 'INCAP1', 'INCAP1', and an empty cell. An error dialog box is open in the foreground, titled 'Error'. It contains a red 'X' icon and the message: 'ORA-02290: ο περιορισμός ελέγχου (BC.CAPABILITYISPARTOF_RC) παραβιάσθηκε'. At the bottom of the dialog are 'OK' and 'Help' buttons.

Name	Condition	Enabled	Deferrable	Deferred	Last change
▶ GOALISPARTOF_RC	mgoalcode<>subgoalcode	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:22 μμ
*		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

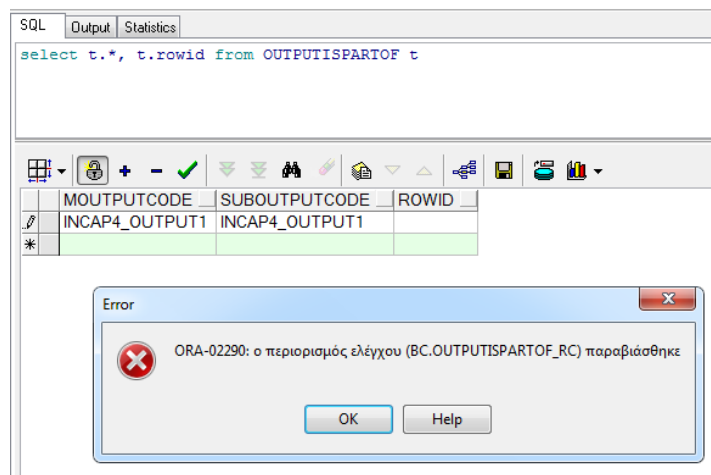
143



We then continue with the table Outputispartof. Thus we are going in the Check Page to check the name of the ring constraint, if the condition is correct or not and if is enabled or not as follows:

General Columns Keys Checks Indexes Privileges						
Name	Condition	Enabled	Deferrable	Deferred	Last change	
OUTPUTISPARTOF_RC	moutputcode<>suboutputcode	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:22 μμ	
*		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

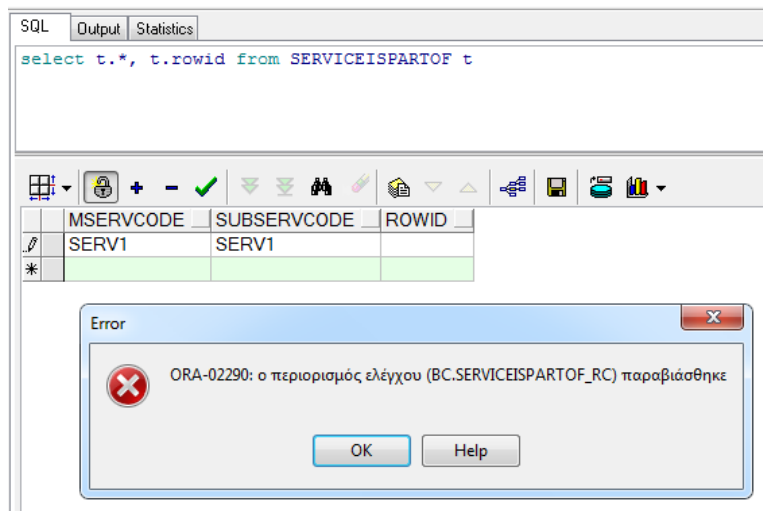
The condition says that the values of an instance with columns moutputcode and suboutputcode is not equal, which is correct. We then go to edit data in that table and we observe that if we try to insert an instance in which the values of moutputcode and suboutputcode is the same, then an error message appears as follows:



Finally we are doing the same for the table Serviceispartof. Thus we are going in the Check Page to check the name of the ring constraint, if the condition is correct or not and if is enabled or not as follows:

General Columns Keys Checks Indexes Privileges						
Name	Condition	Enabled	Deferrable	Deferred	Last change	
▶ SERVICEISPARTOF_RC	mservcode<>subservcode	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:24 μμ	
*		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

The condition says that the values of an instance with columns mservcode and subservcode is not equal, which is correct. We then go to edit data in that table and we observe that if we try to insert an instance in which the values of mservcode and subservcode is the same, then an error message appears as follows:



Worth mentioning that in this step we also checking:

- ✓ Default Definitions
- ✓ Whether a default is bound to correct table columns
- ✓ Whether access privileges are granted to correct groups.

However in our database we have none of the three of them to be tested.

c) Tests in Keys and Indexes. In this step we are checking:

- Primary keys for each table (every table must have a primary key)
- Foreign keys
- Column data types between a foreign key column and a column in other table
- Indices, clustered or nonclustered; unique or not unique

Thus taking the example of table Incapability from the relational schema; *Incapability* (incapcode, capdescr, ownercode), we observing that the incapcode is the primary key in that table (since is doubly underlying) and the ownercode is a foreign key. Now taking the following view of that table from the physical

database as follows, we see that we have created a primary key named Incapability_PK which refers to column incapcode and is enabled.

Name	Type	Columns	Enabled	Referencing table	Referencing columns	On Delete	Deferrable	Deferred	Last change
INCAPABILITY_PK	Primary	INCAPCODE	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:20 μμ
INCAPABILITY_FK	Foreign	OWNERCODE	<input checked="" type="checkbox"/>	OWNER	OWNERCODE	No action	<input type="checkbox"/>	<input type="checkbox"/>	22/3/2016 6:57:21 μμ














The primary key must be unique in each and every row in that table. Thus if we go to the table Incapability and execute a query, from the results we observe that the incapcode for each and every column is unique:

SQL

Output

Statistics

select * from Incapability;

	INCAPCODE	CAPDESC	OWNERCODE	INCAPTYPE
1	INCAP1.1	Ship Financial Management	OWN1	0
2	INCAP1.2	Ship Technical Management	OWN1	0
3	INCAP1.3	Ship Procurement Management	OWN1	0
4	INCAP1.4	International Safety Management	OWN1	0
5	INCAP1.5	Human Resource Management	OWN1	0
6	INCAP1.6	Chartering Management	OWN1	0
7	INCAP1.7	Operation Management	OWN1	0
8	INCAP2	Social Networking Capability	OWN1	1
9	INCAP2.1	Secure Transactions and Communications Capability	OWN1	0
10	INCAP2.2	Marketing Capability	OWN1	0
11	INCAP3	Information Store & Management Capability	OWN1	1
12	INCAP3.1	Information Storing	OWN1	0
13	INCAP3.2	Information Management	OWN1	0
14	INCAP1	Maritime Management Capability	OWN1	1
15	INCAP4	Maritime Compliance Capability	OWN1	1
16	INCAP4.1	Vessel Monitoring Capability	OWN1	0
17	INCAP4.2	Port Regulation Monitoring Capability	OWN1	0
18	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	0

If we try to insert a data that is the same with some other in that column, for example INCAP4.3 code twice, then the following error appears, which is correct:

SQL

Output

Statistics

select t.*, t.rowid from INCAPABILITY t

	INCAPCODE	CAPDESC	OWNERCODE	INCAPTYPE	ROWID
1	INCAP1.1	Ship Financial Management	OWN1	0	AAASNRAAEAAAAKbAAA
2	INCAP1.2	Ship Technical Management	OWN1	0	AAASNRAAEAAAAKbAAB
3	INCAP1.3	Ship Procurement Management	OWN1	0	AAASNRAAEAAAAKbAAC
4	INCAP1.4	International Safety Management	OWN1	0	AAASNRAAEAAAAKbAAD
5	INCAP1.5	Human Resource Management	OWN1	0	AAASNRAAEAAAAKbAAE
6	INCAP1.6	Chartering Management	OWN1	0	AAASNRAAEAAAAKbAAF
7	INCAP1.7	Operation Management	OWN1	0	AAASNRAAEAAAAKbAAG
8	INCAP2	Social Networking Capability	OWN1	1	AAASNRAAEAAAAKbAAH
9	INCAP2.1	Secure Transactions and Communications Capability	OWN1	0	AAASNRAAEAAAAKbAAI
10	INCAP2.2	Marketing Capability	OWN1	0	AAASNRAAEAAAAKbAAJ
11	INCAP3	Information Store & Management Capability	OWN1	1	AAASNRAAEAAAAKbAAK
12	INCAP3.1	Information Storing	OWN1	0	AAASNRAAEAAAAKbAAL
13	INCAP3.2	Information Management	OWN1	0	AAASNRAAEAAAAKbAAM
14	INCAP1	Maritime Management Capability	OWN1	1	AAASNRAAEAAAAKbAAA
15	INCAP4	Maritime Compliance Capability	OWN1	1	AAASNRAAEAAAAKbAAB
16	INCAP4.1	Vessel Monitoring Capability	OWN1	0	AAASNRAAEAAAAKbAAC
17	INCAP4.2	Port Regulation Monitoring Capability	OWN1	0	AAASNRAAEAAAAKbAAD
18	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	0	AAASNRAAEAAAAKbAAE
*	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	0	AAASNRAAEAAAAKbAAF

ORA-00001: παραβιάστηκε περιορισμός μοναδικότητας (BC.INCAPABILITY_PK)

OK

Help

Thus the check for primary keys is done.

Also in that view we observe that we have created a foreign key name Incapability_FK which refers to the column ownercode and references in the table owner, and more specific in the column ownercode of that table. Thereafter we check if the column data types between the foreign key Incapability_FK and the column ownercode in the table Owner are the same. The column ownercode in the table Incapability is varchar2(50). The same exist in the column ownercode in the table Owner as we see in the following view:

Name	Virtual	Type	Nullable	Default/Expr.	Storage	Comments
OWNERCODE	<input type="checkbox"/>	VARCHAR2(50)	<input type="checkbox"/>			Owner Code
OWNERNAME	<input type="checkbox"/>	VARCHAR2(100)	<input type="checkbox"/>			Owner Name

If we go to insert data in the table Incapability, we will see that when choosing the column ownercode then the data comes up automatically:

SQL Output Statistics

```
select t.*, t.rowid from INCAPABILITY t
```

	INCAPCODE	CAPDESC	OWNERCODE	INCAPTYPE	ROWID
1	INCAP1.1	Ship Financial Management	OWN1	0	AAASNRAEAAAAKbAAA
2	INCAP1.2	Ship Technical Management	OWN1	0	AAASNRAEAAAAKbAAB
3	INCAP1.3	Ship Procurement Management	OWN1	0	AAASNRAEAAAAKbAAC
4	INCAP1.4	International Safety Management	OWN1	0	AAASNRAEAAAAKbAAD
5	INCAP1.5	Human Resource Management	OWN1	0	AAASNRAEAAAAKbAAE
6	INCAP1.6	Chartering Management	OWN1	0	AAASNRAEAAAAKbAAF
7	INCAP1.7	Operation Management	OWN1	0	AAASNRAEAAAAKbAAG
8	INCAP2	Social Networking Capability	OWN1	1	AAASNRAEAAAAKbAAH
9	INCAP2.1	Secure Transactions and Communications Capability	OWN1	0	AAASNRAEAAAAKbAAI
10	INCAP2.2	Marketing Capability	OWN1	0	AAASNRAEAAAAKbAAJ
11	INCAP3	Information Store & Management Capability	OWN1	1	AAASNRAEAAAAKbAAK
12	INCAP3.1	Information Storing	OWN1	0	AAASNRAEAAAAKbAAL
13	INCAP3.2	Information Management	OWN1	0	AAASNRAEAAAAKbAAM
14	INCAP4	Maritime Management Capability	OWN1	1	AAASNRAEAAAAKbAAA
15	INCAP4	Maritime Compliance Capability	OWN1	1	AAASNRAEAAAAKbAAA
16	INCAP4.1	Vessel Monitoring Capability	OWN1	0	AAASNRAEAAAAKbAAB
17	INCAP4.2	Port Regulation Monitoring Capability	OWN1	0	AAASNRAEAAAAKbAAC
18	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	0	AAASNRAEAAAAKbAAD
19	INCAP4.4	Regulations Provision Capability	OWN1	0	AAASNRAEAAAAKbAAE

OWNERCODE dropdown values:
 DWN1 DMC
 DWN2 ComSys
 DWN3 Microsoft
 DWN4 Danaos Service/India

Thus the check for the foreign keys is done.

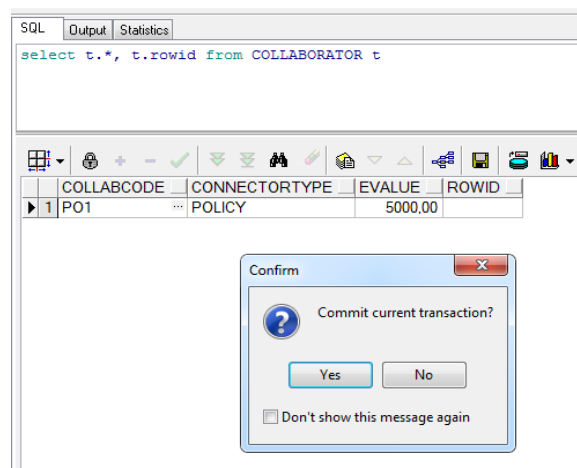
Finally we check the indexes for that table. For every primary key that we have created the PL/SQL creates automatically indexes for that key, since the values must be retrieval more speedier in a query. As we see in the following view in the tab indexes the system has created a unique index for the column incapcode. Thus the check of indexes is done.

Owner	Name	Type	Columns	Compress	Prefix length	Reverse	Local	Storage
BC	INCAPABILITY_PK	Unique	INCAPCODE	<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>	tablespace users pctfree 10 initrans 2 maxtrans 255 storage (initial 64k next 1m minextents 1 maxextents unlimited)

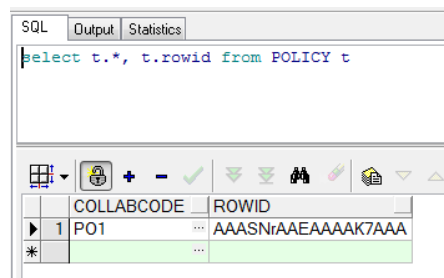
The previous procedure continues for all the tables which are in relationship.

- ii. The **Trigger Testing** includes a procedure in which we are checking the actions of updating triggers, inserting triggers and deleting triggers. In our database system we have created a trigger named Collaborator_TRG which runs automatically when an event of inserting and deleting occurs. In that case we will check only the actions of inserting trigger and deleting trigger. The Collaborator_TRG trigger has being generated for a specific table column and more especially for the column “collabcode” of the table Collaborator and the actions of inserting and deleting references the columns “collabcode” on the tables Policy, Information and Procedure.

If we insert connectortype ‘POLICY’ in the table Collaborator, then the system automatically inserts in the table Policy the value that has the collabcode in the table Collaborator. Thus if we go to the table Collaborator and insert value “PO1” for collabcode, choose the value “Policy” for connectortype, insert value “5.000” for evalue and commit the current transaction,



we will see that in the table Policy the value “PO1” of collabcode has automatically inserted.



The same test exists for tables Information and Procedures. We now go to the table Collaborator and insert the value “PR1” for collabcode, we choose the value “Procedure” for connectortype, we insert the value “10.000” for evalue. Finally we insert the value “IN1” for collabcode, choose the value “Information”

for connectortype and insert value “15.000” for evalue as shown in the following view:

The screenshot shows the SQL Developer interface with a query window displaying the following SQL statement:

```
select t.*, t.rowid from COLLABORATOR t
```

The query result is displayed in a table with the following columns: COLLABCODE, CONNECTORTYPE, EVALUE, and ROWID. The data is as follows:

	COLLABCODE	CONNECTORTYPE	EVALUE	ROWID
1	PO1	POLICY	5000.00	AAASnpAAEAAAKrAAA
2	PR1	PROCEDURE	10000.00	
3	IN1	INFORMATION	15000.00	

A confirmation dialog box is displayed over the table, asking "Commit current transaction?" with "Yes" and "No" buttons. There is also a checkbox for "Don't show this message again".

So if we go to the table Procedures, we will see that the value “PR1” of collabcode has automatically inserted:

The screenshot shows the SQL Developer interface with a query window displaying the following SQL statement:

```
select t.*, t.rowid from PROCEDURES t
```

The query result is displayed in a table with the following columns: COLLABCODE and ROWID. The data is as follows:

	COLLABCODE	ROWID
1	PR1	AAASNvAAEAAAALLAAA

Finally if we go to the table Information, we will see that the value “IN1” of collabcode has automatically inserted:

The screenshot shows the SQL Developer interface with a query window displaying the following SQL statement:

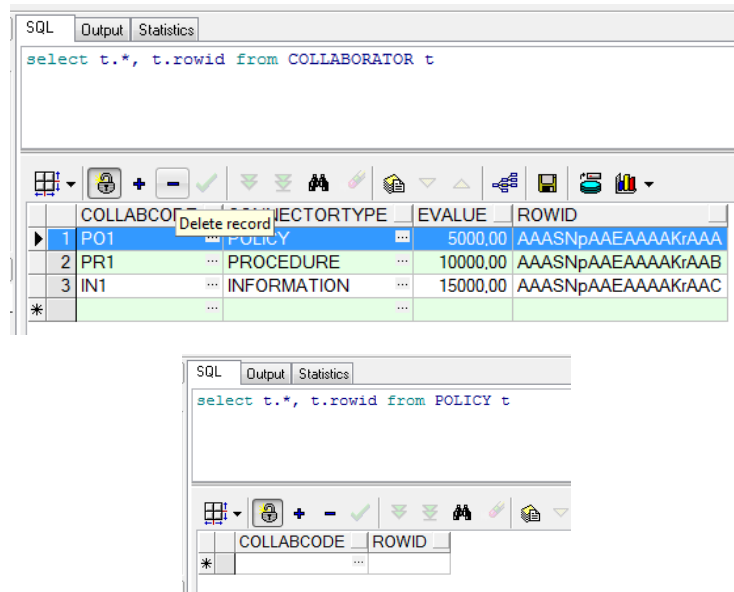
```
select t.*, t.rowid from INFORMATION t
```

The query result is displayed in a table with the following columns: COLLABCODE and ROWID. The data is as follows:

	COLLABCODE	ROWID
1	IN1	AAASNiAAEAAAALbAAA

Thus the inserting test of trigger is done.

Now let' see the case of deleting. In that case if we delete a row in the table Collaborator, then according to the connector type the system goes to a specific table and automatically deletes that collabcode value. For example if we delete the row 1 that has connectortype “Policy” then the system automatically deletes the collabcode from the table Policy, meaning the value “PO1”. Thus if commit the action of delete, then we will see that in the table Policy there is no row with the value PO1:



The same testing has been done for the tables Information and Procedure. Thus the Trigger Test is done.

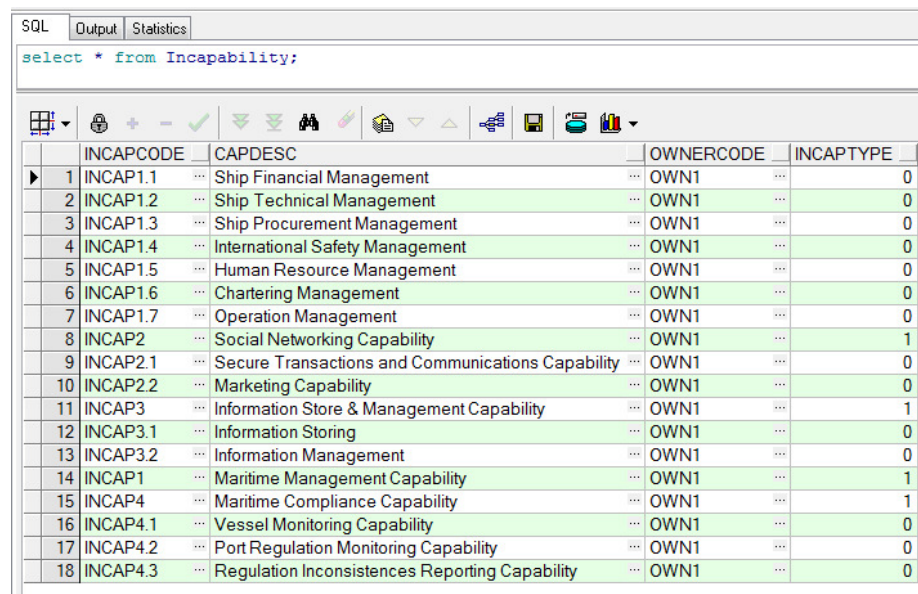
We then continue with the **Functional Testing** Method. By this method we test the functionality and the features of a back – end. In practice in this method we are doing a Database Testing in Data. Thus we will create specific functional groups and we will test it together. The Functional Groups will be:

- Functional Group 1: It will contain the tables Incapability, Excapability, Capabilityispartof and Owner.
- Functional Group 2: It will contain the tables Goal, Goalispartof and Capabilitymeetsgoal.
- Functional Group 3: It will contain the tables Contexts and Capabilityisincontext.
- Functional Group 4: It will contain the tables Output, Outputispartof and Outputisofvalue.
- Functional Group 5: It will contain the tables Collaborations, Collaborator, Procedures, Information and Policy.
- Functional Group 6: It will contain the tables Incapacity, Excapacity and Resources.
- Functional Group 7: It will contain the tables Service, Serviceispartof, Process and Tasks.
- Functional Group 8: It will contain the tables Inability, Exability and Skills.
- Functional Group 9: It will contain the tables Usescapacityforservice, Usesabilityforservice, Abilityusescapacitywithservice.

Since the Functional Testing in all the tables of database, requires a big description and analysis, in this dissertation we will present the full process of testing the Functional Group 1.

Functional Group 1: This functional group contains the tables Incapability, Excapability, Capabilityispartof and Owner. In this functional group we may answer questions referring to Ownership for Business Capability, to hierarchy of Business Capability etc. However initially we have to test its table of this functional group separately.

We begin with the table that refers to Internal Business Capability and we run a query for bringing all the kind of information that describes. We use the select statement as follows:



	INCAPCODE	CAPDESC	OWNERCODE	INCAPTYPE
1	INCAP1.1	Ship Financial Management	OWN1	0
2	INCAP1.2	Ship Technical Management	OWN1	0
3	INCAP1.3	Ship Procurement Management	OWN1	0
4	INCAP1.4	International Safety Management	OWN1	0
5	INCAP1.5	Human Resource Management	OWN1	0
6	INCAP1.6	Chartering Management	OWN1	0
7	INCAP1.7	Operation Management	OWN1	0
8	INCAP2	Social Networking Capability	OWN1	1
9	INCAP2.1	Secure Transactions and Communications Capability	OWN1	0
10	INCAP2.2	Marketing Capability	OWN1	0
11	INCAP3	Information Store & Management Capability	OWN1	1
12	INCAP3.1	Information Storing	OWN1	0
13	INCAP3.2	Information Management	OWN1	0
14	INCAP1	Maritime Management Capability	OWN1	1
15	INCAP4	Maritime Compliance Capability	OWN1	1
16	INCAP4.1	Vessel Monitoring Capability	OWN1	0
17	INCAP4.2	Port Regulation Monitoring Capability	OWN1	0
18	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	0

The results for that table are correct. Also the speed of the system for bringing this kind of query is fast (0,016 seconds), which means that runs speedily.

Now we will run a query in order to see if the system brings correct data according to some rules, concerning column names. One rule is to bring only the data for Business Capability Code and Business Capability Description for a specific Internal Business Capability for example the one that has incapcode= INCAP1. In that case we have the query:

SQL		Output	Statistics
<pre>select A.incapcode, A.Capdesc from Incapability A where incapcode= 'INCAP1'</pre>			
	INCAPCODE	CAPDESC	
1	INCAP1	Maritime Management Capability	...

The result is correct and the speed for executing the query very fast (0 seconds). However in order to insure that the system bring the correct data in that table we may ask a negative query. In the data we can see that for this kind of Business Capability there is only one owner with code OWN1. Thus we will make a query in which we will ask whether Internal Capability has an owner with ownercode “OWN2” and “OWN3” as follows:

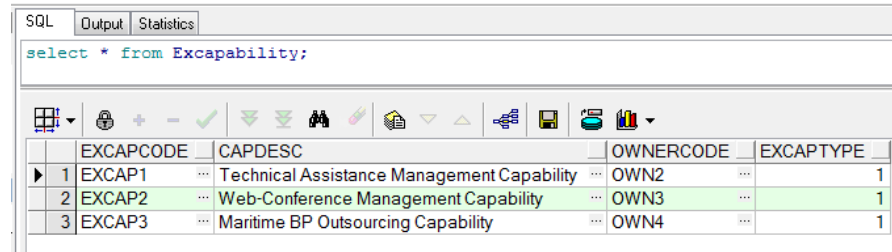
SQL		Output	Statistics
<pre>select * from Incapability where ownercode= 'OWN2' and ownercode='OWN3';</pre>			
	INCAPCODE	CAPDESC	OWNERCODE

In that case it doesn't bring any information, which is correct. Also the speed for executing this query is fast enough, since it is only 0,016 seconds. Finally we will check for duplicates in that table according to the primary key (incapcode). By that we want to ensure that there are no duplicates since the primary key is unique. Thus we will run the query:

SQL		Output	Statistics
<pre>Select A.incapcode, count (A.incapcode) from Incapability A group by A.incapcode;</pre>			
	INCAPCODE	COUNT(A.INCAPCODE)	
1	INCAP1	1	
2	INCAP1.1	1	
3	INCAP1.2	1	
4	INCAP1.3	1	
5	INCAP1.4	1	
6	INCAP1.5	1	
7	INCAP1.6	1	
8	INCAP1.7	1	
9	INCAP2	1	
10	INCAP2.1	1	
11	INCAP2.2	1	
12	INCAP3	1	
13	INCAP3.1	1	
14	INCAP3.2	1	

From the results we observe that there are no duplicates, which are correct, and the speed of executing the query very fast (0,016 seconds). Thus the check for table Incapability is done.

We now continue with the table for External Business Capability and implement the same queries as we did previous in the Internal Business Capability table. Thus we run a query for bringing all the data for that table as follows:



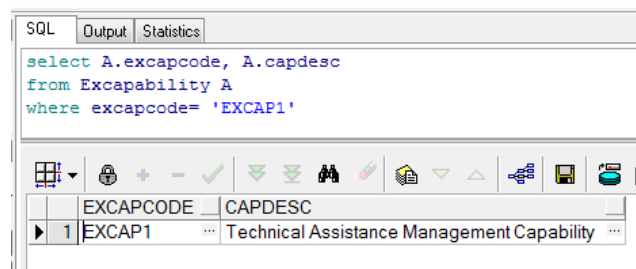
The screenshot shows a SQL query window with the query `select * from Excapability;` and its results. The results table has five columns: EXCAPCODE, CAPDESC, OWNERCODE, and EXCAPTYPE. There are three rows of data.

	EXCAPCODE	CAPDESC	OWNERCODE	EXCAPTYPE
1	EXCAP1	Technical Assistance Management Capability	OWN2	1
2	EXCAP2	Web-Conference Management Capability	OWN3	1
3	EXCAP3	Maritime BP Outsourcing Capability	OWN4	1

The results is correct and the speed of executing very fast (0,016 seconds).

Now let's implement a specific data test, for example we want to see the External Capability Description for the External Capability with code "EXCAP1".

We run a query as follows:

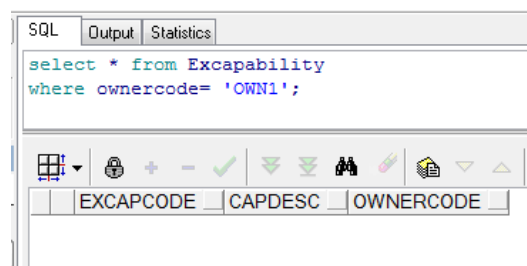


The screenshot shows a SQL query window with the query `select A.excapcode, A.capdesc from Excapability A where excapcode= 'EXCAP1'` and its results. The results table has two columns: EXCAPCODE and CAPDESC. There is one row of data.

	EXCAPCODE	CAPDESC
1	EXCAP1	Technical Assistance Management Capability

The result is correct and the speed of executing the query is very fast (0,016 seconds).

Now we may ask a negative query in that table. In the data we can see that External Business Capability owned by three kind of owners: OWN2, OWN3 and OWN4. Thus we will make a query in which we will ask whether External Capability has an owner with ownercode "OWN1" as follows:



The screenshot shows a SQL query window with the query `select * from Excapability where ownercode= 'OWN1';` and its results. The results table has three columns: EXCAPCODE, CAPDESC, and OWNERCODE. There are no rows of data.

	EXCAPCODE	CAPDESC	OWNERCODE
--	-----------	---------	-----------

In that case is doesn't bring any data, which is correct, and the speed of executing the query is very fast (0,016 seconds). Finally we check for duplicates in that table according to the primary key (incapcode). Thus we will run the query:

SQL		
<pre>select A.excapcode, count (A.excapcode) from Excapability A group by A.excapcode;</pre>		
	EXCAPCODE	COUNT(A.EXCAPCODE)
1	EXCAP1	1
2	EXCAP2	1
3	EXCAP3	1

From the results we see that there are no duplicates, which is correct, and the speed of executing the query very fast (0,016 seconds). Thus the check for table Excapability is done.

We now continue with the table Owner. First we run a query to bring all the data as follows:

SQL		
<pre>select * from Owner;</pre>		
	OWNERCODE	OWNERNAME
1	OWN1	DMC
2	OWN2	ComSys
3	OWN3	Microsoft
4	OWN4	Danaos Service/India

The result is correct and the speed of executing the query is very fast (0,031 seconds). We create a scenario by executing a query that will bring all the information for the owner with ownername= "DMC" as follows:

SQL		
<pre>select * from Owner where ownername= 'DMC';</pre>		
	OWNERCODE	OWNERNAME
1	OWN1	DMC

The result is correct and the speed of executing the query is very fast (0,015 seconds). Then we run a negative query for bringing the data that are not exist in the database. For example bring all the information for the owner with ownercode= "DMC" as follows:

SQL		
<pre>select * from Owner where ownercode= 'DMC';</pre>		
	OWNERCODE	OWNERNAME

The result is correct since there is no owner with ownercode= “DMC”, and also the speed of executing the query is very fast (0 seconds). Finally we check for duplicates according to the primary key as follows:

SQL		
Output		
Statistics		
<pre>select A.ownercode, count (A.ownercode) from Owner A group by A.ownercode;</pre>		
	OWNERCODE	COUNT(A.OWNERCODE)
1	OWN1	1
2	OWN2	1
3	OWN3	1
4	OWN4	1

The result is correct, since there are no duplicates, and also the speed of executing the query is very fast (0,031 seconds). Thus the check for table Owner is done.

We now continue with the table Capabilityispartof. For that table we first run a query for bringing all the kind of information that describes. We use the select statement as follows:

SQL		
Output		
Statistics		
<pre>select * from Capabilityispartof;</pre>		
	MCAPCODE	SUBCAPCODE
1	INCAP1	INCAP1.1
2	INCAP1	INCAP1.2
3	INCAP1	INCAP1.3
4	INCAP1	INCAP1.4
5	INCAP1	INCAP1.5
6	INCAP1	INCAP1.6
7	INCAP1	INCAP1.7
8	INCAP2	INCAP2.1
9	INCAP2	INCAP2.2
10	INCAP3	INCAP3.1
11	INCAP3	INCAP3.2
12	INCAP4	INCAP4.1
13	INCAP4	INCAP4.2
14	INCAP4	INCAP4.3

The result is correct and the speed of executing the query is very fast (0,016 seconds).

We then continue by checking the results of a query that refers to a specific column value. For example we want to see the INCAP1 Internal Business Capability into which sub-capabilities is decomposed to. Thus we run a query as follows:

SQL			Output	Statistics
<pre>select * from Capabilityispartof where mcapcode= 'INCAP1' order by subcapcode</pre>				
	MCAPCODE	SUBCAPCODE		
1	INCAP1	INCAP1.1		
2	INCAP1	INCAP1.2		
3	INCAP1	INCAP1.3		
4	INCAP1	INCAP1.4		
5	INCAP1	INCAP1.5		
6	INCAP1	INCAP1.6		
7	INCAP1	INCAP1.7		

The result is correct and the speed of executing the query is very fast (0,016 seconds).

Then we run a negative query for bringing the data that are not exist in the database. For example bring all the information for the Business Capability hierarchies with mcapcode= "GOAL1" as follows:

SQL			Output	Statistics
<pre>select * from Capabilityispartof where mcapcode= 'GOAL1' order by subcapcode</pre>				
	MCAPCODE	SUBCAPCODE		

The result is correct since there is no Business Capability hierarchies with mcapcode= "GOAL1", and also the speed of executing the query is very fast (0 seconds)

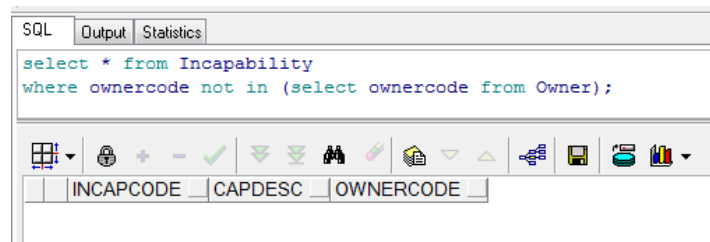
Finally we check for duplicates according the primary key. Here the primary key is a combination between two column; mcapcode and subcapcode. Thus we run a query as follows:

SQL			Output	Statistics
<pre>select mcapcode,subcapcode, count(*) from Capabilityispartof group by mcapcode,subcapcode</pre>				
	MCAPCODE	SUBCAPCODE	COUNT(*)	
1	INCAP1	INCAP1.1	1	
2	INCAP1	INCAP1.2	1	
3	INCAP1	INCAP1.3	1	
4	INCAP1	INCAP1.4	1	
5	INCAP1	INCAP1.5	1	
6	INCAP1	INCAP1.6	1	
7	INCAP1	INCAP1.7	1	
8	INCAP2	INCAP2.1	1	
9	INCAP2	INCAP2.2	1	
10	INCAP3	INCAP3.1	1	
11	INCAP3	INCAP3.2	1	
12	INCAP4	INCAP4.1	1	
13	INCAP4	INCAP4.2	1	
14	INCAP4	INCAP4.3	1	

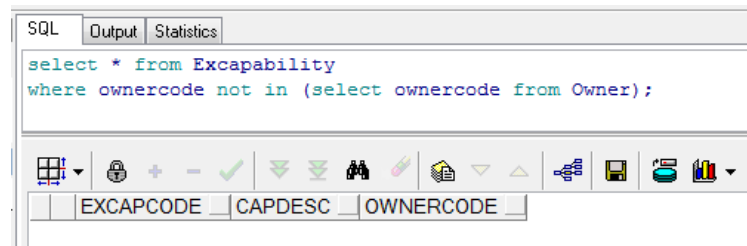
The result is correct, since there are no duplicates, and also the speed of executing the query is very fast (0,016 seconds).

Since we have already checked the data for each table separately, ***we now continue by verifying if the tables have their relationships correct, by checking if their keys matching.***

Firstly we will run a query by which *we will check if there is any value that exists in the foreign key of a table and it does not exist in the reference primary key of the other table.* For example we will check the values from the tables Incapability and Owner, according to the ownercode. The ownercode is a foreign key for the first table and a primary key for the second table. Thus we run a query as follows:



The result is correct, since every value of a primary key must match with the value of a foreign key. Also the speed of executing the query is very fast (0 seconds). We are doing the same for the tables Excapability and Owner as follows:



The result is also correct and the speed of executing the query is very fast (0 seconds).

Another test *for checking if the keys between two tables are matching is by using an example of join between those tables with the WHERE SQL condition.* For example we want all the information about Internal Capability but instead of ownercode, we want the system to bring as for each value in the instance the ownername from the Owner table. This is implemented as follows:

SQL Output Statistics			
<pre>select A.incapcode, A.capdesc, B.ownername from Incapability A, Owner B where B.ownercode= A.ownercode order by A.incapcode</pre>			
	INCAPCODE	CAPDESC	OWNERNAME
1	INCAP1	Maritime Management Capability	DMC
2	INCAP1.1	Ship Financial Management	DMC
3	INCAP1.2	Ship Technical Management	DMC
4	INCAP1.3	Ship Procurement Management	DMC
5	INCAP1.4	International Safety Management	DMC
6	INCAP1.5	Human Resource Management	DMC
7	INCAP1.6	Chartering Management	DMC
8	INCAP1.7	Operation Management	DMC
9	INCAP2	Social Networking Capability	DMC
10	INCAP2.1	Secure Transactions and Communications Capability	DMC
11	INCAP2.2	Marketing Capability	DMC
12	INCAP3	Information Store & Management Capability	DMC
13	INCAP3.1	Information Storing	DMC
14	INCAP3.2	Information Management	DMC

The result is also correct and the speed of executing the query is very fast (0,032 seconds).

We are doing the same for the tables External Capability and Owner by using a column-list joins condition as follows:

SQL Output Statistics			
<pre>select A.excapcode, A.capdesc, B.ownername from Excapability A join Owner B using (ownercode) order by A.excapcode</pre>			
	EXCAPCODE	CAPDESC	OWNERNAME
1	EXCAP1	Technical Assistance Management Capability	ComSys
2	EXCAP2	Web-Conference Management Capability	Microsoft
3	EXCAP3	Maritime BP Outsourcing Capability	Danaos Service/India

The result is also correct and the speed of executing the query is very fast (0,015 seconds).

In that case, since we have already known that there is no External Business Capability that is owned by DMC, we run a negative query as follows, and expecting that no instance will appear. Thus:

SQL Output Statistics			
<pre>select A.excapcode, A.capdesc, B.ownername from Excapability A join Owner B using (ownercode) where B.ownername= 'DMC' order by A.excapcode</pre>			
	EXCAPCODE	CAPDESC	OWNERNAME

The result is also correct and the speed of executing the query is very fast (0 seconds).

Finally another test is for checking whether in a case of union between two tables, the system brings the correct data. Thus we will run a query in order to see the totally information about Business Capability, meaning both External Business Capability and both Internal Business Capability as follows:

SQL Output Statistics			
<pre>select capcode, capdesc, ownercode from (select A.incapcode capcode, A.capdesc, A.ownercode from Incapability A union (select B.excapcode, B.capdesc, B.ownercode from Excapability B)) join Owner C using (ownercode) order by ownercode, capcode</pre>			
	CAPCODE	CAPDESC	OWNERCODE
1	INCAP1	Maritime Management Capability	OWN1
2	INCAP1.1	Ship Financial Management	OWN1
3	INCAP1.2	Ship Technical Management	OWN1
4	INCAP1.3	Ship Procurement Management	OWN1
5	INCAP1.4	International Safety Management	OWN1
6	INCAP1.5	Human Resource Management	OWN1
7	INCAP1.6	Chartering Management	OWN1
8	INCAP1.7	Operation Management	OWN1
9	INCAP2	Social Networking Capability	OWN1
10	INCAP2.1	Secure Transactions and Communications Capability	OWN1
11	INCAP2.2	Marketing Capability	OWN1
12	INCAP3	Information Store & Management Capability	OWN1
13	INCAP3.1	Information Storing	OWN1
14	INCAP3.2	Information Management	OWN1
15	INCAP4	Maritime Compliance Capability	OWN1
16	INCAP4.1	Vessel Monitoring Capability	OWN1
17	INCAP4.2	Port Regulation Monitoring Capability	OWN1
18	INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1
19	EXCAP1	Technical Assistance Management Capability	OWN2
20	EXCAP2	Web-Conference Management Capability	OWN3
21	EXCAP3	Maritime BP Outsourcing Capability	OWN4

The result is correct the speed of executing the query is very fast (0,031 seconds). Now if in the same query we want to see and the ownername instead of ownercode, we run a query as follows:

SQL Output Statistics			
<pre>select capcode, capdesc, ownername from (select A.incapcode capcode, A.capdesc, A.ownercode from Incapability A union (select B.excapcode, B.capdesc, B.ownercode from Excapability B)) join Owner C using (ownercode) order by ownercode, capcode</pre>			
	CAPCODE	CAPDESC	OWNERNAME
1	INCAP1	Maritime Management Capability	DMC
2	INCAP1.1	Ship Financial Management	DMC
3	INCAP1.2	Ship Technical Management	DMC
4	INCAP1.3	Ship Procurement Management	DMC
5	INCAP1.4	International Safety Management	DMC
6	INCAP1.5	Human Resource Management	DMC
7	INCAP1.6	Chartering Management	DMC
8	INCAP1.7	Operation Management	DMC
9	INCAP2	Social Networking Capability	DMC
10	INCAP2.1	Secure Transactions and Communications Capability	DMC
11	INCAP2.2	Marketing Capability	DMC
12	INCAP3	Information Store & Management Capability	DMC
13	INCAP3.1	Information Storing	DMC
14	INCAP3.2	Information Management	DMC
15	INCAP4	Maritime Compliance Capability	DMC
16	INCAP4.1	Vessel Monitoring Capability	DMC
17	INCAP4.2	Port Regulation Monitoring Capability	DMC
18	INCAP4.3	Regulation Inconsistences Reporting Capability	DMC
19	EXCAP1	Technical Assistance Management Capability	ComSys
20	EXCAP2	Web-Conference Management Capability	Microsoft
21	EXCAP3	Maritime BP Outsourcing Capability	Danaos Service/India

The result is correct the speed of executing the query is very fast (0,031 seconds).

We also run a query in order to see the composition of Business Capability into Sub-Capabilities and their description. Thus we run a query as follows:

SQL	Output	Statistics
<pre>select incapcode capcode, capdesc, decode(incaptype,0,'Sub','Main') captype from incapability union select excapcode capcode, capdesc, decode(excaptype,0,'Sub','Main') captype from excapability order by 1, 3</pre>		
CAPCODE	CAPDESC	CAPTYPE
1	EXCAP1 ... Technical Assistance Management Capability	Main
2	EXCAP2 ... Web-Conference Management Capability	Main
3	EXCAP3 ... Maritime BP Outsourcing Capability	Main
4	INCAP1 ... Maritime Management Capability	Main
5	INCAP1.1 ... Ship Financial Management	Sub
6	INCAP1.2 ... Ship Technical Management	Sub
7	INCAP1.3 ... Ship Procurement Management	Sub
8	INCAP1.4 ... International Safety Management	Sub
9	INCAP1.5 ... Human Resource Management	Sub
10	INCAP1.6 ... Chartering Management	Sub
11	INCAP1.7 ... Operation Management	Sub
12	INCAP2 ... Social Networking Capability	Main
13	INCAP2.1 ... Secure Transactions and Communications Capability	Sub
14	INCAP2.2 ... Marketing Capability	Sub
15	INCAP3 ... Information Store & Management Capability	Main
16	INCAP3.1 ... Information Storing	Sub
17	INCAP3.2 ... Information Management	Sub
18	INCAP4 ... Maritime Compliance Capability	Main
19	INCAP4.1 ... Vessel Monitoring Capability	Sub
20	INCAP4.2 ... Port Regulation Monitoring Capability	Sub
21	INCAP4.3 ... Regulation Inconsistencies Reporting Capability	Sub

The result is correct the speed of executing the query is very fast (0,031 seconds).















Finally we will run a query to see the total information for all the tables of that functional group, meaning to see what Business Capability the organizations has, who is the owner of that Business Capability and which of them are Main or Sub Business Capabilities. Thus we run a query as follows:

SQL

Output

Statistics

```
select capcode, capdesc, ownername, captype from
(select A.incapcode capcode, A.capdesc, A.ownercode, decode(incaptype,0,'Sub','Main') cap
from Incapability A
union
(select B.excapcode, B.capdesc, B.ownercode, decode(excaptype,0,'Sub','Main') captype
from Excapability B))
join Owner C
using (ownercode)
order by ownercode, capcode
```

	CAPCODE		CAPDESC		OWNERNAME		CAPTYPE	
▶	1	INCAP1	...	Maritime Management Capability	...	DMC	...	Main
	2	INCAP1.1	...	Ship Financial Management	...	DMC	...	Sub
	3	INCAP1.2	...	Ship Technical Management	...	DMC	...	Sub
	4	INCAP1.3	...	Ship Procurement Management	...	DMC	...	Sub
	5	INCAP1.4	...	International Safety Management	...	DMC	...	Sub
	6	INCAP1.5	...	Human Resource Management	...	DMC	...	Sub
	7	INCAP1.6	...	Chartering Management	...	DMC	...	Sub
	8	INCAP1.7	...	Operation Management	...	DMC	...	Sub
	9	INCAP2	...	Social Networking Capability	...	DMC	...	Main
	10	INCAP2.1	...	Secure Transactions and Communications Capability	...	DMC	...	Sub
	11	INCAP2.2	...	Marketing Capability	...	DMC	...	Sub
	12	INCAP3	...	Information Store & Management Capability	...	DMC	...	Main
	13	INCAP3.1	...	Information Storing	...	DMC	...	Sub
	14	INCAP3.2	...	Information Management	...	DMC	...	Sub
	15	INCAP4	...	Maritime Compliance Capability	...	DMC	...	Main
	16	INCAP4.1	...	Vessel Monitoring Capability	...	DMC	...	Sub
	17	INCAP4.2	...	Port Regulation Monitoring Capability	...	DMC	...	Sub
	18	INCAP4.3	...	Regulation Inconsistencies Reporting Capability	...	DMC	...	Sub
	19	EXCAP1	...	Technical Assistance Management Capability	...	ComSys	...	Main
	20	EXCAP2	...	Web-Conference Management Capability	...	Microsoft	...	Main
	21	EXCAP3	...	Maritime BP Outsourcing Capability	...	Danaos Service/India	...	Main

The result is also correct the speed of executing the query is very fast (0,031 seconds).

6.4 Chapter Summary

The concern of this Chapter was the physical database of under development maritime application. More specific in this chapter we have worked in a third level of analysis according to ORM, with the main objective to create a physical database for our application. As we have already discussed in this Chapter this database has being guided by the Relational Schema that has being created in the previous chapter.

Before we provide a presentation of the physical tables of this database, we thought that it was important to give a brief description of the reasons why we have chosen a specific software DBMS for implemented the database for our Application. Thus we have chosen the Oracle commercial DBMS for this purpose, since it is the most popular in that period (www.db-engines.com).

Then we have presented the basic Architecture for our application. More specific we have use a three-tier architecture ([Tickoo & Raina, 2010](#)) of client-server (client, application and database server), in which the first level refers to a specific database server (the Oracle Database 11g), the second level to a specific Integrated Development Environment – IDE (the Oracle PL/SQL 6i) and the third to a specific application server (the Oracle Forms & Reports 6i).

Thereafter the physical tables of this database were presented, with a brief description whenever required, of the implemented constraints or triggers in the SQL language. Except from the physical tables we have also presented some other objects of this database, meaning the created sequences and views.

Finally in order to insure data validation, data integrity, performance checks to database, testing of the triggers and that the database transactions are processed concurrently, meaning that they satisfy the ACID properties (Atomicity, Consistence, Isolation and Durability), we have followed a specific process of Database Testing. The steps of this process was first the preparation of the environment, second the procedure of running the test, third the checking of the test results, fourth the validation of this process and fifth the reporting of the findings. Thus firstly we have taken real examples of data from the use case of the Danaos Management Consultant and we use it for preparing the environment. Then we followed specific procedures of Database Testing and more specific Structural Database Testing and Functional Database Testing as described by (www.tutorialspoint.com). Worth mentioning that during these procedures the other four steps of the Process of Testing the Database are overlapped, meaning that some of these steps are implemented at the same time.

For implementing the Structural Database Testing firstly we are executing a Database Schema Testing and then a Trigger testing. For the first of this option, in the first step we are executing test in Database name and we are looking if there is enough existence of space in the database. Then in the second step we continue with tests in tables, columns, columns types, default values and rules. Thus we are checking at least once for the entire tables the Table Names and the Columns Names in relation with the Relational Schema. Also we are testing if the values types are the appropriate to describe this kind of data and if the length of those values is the efficient enough. Thereafter we are testing whether a rule is bound to correct table columns, meaning we are testing if check, value and ring constraints are implemented correctly. In the third step we are executing test in keys and indexes, meaning we are checking if for the implemented primary keys when inserted a data in the database is violated the rule that this key must be unique and also if the foreign keys values are automatically appeared from the primary keys values. Finally if for every candidate key it has being created by the system a specific index. Finally we are executing a Trigger Testing and more detail we check if all the actions that have being defined by the trigger work properly.

Then we are implementing a Functional Testing, in which we have divided all the database tables in nine functional groups. For each group we are testing with the help of the SQL language if the system brings the correct results in a query according to specific rules and the speed of executing the queries. Also we execute negative queries like checking for duplicates in primary keys. Finally we execute queries in order to verifying if their tables have their relationship correct, by checking if their keys matching.

From the above that described in this Chapter we can say that it is important how we have designed the Relational Schema in order to create the physical database of our system. That's because everything is depicted in this schema, must then take the form of an object in the database and sometimes this is not feasible at once. This means we have to implement some extra procedural code like triggers, sequences etc, in order the information to be maintained with the right way. The last one presupposes a good knowledge of SQL language by the developer of this system. However in this stage as we observe some of the textual constraints that refers to unions of specific object (e.g. Business Capability (capcode) = InternalCapability (incapcode) union ExternalCapability (excapcode)), has not being implemented yet. Although in this stage it is possible to miss some of the required implemented constraints, however as we will see we can implemented them in the application level.

CHAPTER 7: User Interface Design & Implementation

Structure of this Chapter

- 7.1 User Interface Design Process & Quality Characteristics
- 7.2 Use Case Diagram as a Description of the Main Windows
- 7.3 Hierarchy of Forms
- 7.4 Basic Flow Chart for Data Entry
- 7.5 Application Screens
- 7.6 Chapter Summary

This chapter deals with working in a fourth level of our Approach according to ORM, meaning the designing and implementation of a User Interface that will be used for the maritime application of Danaos Management Consultant Company. Thus in Section 7.1 a User Interface designing process is presented and the quality characteristics that a User Interface must have. Then in Section 7.2 we present a Use Case Diagram for describing the Main Windows of this application. Also in Section 7.3 a diagram of the hierarchies of forms is given in order to help the developer creating the main menu for navigating between the forms. Then in Section 7.4 a basic flow chart for data entry in a case of a new Business Capability is presented, in order to facilitate a user in how he works with the application and also the developer in the creation of this User Interface. Finally in Section 7.5 the application screens are presented and in 7.6 a brief summary of this chapter.

7.1 User Interface Design Process & Quality Characteristics

Since we have created the physical database schema, in this chapter we will create an external schema, which involves the designing of an appropriate interface for users.

As referred by (Mandel, 2002) an Interface is *“the presentation, communication and interaction between the user and the system”*. However a user interface design is *“the process of designing the way in which systems user can access system functionality, and the way that information produced by the system is displayed”* (Sommerville, 2007).

The process of developing a User’s Interface has four major phases (Mandel, 2002) as shown in Figure 44. The first phase deals with gathering/ analyzing user’s information, the second deals with the designing of user’s interface, the third with constructing the user’s interface and finally the fifth with validating the user’s interface.

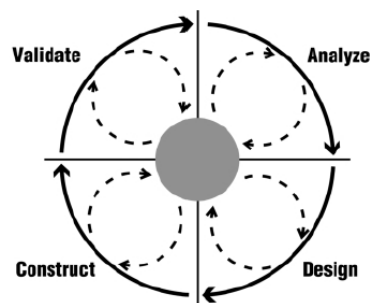


Figure 45: The elements of User Interface Design (Mandel, 2002)

Sommerville (2007) refers that the process of designing an effective interface is crucial for the application development since users often judge a system by its interface rather than its functionality. Also he mentions that a poorly designed interface can cause a user to make catastrophic errors and is the main reason why many software systems are never used. Thus Sommerville (2007) taking into account the previous mentioned that it is important to take into account specific principles when we are making user interface design decisions. Those are:

Principle	Description
User familiarity	The interface should use terms and concepts drawn from the experience of the people who will make most use of the system.
Consistency	The interface should be consistent in that, wherever possible, comparable operations should be activated in the same way.
Minimal surprise	Users should never be surprised by the behaviour of a system.
Recoverability	The interface should include mechanisms to allow users to recover from errors.
User guidance	The interface should provide meaningful feedback when errors occur and provide context-sensitive user help facilities.
User diversity	The interface should provide appropriate interaction facilities for different types of system users.

Figure 46: User Interface Design Principles (Sommerville, 2007)

In general in order to create a quality software system from user perspective according ISO/IEC 9126, this system must have the characteristics that shown in Figure 46.

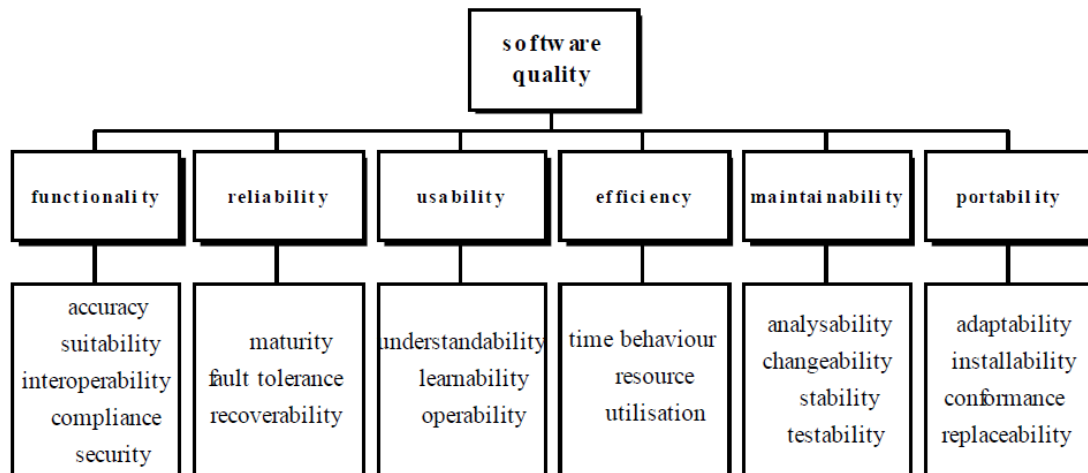


Figure 47: Software Quality Characteristics (Bevan, 1999)

However a user interface is an essential component of any software system, thus while design it is necessary to taking into account some specific quality criteria. According to (Oren & Çetin, 1999) twenty seven quality criteria are identified for user/system interfaces which are grouped in four areas namely, convenience (or usability), communicativeness, reliability and evolvability. Those criteria are shown in the following graph:

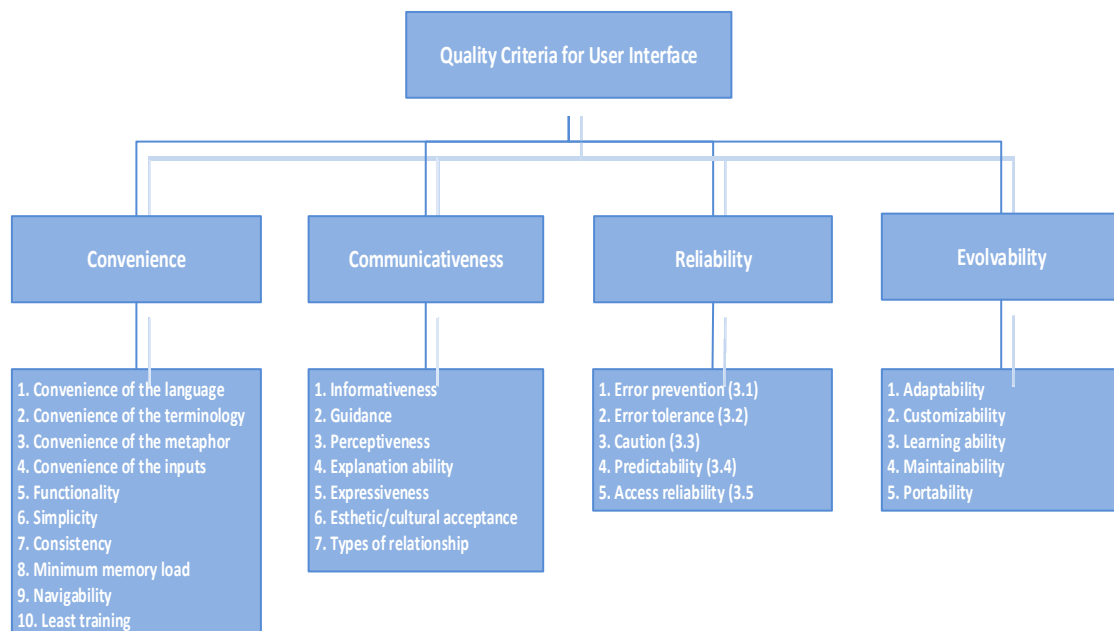


Figure 48: Quality Criteria for User/System Interface (Oren & Çetin, 1999)

The rest of this chapter deals with giving a Use Case Diagram as a description of the Main Windows of our Application, also in giving a hierarchy of Forms, and by giving a basic flow chart which describes the steps that a user follows in a case of a data entry. Finally it presents the Final Screens of our Application, detailed by the basic characteristics of their fields, in order a user to have the knowledge about the fields that fill in.

7.2 Use Case Diagram as a Description of the Main Windows

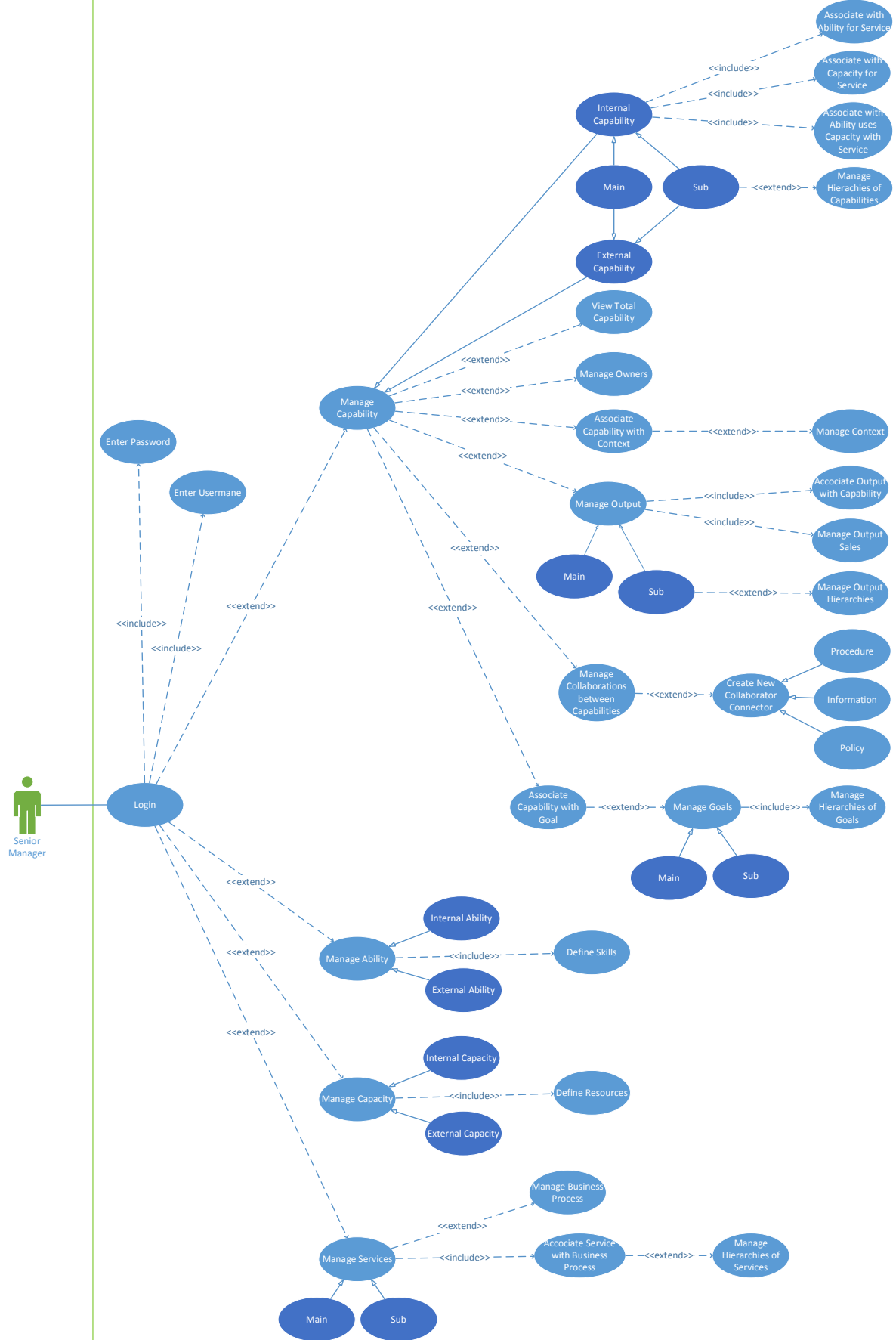
In the previous section we discuss about the interface development process and the quality criteria during this process. In this section we will describe how a user interacts with the Application in order to achieve the goals of this application, by using a Use Case Diagram.

This Use Case Diagram will be a mean for designing the User's Interface of our Application and not a mean for describing the functional or non-functional requirement of the Application. This means that this Use Case Diagram can be seen as a description of the main windows of the Application.

Users that may interact with the Application are represented by "actors". In our system different actors may interact with the system, for example an Owner User may have the ability to see only the information about the Total Business Capability or a Senior Manager User may have the ability to manage all the information about Business Capability. However in this Use Case Diagram we will only represent only one actor, the Senior Manager. Also the "use cases" will represent the set of tasks that the actor carry out, have "include" and "extension" relationships, and can also be related with "generalization" relationships that compare more particular tasks.

Thus the Use Case Diagram is following thereafter:

System Boundary



7.3 Hierarchy of Forms

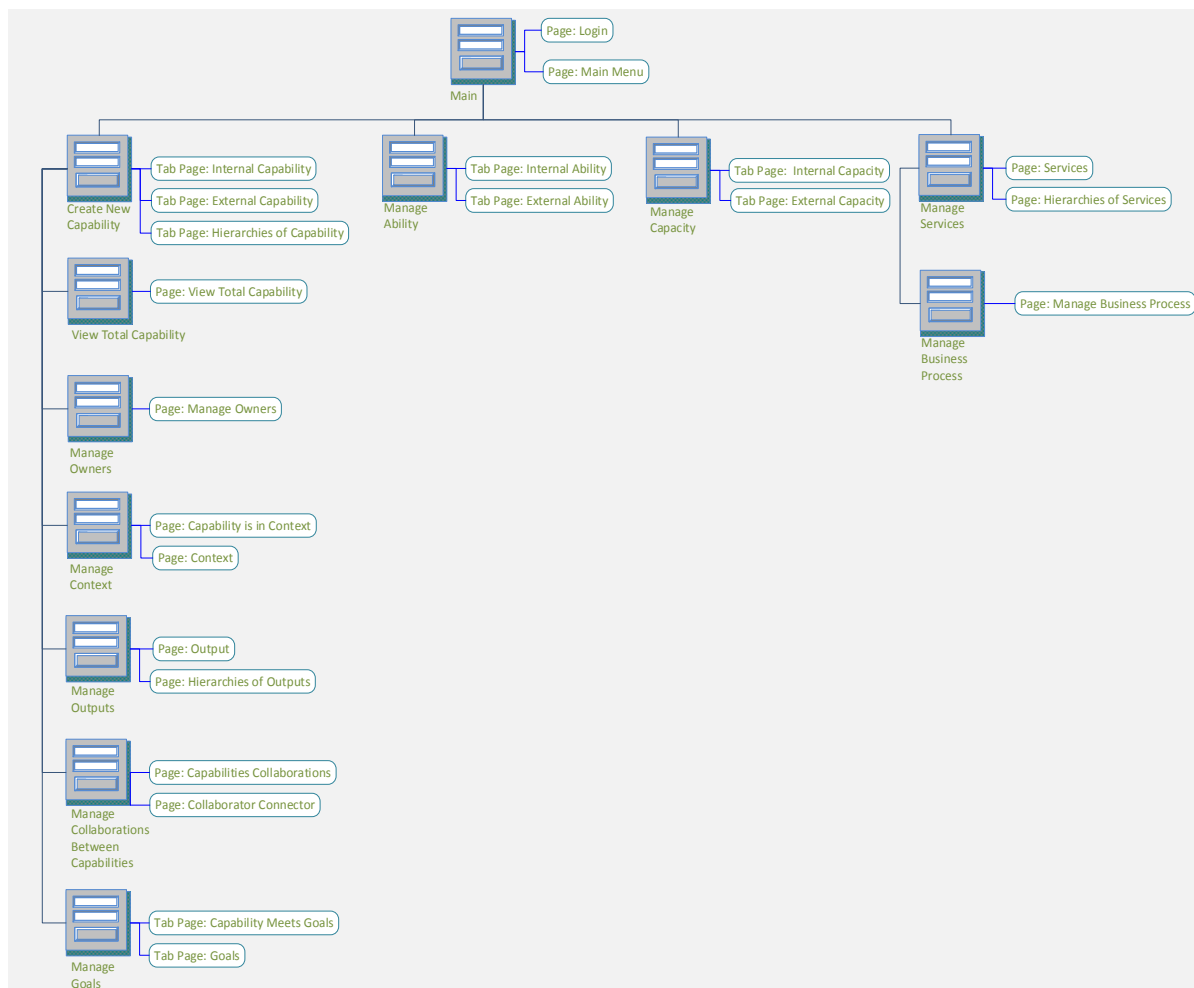
In the previous section we discuss some a Use Case Diagram as a description of the main windows of our Application. However before we give the final Pages (Screens) for the totality of Forms of our Application, it is important to discuss the hierarchy of them.

Our Application contains twelve Forms, which in some cases includes more than Tab Pages. Those Forms are the following:

- **Main:** This Form concerns the action of login the system and then navigating between the different Pages of our Application by a main menu. Thus it contains two Sub Pages:
 - a) The Sub Page “Login”
 - b) The Sub Page “Main Menu”.
- **Create New Capability:** This Form concerns the action of Inserting a New Business Capability. However except of the inserting the system allows the actions of deleting, updating, querying and printing. It contains three Tab Pages:
 - a) The Tab Page “Internal Capability”
 - b) The Tab Page “External Capability”
 - c) The Tab Page “Hierarchies of Capabilities”.
- **View Total Capability:** This Form concerns the action of Viewing Total Business Capability. Thus in this Form we can view for each Business Capability, his Type (meaning External or Internal), his hierarchies Type (meaning Main or Sub), his owner, its Sub Capabilities, the outputs that delivers, the associated goals, the context in which it exists, the collaborations, what ability uses for delivering a specific service, what capacity uses for delivering a specific Service and finally what Ability and Capacity uses in association with Service.
- **Manage Owners:** This Form concerns the actions for management the information about Owners (inserting, deleting, updating, querying and printing).
- **Manage Context:** This Page concerns the actions for management the information about Context (inserting, deleting, updating, querying and printing) and contains two Tab Pages:
 - a) The Tab Page “Capability is in Context”
 - b) The Tab Page “Context”.
- **Manage Goals:** This Form concerns the actions for management the information about Goals and the hierarchies of them (inserting, deleting, updating, querying and printing) contains two Tab Pages:

- a) The Tab Page “Capability Meets Goal”
 - b) The Tab Page “Goals”.
- **Manage Outputs:** This Form concerns the actions for management the information about Output and the hierarchies of them (inserting, deleting, updating, querying and printing) contains two Tab Pages:
 - a) The Tab Page “Outputs”
 - b) The Tab Page “Hierarchies of Outputs”.
- **Manage Collaborations Between Capabilities:** This Form concerns the actions for management the information about Collaborations between Business Capabilities through a Collaborator Connector (inserting, deleting, updating, querying and printing) and contains two Tab Pages:
 - a) The Tab Page “Capabilities Collaborations”
 - b) The Tab Page “Collaborator Connector”.
- **Manage Ability:** This Form concerns the actions for management the information about Ability in relation with Skills (inserting, deleting, updating, querying and printing) and contains two Tab Pages:
 - a) The Tab Page “Internal Ability”
 - b) The Tab Page “External Ability”.
- **Manage Capacity:** This Form concerns the actions for management the information about Capacity in relation with Resources (inserting, deleting, updating, querying and printing) and contains two Tab Pages:
 - a) The Tab Page “Internal Capacity”
 - b) The Tab Page “External Capacity”
- **Manage Services:** This Form concerns the actions for management the information about Services and the hierarchies of them (inserting, deleting, updating, querying and printing), and contains two Tab Pages:
 - a) The Tab Page “Services”
 - b) The Tab Page “Hierarchies of Services”
- **Manage Business Process:** This Form concerns the actions for management the information about Process in relation with Tasks (inserting, deleting, updating, querying and printing).

Thus according to the previous we now specify a hierarchy of the Forms, which will also be used for the development of Main Menu, that will help us in navigation between the Pages in our application. Thus this hierarchy is given in the following diagram:



7.4 Basic Flow Chart for Data Entry

We now continue *by describing the main flows in a case of data entry a New Business Capability*. When a new Business Capability exists, then the user of the application deals with specific questions, who results to specific processes or sub-processes. In this procedure there are specific steps. Those steps are:

- Step 1:** Go to the Page “Login” and fill in User Name and Password. If the User Name and Password is Correct, then go to the Page “Main Menu” and continue with the next Steps. Otherwise re fill in the same fields.
- Step 2:** The user questions whether this new Business Capability has a new owner. If a new owner exists then the user goes to the Page “Manage Owners” and

inserts the new owner. Then he continues with the Step 3. Otherwise if there is no New Owner he continues with the Step 3.

Step 3: In this Step the user questions whether this New Business Capability is Internal. If not (which means that the Business Capability is External) then the user questions whether this Capability is a Sub Capability. If the External Business Capability is a Sub Capability then the user goes to the Tab Page “Hierarchies of Capabilities” in Form “Create New Capability” and inserts the Information about the hierarchy of Capabilities. Thereafter he goes to the Tab Page “External Capability” in the same Form and inserts the information concerning External Capability. Then he continues with the STEPS 10, 11, 12, 13 & 14. Finally If the New Business Capability is Internal then the user continues with Step 4.

Step 4: In this Step the user questions whether this New Business Capability has a New Ability. If it hasn’t then he proceeds to Step 5. Otherwise question whether the New Ability is Internal. If this exist then goes to the Tab Page “Internal Ability” in the Form “Manage Ability” and inserts the associated information. At the same Tab Page the user also inserts the information about the Skills that defines Internal Ability. Finally he continues with the next Step.

If the New Ability is not Internal (which means that is External) then the user goes to the Tab Page “External Ability” in the same Form and insert the associated information. In this Layout the user also inserts the information about the Skills that defines External Ability. Finally he continues with the next Step.

Step 5: In this Step the user questions whether this New Business Capability has a New Capacity. If it hasn’t then he proceeds to Step 6. Otherwise question whether the New Capacity is Internal. If this exist then goes to the Tab Page “Internal Capacity” in the Form “Manage Capacity” and inserts the associated information. At the same Tab Page the user also inserts the information about the Resources that defines Internal Capacity. Finally he continues with the next Step.

If the New Capacity is not Internal (which means that is External) then the user goes to the Tab Page “External Capacity” in the same Form and insert the associated information. In this Tab Page the user also inserts the

information about the Resources that defines External Capacity. Finally he continues with the next Step.

Step 6: In this Step the user questions whether this New Business Capability has a New Service. If this not exists then continues with the Step 9. Otherwise he questions whether this Service has a specific Business Process. If it has then the user goes to the Page “Manage Business Process” and inserts the information about it. Also in the same Page he associates Business Process with specific Tasks. Finally he continues with the next step.

If the Service is not delivered by a Specific Business Process, then the user continues with the next Step.

Step 7: In this Step the user questions whether this New Service is a Sub Service. If the Service is a Sub Service, then the user goes to the Page “Hierarchies of Services” in the Form “Manage Services” and inserts the Information about the hierarchy of Services. Thereafter he goes in the same Form to the Page “Services” and inserts the information about it.

If the Service is not a Sub Service, then the user goes to the Page “Services”, inserts the information about it and continues with the next Step.

Step 8: In this Step the user questions whether the New Internal Capability is a Sub Capability. If the Internal Capability is a Sub Capability, then the user goes to the Tab Page “Hierarchies of Capabilities” in the Form “Create New Capability” and inserts the Information about the hierarchy of Capabilities. Thereafter he goes in the same Form to the Tab Page “Internal Capability” and inserts the information about it.

If the Internal Capability is not a Sub Capability, then the user goes in the same Form to the Tab Page “Internal Capability”, inserts the information about it and continues with the next Step.

NOTE: The next steps concern either Internal Business Capability of External Business Capability.

Step 9: In this Step the user questions whether this New Business Capability is in a New Context. If this not exists then the user goes to the Page “Capability is in Context”, in the Form “Manage Context” and associate the Business Capability with a specific context. Finally he continues with the next Step.

If the Business Capability is in a New Context, then the user goes in the same Form to the Page “Context” and inserts the information about the New

Context. Then goes to the Page “Capability is in Context”, associate the Business Capability with a specific context and continues with the next Step.

Step 10: In this Step the user questions whether the Output is delivered by Business Capability is a Sub Output. If the Output is a Sub Output, then the user goes in the Form “Manage Outputs” to the Page “Hierarchies of Outputs” and inserts the Information about the hierarchy of Outputs. Thereafter he goes to the in the same Form to the Page “Output” and inserts the information about it.

If the Output is not a Sub Output, then the user goes to the Page “Output”, inserts the information about it and continues with the next Step.

Step 11: In this Step the user questions whether this New Business Capability collaborates with some other Business Capability through a Collaborator Connector. If this not exists then the user goes to the Next Step. Otherwise he questions if in this type of Collaboration, there is a New Collaborator Connector. If this exists then the user goes to the Form “Manage Collaborations between Capabilities” to the Page “Collaborator Connector” and inserts the information about the Collaborator Connector. Then the user goes in the same Form to the Page “Capabilities Collaborations”, and correlates the Business Capabilities with the New Collaborator Connector.

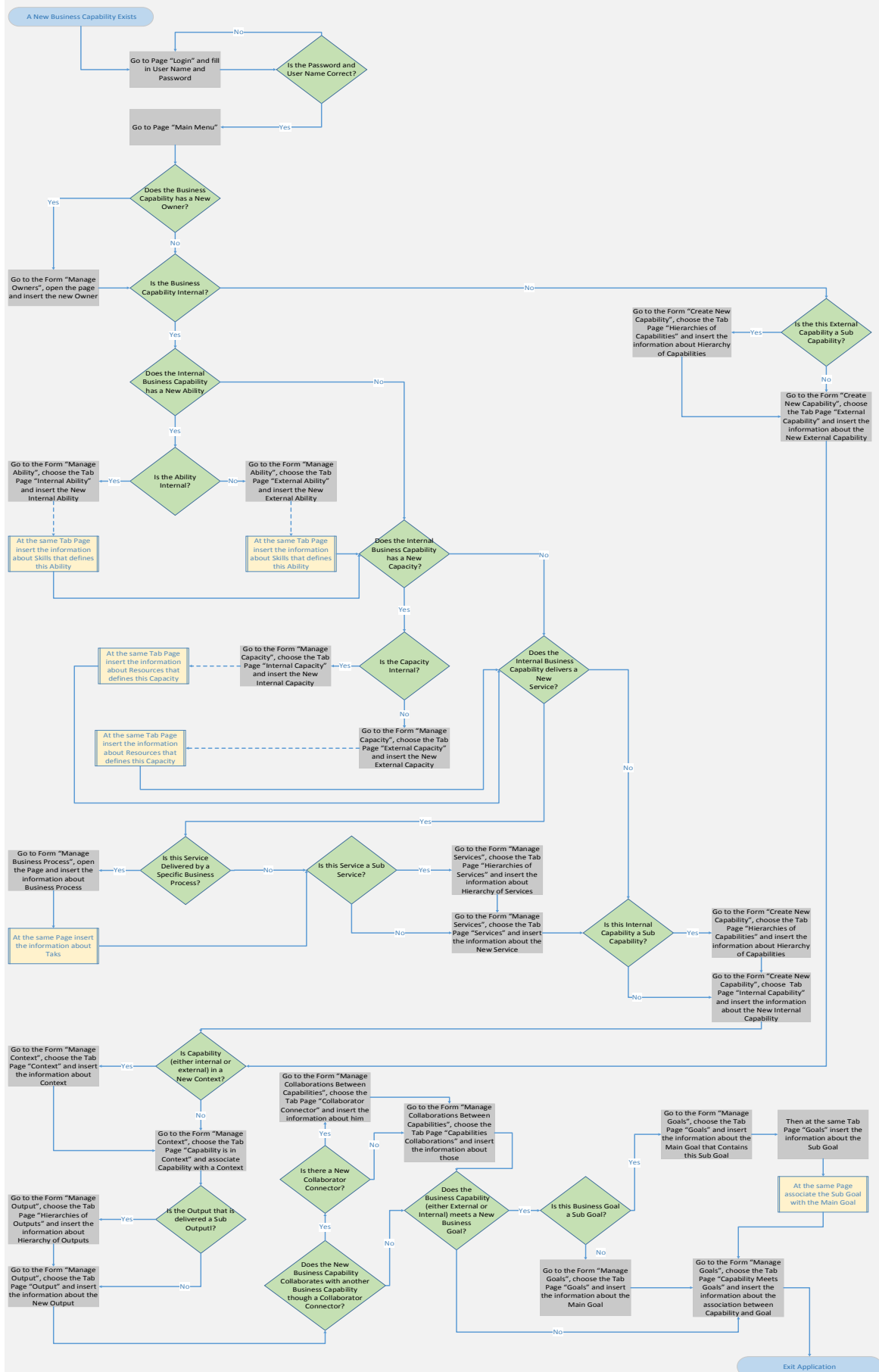
If there is not a New Collaborator Connector, then the user goes to the Page “Capabilities Collaborations”, correlates the Business Capabilities with the New Collaborator Connector and continues with the next Step.

Step 12: In this Step the user questions whether this New Business Capability meet a New Business Goal. If this not exists then the user goes in the Form “Manage Goals” to the Tab Page “Capability Meets Goal” and correlate the Business Capability with a pre-existing Goal. Otherwise he questions whether the new Goal is a Sub Goal. If this goal is a Sub Goal, then the user goes in the same Form to the Tab Page “Goals” first inserts the information about the main goal and then inserts the sub goal and the hierarchy of him. Otherwise in the same Tab Page he inserts only the information about the main goal and the goes to the Tab Page “Capability Meets Goal”, to correlate the Business Capability with the new Goal.

Step 13: In this Step the user exits the Application

The previous steps are shown in the following Flow Chart:

Basic Flow Chart of Navigation Between Pages of Application in a Case of A New Business Capability Existence.



7.5 Application Screens

In this section we present the final Screens for our Application. Also in the bottom of every Screen we provide a brief description of the fields they contain, in order the user to be familiar with the way they fill in it.

Thus the Main Screens of our Application follows thereafter:

1. Main:

This Screen contains the Sub Pages:

Login:

The main characteristics of the fields for “Login” Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
User Name	Yes	Character (10)	Upper	No	By the User
Password	Yes	Character (10)	Upper – Concealed Data	No	By the User

Main Menu:

2. Create New Capability:

This Screen contains the Tab Pages:

Internal Capability

The main characteristics of the fields for “Internal Capability” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Internal Capability Code	Yes	Character (50)	Upper	No	By the User
Internal Capability Description	Yes	Character (250)	Mixed	No	By the User
Internal Capability Owner	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Owner Code and Owner Description Values.	By the User
Internal Capability Type	Yes	Number(1) List Item	–	Main (=1) or Sub (=0)	By the User
Capacity Description	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Capability Description Values.	By the User
Service Name	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Service Name Values.	By the User
Ability Description	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Ability Description Values.	By the User

External Capability

Oracle Forms Runtime - [CAPABILITY.WJ]
 Action Edit Query Block Record Field Window Help
 CREATE NEW CAPABILITY
 Internal Capability External Capability Hierarchies of Capabilities
 External Capability Code
 External Capability Description
 External Capability Owner
 External Capability Type Hierarchies of Capabilities

The main characteristics of the fields for “Internal Capability” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
External Capability Code	Yes	Character (50)	Upper	No	By the User
External Capability Description	Yes	Character (250)	Mixed	No	By the User
External Capability Owner	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Owner Code and Owner Description Values.	By the User
External Capability Type	Yes	Number(1) List Item	–	Main (=1) or Sub (=0)	By the User

Hierarchies of Capabilities

Oracle Forms Runtime - [CAPABILITY.WJ]
 Action Edit Query Block Record Field Window Help
 CREATE NEW CAPABILITY
 Internal Capability External Capability Hierarchies of Capabilities
 Main Capability Sub Capability
 Back to Internal Capability Back to External Capability

The main characteristics of the fields for “Internal Capability” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Main Capability	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Main Capability Code and Main Capability Description Values.	By the User
Sub Capability	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Sub Capability Code and Sub Capability Description Values.	By the User

3. View Total Capability:

This Page contains all the information about Business Capability. A user may execute a query by inserting the “Capability Code” field or the “Capability Description field” and then the system brings all the other fields that associated with the inserted value.

The main characteristics of the top fields for “View Total Capability” Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Capability Code	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Capability Code and Capability Description Values.	By the User
Capability Description	Yes	Character (250)	–	Returns Capability Description Values.	Automatic by the System

Capability Type	Yes	Character (8)	–	External or Internal	Automatic by the System
Owner Name	Yes		–	–	Automatic by the System

The main characteristic of the fields that included in each Tab Page are given thereafter.

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																				
<div> <div>Sub Capability Code</div> <div>Sub Capability Description</div> <table border="1"> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> </table> </div>																											

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Sub Capability Code	Yes	Character (50)	–	–	Automatic by the System
Sub Capability Description	Yes	Character (250)	–	–	Automatic by the System

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																														
<div> <div>Output Code</div> <div>Output Name</div> <div>Output Type</div> <table border="1"> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> </table> </div>																																					

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Output Code	Yes	Character (50)	–	–	Automatic by the System
Output Name	Yes	Character (250)	–	–	Automatic by the System
Output Type	Yes	Number(1)	–	Main (=1) or Sub (=2)	Automatic by the System

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																														
<table border="1"> <thead> <tr> <th>Goal Code</th> <th>Goal Name</th> <th>Goal Type</th> </tr> </thead> <tbody> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td></tr> </tbody> </table>								Goal Code	Goal Name	Goal Type																											
Goal Code	Goal Name	Goal Type																																			

Characteristics of fields

FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Goal Code	Yes	Character (50)	–	–	Automatic by the System
Goal Name	Yes	Character (250)	–	–	Automatic by the System
Goal Type	Yes	Number(1)	–	Main (=1) or Sub (=2)	Automatic by the System

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																				
<table border="1"> <thead> <tr> <th>Context Code</th> <th>Context Description</th> </tr> </thead> <tbody> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> <tr><td></td><td></td></tr> </tbody> </table>								Context Code	Context Description																		
Context Code	Context Description																										

Characteristics of fields

FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Context Code	Yes	Character (50)	–	–	Automatic by the System
Context Description	Yes	Character (150)	–	–	Automatic by the System

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																																								
<table border="1"> <thead> <tr> <th>Capability Code</th> <th>Capability Description</th> <th>Collaborator Code</th> <th>Collaborator Type</th> </tr> </thead> <tbody> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> </tbody> </table>								Capability Code	Capability Description	Collaborator Code	Collaborator Type																																				
Capability Code	Capability Description	Collaborator Code	Collaborator Type																																												

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Capability Code	Yes	Character (50)	–	–	Automatic by the System
Capability Description	Yes	Character (250)	–	–	Automatic by the System
Collaborator Code	Yes	Character (50)	–	–	Automatic by the System
Collaborator Type	Yes	Character (20)	–	Policy or Information or Procedure	Automatic by the System

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																																												
<table border="1"> <thead> <tr> <th>Ability Code</th> <th>Ability Description</th> <th>Service Code</th> <th>Service Name</th> </tr> </thead> <tbody> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> </tbody> </table>								Ability Code	Ability Description	Service Code	Service Name																																								
Ability Code	Ability Description	Service Code	Service Name																																																

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Ability Code	Yes	Character (50)	–	–	Automatic by the System
Ability Description	Yes	Character (250)	–	–	Automatic by the System
Service Code	Yes	Character (50)	–	–	Automatic by the System
Service Name	Yes	Character (250)	–	–	Automatic by the System

Sub Capabilities	Outputs	Goals	Context	Collaborates	Uses Ability for Service	Uses Capacity for Service	Uses Ability - Capacity with Service																																												
<table border="1"> <thead> <tr> <th>Capacity Code</th> <th>Capacity Description</th> <th>Service Code</th> <th>Service Name</th> </tr> </thead> <tbody> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> <tr><td></td><td></td><td></td><td></td></tr> </tbody> </table>								Capacity Code	Capacity Description	Service Code	Service Name																																								
Capacity Code	Capacity Description	Service Code	Service Name																																																

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Capacity Code	Yes	Character (50)	–	–	Automatic by the System
Capacity Description	Yes	Character (250)	–	–	Automatic by the System
Service Code	Yes	Character (50)	–	–	Automatic by the System
Service Name	Yes	Character (250)	–	–	Automatic by the System

[illegible]

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Ability Code	Yes	Character (50)	–	–	Automatic by the System
Ability Description	Yes	Character (250)	–	–	Automatic by the System
Capacity Code	Yes	Character (50)	–	–	Automatic by the System
Capacity Description	Yes	Character (250)	–	–	Automatic by the System
Service Code	Yes	Character (50)	–	–	Automatic by the System
Service Name	Yes	Character (250)	–	–	Automatic by the System

4. Manage Ability:

This Screen contains the Tab Pages:

Internal Ability

The main characteristics of the fields for “Internal Ability” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Internal Ability Code	Yes	Character (50)	Upper	No	By the User
Internal Ability Description	Yes	Character (250)	Mixed	No	By the User
Economic Value	Yes	Number (22) with Format Mask “999,999.00”	Mixed	No	By the User
Internal Ability Type	Yes	Number(1) List Item	–	Internal (=1)	By the User
Skill Code	Yes	Character (50)	–	No	Automatic by the System
Skill Name	Yes	Character (250)	Mixed	No	By the User

External Ability

The main characteristics of the fields for “External Ability” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
External Ability Code	Yes	Character (50)	Upper	No	By the User
External Ability Description	Yes	Character (250)	Mixed	No	By the User
Economic Value	Yes	Number (22) with Format Mask “999,999.00”	–	No	By the User
External Ability Type	Yes	Number(1) List Item	–	External (=0)	By the User
Skill Code	Yes	Character (50)	–	No	Automatic by the System
Skill Name	Yes	Character (250)	Mixed	No	By the User

5. Manage Capacity:

This Screen contains the Tab Pages:

Internal Capacity

The main characteristics of the fields for “Internal Capacity” Tab Page are shown in the following table

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Internal Capacity Code	Yes	Character (50)	Upper	No	By the User
Internal Capacity Description	Yes	Character (250)	Mixed	No	By the User
Economic Value	Yes	Number (22) with Format Mask “999,999.00”	–	No	By the User
Internal Capacity Type	Yes	Number(1) List Item	–	Internal (=1)	By the User
Resource Code	Yes	Character (50)	–	No	Automatic by the System
Resource Type	Yes	Character (10) List Item	–	Physical or Legal or Procedural or Human or Technological or Financial or Datalogical	By the User
Resource Description	Yes	Character (1000)	Mixed	No	By the User

External Capacity

The main characteristics of the fields for “External Capacity” Tab Page are shown in the following table

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
External Capacity Code	Yes	Character (50)	Upper	No	By the User
External Capacity Description	Yes	Character (250)	Mixed	No	By the User
Economic Value	Yes	Number (22) with Format Mask “999,999.00”	–	No	By the User
External Capacity Type	Yes	Number(1) List Item	–	External (=0)	By the User
Resource Code	Yes	Character (50)	–	No	Automatic by the System
Resource Type	Yes	Character (10) List Item	–	Physical or Legal or Procedural or Human or Technological or Financial or Datalogical	By the User
Resource Description	Yes	Character (1000)	Mixed	No	By the User

6. Manage Business Process:

The main characteristics of the fields for “Manage Business Process” Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Process Code	Yes	Character (50)	Upper	No	Automatic by the System
Process Name	Yes	Character (250)	Mixed	No	By the User
Task Name	No	Character (1.000)	Mixed	No	By the User

7. Manage Services:

This Screen contains the Tab Pages:

Services

The main characteristics of the fields for “Services” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Service Code	Yes	Character (50)	Upper	No	Automatic by the System
Service Name	Yes	Character (250)	Mixed	No	By the User
Process Code	No	Character (50) List of Values - LOV (Validated from List)	–	Returns Process Code Values	By the User
Service Type	Yes	Number(1) List Item	–	Main (=1) or Sub (=0)	By the User
Process Name	No	Character (250)	–	Returns Process Name Values according to Process Code	Automatic by the System
Tasks	No	Character (1.000)	–	Returns Tasks Name Values according to Process Code	Automatic by the System

Hierarchies of Services

The main characteristics of the fields for “Hierarchies of Services” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Main Service	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Main Service Code and Main Service Description Values.	By the User
Sub Service	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Sub Service Code and Sub Service Description Values.	By the User

8. Manage Context:

This Screen contains the Tab Pages:

Capability is in Context

The main characteristics of the fields for “Capability is in Context” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Capability Code	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Business Capability Code (Internal & External) Values.	By the User
Context Description	Yes	Character (150) List of Values - LOV (Validated from List)	–	Returns Context Description Values.	By the User

Context

The screenshot shows the Oracle Forms Runtime interface for the 'CONTEXT_W' window. The 'MANAGE CONTEXT' tab is selected, and the 'Context' sub-tab is active. It displays a table with two columns: 'Context Code' and 'Context Description'. The table has 10 empty rows. An 'Exit Context' button is located to the right of the table.

The main characteristics of the fields for “Context” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Context Code	Yes	Character (50)	–	No	Automatic by the System
Context Description	Yes	Character (150)	Mixed	No	By the User

9. Manage Owners:

The main characteristics of the fields for “Manage Owners” Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Owner Code	Yes	Character (50)	–	No	Automatic by the System
Owner Name	Yes	Character (100)	Mixed	No	By the User

10. Manage Outputs:

This Screen contains the Tab Pages:

Output

The main characteristics of the fields for “Outputs” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Output Code	Yes	Character (50)	Upper	No	By the User
Output Name	Yes	Character (50)	Mixed	No	By the User
Capability Code	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Business Capability Code (Internal & External) and Business Capability Description Values.	By the User
Output Type	Yes	Number(1) List Item	–	Main (=1) or Sub (=0)	By the User
Economic Value	Yes	Number (22) with Format Mask “999,999.00”	–	No	By the User
Recipient Name	Yes	Character (150)	Mixed	No	By the User

Hierarchies of Outputs

The screenshot shows the Oracle Forms Runtime window titled "Oracle Forms Runtime - [OUTPUT_W]". The menu bar includes Action, Edit, Query, Block, Record, Field, Window, and Help. The toolbar contains various icons for navigation and editing. The main window is titled "MANAGE OUTPUTS" and has two tabs: "Outputs" and "Hierarchies of Outputs". The "Hierarchies of Outputs" tab is selected, showing a tree structure. The tree has two main sections: "Main Output" and "Sub Output". Each section contains a list of empty rows. A "Back to Outputs" button is located at the bottom of the tree structure.

The main characteristics of the fields for “Hierarchies of Outputs” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Main Output	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Main Output Code and Description Values.	By the User
Sub Output	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Sub Output Code and Description Values.	By the User

11. Manage Collaborations Between Capabilities:

This Screen Contains the Tab Pages:

Capabilities Collaborations

The main characteristics of the fields for “Hierarchies of Outputs” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Capability A	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Business Capability Code (Internal & External) and Business Capability Description Values.	By the User
Capability B	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Business Capability Code (Internal & External) and Business Capability Description Values.	By the User
Collaborator Connector	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Collaborator Connector Code and Description Values.	By the User

Collaborator Connector

The main characteristics of the fields for “Collaborator Connector” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Collaborator Connector Code	Yes	Character (50)	Upper	No	By the User
Collaborator Connector Type	Yes	Character (20) List Item	–	Procedure or Information or Policy	By the User
Economic Value	Yes	Number (22) with Format Mask “999,999.00”	Mixed	No	By the User

12. Manage Goals:

This Screen Contains the Tab Pages:

Capability Meets Goals

The main characteristics of the fields for “Capability Meets Goals” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Capability	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Business Capability Code (Internal & External) and Business Capability Description Values.	By the User
Goals	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Goal Code and Goal Description Values.	By the User

Goals

In this Tab Page there is one field, the Main Goal Code, that is visible only when the Goal Type field takes the value Sub. Thus the main characteristics of the fields for “Goals” Tab Page are shown in the following table:

Characteristics of fields					
FIELD NAME	Required	Data Type/Length	Case Restriction when the user fill in the field	Visible Default Values	Filled in
Goal Code	Yes	Character (50)	Upper	No	By the User
Goal Name	Yes	Character (250)	Mixed	No	By the User
Goal Type	Yes	Number (1) List Item	–	Main (=1) or Sub (=0)	By the User
Sub Goal Code	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Sub Goal Code Values.	Automatic by the system
Sub Goal Description	Yes	Character (250)	–	Automatic Returns Sub Goal Description Values.	Automatic by the system
Main Goal Code	Yes	Character (50) List of Values - LOV (Validated from List)	–	Returns Main Goal Code Values.	By the User

7.6 Chapter Summary

Summarizing, in this chapter we have worked in a fourth level of analysis of our approach, meaning we have created an external schema, which involves the designing of the appropriate User Interface for the Maritime Application. In order this to be designed we have taken into account the physical database that was created in the previous Chapter.

As we have already referred an Interface is “the presentation, communication and interaction between the user and the system”. The process of developing a User’s Interface has four major phases, in which the first deals with gathering/ analyzing user’s information, the second deals with the designing of user’s interface, the third with constructing the user’s interface and finally the fifth with validating the user’s interface ([Mandel, 2002](#)).

Designing an effective interface is important, since users judge a system by this rather than is functionality. Also a poorly designed interface can cause a user to make catastrophic errors and is the main reason why many software systems are never used. Thus when designing an interface we should take into account specific principles: user familiar, consistency, minimal of surprise, recoverability, user guidance and user diversity ([Sommerville, 2007](#)). Finally when designing an interface except from the basic criteria that must be taken into account in the software development in general, meaning Functionality, Reliability, Usability, Efficiency, Maintainability and Portability ([Bevan, 1999](#)), some extra criteria must be identified, which are Convenience, Communicativeness, Reliability and Evolvability ([Oren & Çetin, 1999](#)).

In order to the previous to be specified in this Chapter we have created a Use Case Diagram, by which we intent to describe the Main Windows of our Application and her Functional and Non-Functional requirements. In this diagram the users are presented as actors that interact with the system, use cases represent the set of tasks that actor carry out, the relationships are represented as include and extension, and the comparison of some more particular tasks are represented with generalization.

Then we have created a Hierarchy of Form diagram, in order to help us in creating the Main Menu, by which a user can be navigate between the different forms. Also we have created a Basic Flow Chart for data entry in a case of a new Business Capability existence. In more specific this chart appears all the process and decisions that a user takes, during the procedure of inserting this kind of data. Finally we have presented the Application Screen with a description of the characteristics of the fields that contains, in order a user to be familiar with the way he fill in it.

During the designing of the interface for the maritime application we can say that a good knowledge of SQL language was required. Also was required the using of UML language in order to describe the main windows, to give a hierarchy of forms and to describe the steps that a user follows in a case of a data entry. By creating the previous three kinds of diagrams we were able to understand how this application must be created, which was an important factor in order the current user interface to be characterized by the previous referred criteria. Finally except from that we were able to implement some of the constraints that were missing from the previous steps of ORM technique, which had to do with union textual constraints that discussed in a previous Chapter.

CHAPTER 8: Case Study from the Maritime Field

Structure of this Chapter

- 8.1 DMC Data Description
- 8.2 Inform the Application about a New Business Capability
- 8.3 Executing Queries
- 8.4 Removing Current Records in Specific Forms
- 8.5 Chapter Summary

This chapter deals with describing how a User may interact with the Maritime Application. Thus in Section 8.1 we give a sample of data description of the Danaos Management Consultant Company, which are focus in a specific Business Capability that this company has, the INCAP4: Maritime Compliance Capability. Then in Section 8.2 we present the procedure of informing the application about this case of Capability and in Section 8.3 we show how a user may execute specific queries in the application. In Section 8.4 we present how we removing specific records in specific forms and in Section 8.4 we present a brief summary of this chapter.

8.1 DMC Data Description

The case study was based on an enterprise from the maritime domain field, the Danaos Management Consultants (DMC), who is a software and services company specializing in maritime IT company for over 30 years and one of the three subsidiaries of Danaos Corporation. DMC capabilities have already being identified, decomposed and also examined from the aspect of their collaborations from [Loucopoulos et al \(2013\)](#). The same has being done to the services that DMC provides though these capabilities, to the business goals that meet DMC capabilities, to the business process that DMC follows and to the business context in which these capabilities exists.

As we have seeing at [Loucopoulos et al \(2013\)](#), DMC provides a variety of capabilities. However we will focus in one of them, the Internal Maritime Compliance Capability and thereafter in this section we will specially focus in giving a sample of data for this capability and for the other Capabilities only where is needed.

Thus as already being described by [Loucopoulos et al \(2013\)](#), DMC Company has created four main internal capabilities and three main external capabilities which are discomposed into sub-capabilities. A codification and description of those capabilities is given in the following table:

Attributes	Business Capability Code (capcode)			
	Capability Codification (mcapcode)	Capabilities Description (capdescr)	Sub-Capability Codification (subcapcode)	Sub-Capabilities Description (capdescr)
Internal Capability (incapcode)	INCAP1	Maritime Management Capability	INCAP1.1 INCAP1.2 INCAP1.3 INCAP1.4 INCAP1.5 INCAP1.6 INCAP1.7	Ship Financial Management Ship Technical Management Ship Procurement Management International Safety Management Human Resource Management Chartering Management Operation Management
	INCAP2	Social Networking Capability	INCAP2.1 INCAP2.2	Secure Transactions and Communications Capability Marketing Capability
	INCAP3	Information Store & Management Capability	INCAP3.1 INCAP3.2	Information Storing Information Management
	INCAP4	Maritime Compliance Capability	INCAP4.1 INCAP4.2 INCAP4.3	Vessel Monitoring Capability Port Regulation Monitoring Capability Regulation Inconsistences Reporting Capability
External Capability (excapcode)	EXCAP1	Technical Assistance Management Capability		
	EXCAP2	Web-Conference Management Capability		
	EXCAP3	Maritime BP Outsourcing Capability		

Those Capabilities are associated with specific owners as shown in the following tables:

Owner	
Owner Codification (ownercode)	Owner Name (ownername)
OWN1	DMC
OWN2	ComSys
OWN3	Microsoft
OWN4	Danaos Services/India

Attributes	Main Capability Codification (mcapcode)	Main Capabilities Description (capdescr)	Sub-Capability Codification (subcapcode)	Sub-Capabilities Description (capdescr)	Owner Codification (ownercode)	Owner Name (ownername)
Internal Capability (incapcode)	INCAP1	Maritime Management Capability	INCAP1.1	Ship Financial Management	OWN1	DMC
			INCAP1.2	Ship Technical Management	OWN1	DMC
			INCAP1.3	Ship Procurement Management	OWN1	DMC
			INCAP1.4	International Safety Management	OWN1	DMC
			INCAP1.5	Human Resource Management	OWN1	DMC
			INCAP1.6	Chartering Management	OWN1	DMC
			INCAP1.7	Operation Management	OWN1	DMC
	INCAP2	Social Networking Capability	INCAP2.1	Secure Transactions and Communications Capability	OWN1	DMC
			INCAP2.2	Marketing Capability	OWN1	DMC
	INCAP3	Information Store & Management Capability	INCAP3.1	Information Storing	OWN1	DMC
			INCAP3.2	Information Management	OWN1	DMC
	INCAP4	Maritime Compliance Capability	INCAP4.1	Vessel Monitoring Capability	OWN1	DMC
			INCAP4.2	Port Regulation Monitoring Capability	OWN1	DMC
			INCAP4.3	Regulation Inconsistences Reporting Capability	OWN1	DMC
External Capability (excacode)	EXCAP1	Technical Assistance Management Capability			OWN2	ComSys
	EXCAP2	Web-Conference Management Capability			OWN3	Microsoft
	EXCAP3	Maritime BP Outsourcing Capability			OWN4	Danaos Services/India

One of the main internal capabilities of DMC, as presented in the previous tables is that of Maritime Compliance Capability (INCAP4). DMC has created and maintaining this capability because in maritime industry it is essential for every shipping company and each vessel to conform to all of required regulations and rules at each port, which often differ and involving a large number of documents. The failure to comply with these may affect in serious way the operation of the shipping company. Thus analyzing systems and processes that deals with that data is important for that capability.

As already being described by [Loucopoulos et al \(2013\)](#) Maritime Compliance Capability, owned by DMC (OWN1) and has three sub-capabilities. Those are the Vessel Monitoring Capability (INCAP4.1), the Port Regulation Monitoring Capability (INCAP4.2) and the Regulation Inconsistences Reporting Capability (INCAP4.3). The first one deals with the ability to monitor the data concerning vessel's status (cargo, medical conditions, emissions and environmental issues), the second one deals with the effort of the vessel to exchange information with the port and be aware of regulations enforced by the specific port authority, and the last one deals with the ability to comparing vessel's data with related to them regulations and create alerts in case of non-compliance.

In order DMC to deliver Maritime Compliance Capability collaborations with other capabilities may exist through the exchange of data, the execution of a business process or the sharing of a specific data, which collaborations has some economic values. More especially Vessel Monitoring Capability (INCAP4.1) collaborates with Human Resource Management (INCAP1.5) through information and collaborates with Operation Management (INCAP1.7) through procedure. The Port Regulation Monitoring Capability (INCAP4.2) collaborates with International Safety Management (INCAP1.4) through information. Finally the Regulation Inconsistences Reporting Capability (INCAP4.3) collaborates with Operation Management (INCAP1.7) through information. Sample of data for collaborations in Maritime Compliance Capability are given in the following tables:

Collaboration Connector (collabcode)		
Procedure (collabcode)	Information (collabcode)	Policy (collabcode)
PR1	IN1	PO1
PR2	IN2	PO2
PR3	IN3	PO3
PRn	Inn	POn

Business Capability		collaborates with Business Capability		through	
Codification (capcode1)	Sub-Capabilities Description (capdescr)	Codification (capcode2)	Sub-Capabilities Description (capdescr)	Collaboration Connector Code (collabcode)	(connectortype)
INCAP4.1	Vessel Monitoring Capability	INCAP1.5 INCAP1.7	Human Resource Management Operation Management	IN1 PR1	INFORMATION PROCEDURE
INCAP4.2	Port Regulation Monitoring Capability	INCAP1.4	International Safety Management	IN2	INFORMATION
INCAP4.3	Regulation Inconsistences Reporting Capability	INCAP1.7	Operation Management	IN3	INFORMATION

Also Maritime Compliance Capability (INCAP4) is related to a specific context within it exists. The context of Maritime Compliance Capability concerns changes in maritime regulations, in laws enforced from port authorities and in vessel's status. A codification of the context related to Maritime Compliance Capability is shown in the following tables:

Context	
Codification (contcode)	Context Description (contdescr)
CONT1	Local Legislations
CONT2	Port Authorities Regulations
CONT3	Vessel's status

Business Capability		Is in context	
Codification (capcode)	Capabilities Description (capdescr)	Codification (contcode)	Context Description (contdescr)
INCAP4	Maritime Compliance Capability	CONT1 CONT2 CONT3	Local Legislations Port Authorities Regulations Vessel's status
INCAP4.1	Vessel Monitoring Capability	CONT3	Vessel's status
INCAP4.2	Port Regulation Monitoring Capability	CONT2	Port Authorities Regulations
INCAP4.3	Regulation Inconsistences Reporting Capability	CONT1 CONT2 CONT3	Local Legislations Port Authorities Regulations Vessel's status

Furthermore Maritime Compliance Capability (INCAP4) is related to top – level strategic goals, while its sub-capabilities are related to lower level operational goals. Worth mentioning that when depicting in a goal graph the business goals of an organization the top – level goals is the strategic goals, while the high – level is the operational goals. Thus the top – level goals of Maritime Compliance Capability are:

- Goal 9: To participate in research projects
- Goal 10: To collaborate with academic research
- Goal 13: To identify client' s needs
- Goal 20: To comply with regulations

In order the company to fulfill the “Goal 20: To comply with regulations” then it has its sub-capabilities to fulfill the low level operational goals:

- Goal 41: To monitor vessel status
- Goal 42: To be informed about the regulation of each port
- Goal 43: To get alert when regulation are not met

A codification of business goals related with Maritime Compliance Capability is shown in the following tables:

Business Goal (goalcode)			
Main Goal Codification (mgoalcode)	Main Goal Name (goalname)	Sub-Goal Codification (subgoalcode)	Sub-Goal Name (goalname)
INCAP4_GOAL1	Goal 9: To participate in research projects		
INCAP4_GOAL2	Goal 10: To collaborate with academic research		
INCAP4_GOAL3	Goal 13: To identify client' s needs		
INCAP4_GOAL4	Goal 20: To comply with regulations	INCAP4_GOAL4.1	Goal 41: To monitor vessel status
		INCAP4_GOAL4.2	Goal 42: To be informed about the regulation of each port
		INCAP4_GAOL4.3	Goal 43: To get alert when regulation are not met

Business Capability		Meets Business goal	
Codification (capcode)	Capabilities Description (capdescr)	Codification (goalcode)	Goal Description (goaldescr)
INCAP4	Maritime Compliance Capability	INCAP4_GOAL1 INCAP4_GOAL2 INCAP4_GOAL3 INCAP4_GOAL4	Goal 9: To participate in research projects Goal 10: To collaborate with academic research Goal 13: To identify client' s needs Goal 20: To comply with regulations
INCAP4.1	Vessel Monitoring Capability	INCAP4_GOAL4.1	Goal 41: To monitor vessel status
INCAP4.2	Port Regulation Monitoring Capability	INCAP4_GOAL4.2	Goal 42: To be informed about the regulation of each port
INCAP4.3	Regulation Inconsistences Reporting Capability	INCAP4_GOAL4.3	Goal 43: To get alert when regulation are not met

In addition Maritime Compliance Capability (INCAP4) produces some Business Output, meaning some services which are of economic value and received by some recipients. One main output that Maritime Compliance Capability produces is that of Rule Compliance Services, which has as sub-output the Service Monitoring Service, the Vessel Monitoring Service and the Regulation Inconsistence Service. A codification of output and sub-output in relation with Maritime Compliance Capability is shown in the following tables:

Business Output (outputcode)			
Main Output Codification (moutputcode)	Main Output Name (outputname)	Sub-Output Codification (suboutputcode)	Sub-Output Name (outputname)
INCAP4_OUTPUT1	Rule Compliance Services	INCAP4_OUTPUT1.1	Vessel Monitoring Services
		INCAP4_OUTPUT1.2	Port Regulation Services
		INCAP4_OUTPUT1.3	Regulation Inconsistencies Reporting Services

Business Capability		Delivers Business Output	
Codification (capcode)	Capabilities Description (capdescr)	Codification (outputcode)	Output Name (outputname)
INCAP4	Maritime Compliance Capability	INCAP4_OUTPUT1	Rule Compliance Services
INCAP4.1	Vessel Monitoring Capability	INCAP4_OUTPUT1.1	Vessel Monitoring Services
INCAP4.2	Port Regulation Monitoring Capability	INCAP4_OUTPUT1.2	Port Regulation Services
INCAP4.3	Regulation Inconsistencies Reporting Capability	INCAP4_OUTPUT1.3	Regulation Inconsistencies Reporting Services

Also DMC for the total of capabilities produces different type of services and sub-services, which are shown in the following table:

Service (servcode)			
Main Service Codification (mservcode)	Main Service Name (servname)	Sub – Service Codification (subservcode)	Sub-Service Name (servname)
SERV1	Danaos Enterprise Maritime Solutions (DEMS)	SERV1.1	Ship Management System
		SERV1.2	Commercial Operation Management System
		SERV1.3	Financial Management System
		SERV1.4	Optimal Routing System
		SERV1.5	Integrated Communications Package
		SERV1.6	Fleet Performance Monitoring System (WAVES)
		SERV1.7	KPIs Monitoring System
SERV2	Social Platform		
SERV3	Mobile Apps		
SERV4	Outsourcing		
SERV5	E-Compliance System	SERV5.1	Port of Calls Application

From the previous services worth mentioning that Port of Calls Application as described by Loucopoulos et al (2013) was designed for Port Authorities and Ship Management Companies in order to ease the submission procedures of required compliance documents. As they stated, this application utilizes a single platform with a comprehensive user interface, a build-in rule specific scripting language, and a storage facility ready to

directly read and write various files ready for automation & orchestration of data flow. Also it is capable of collecting data form excel forms in one database and compare the data retrieved from vessel and ports with existing rules in Database. If there are inconsistencies between data and rules, the application creates alerts via an alert mechanism in order for the vessel to take care of them and align to the current constraints and rules by preparing the required paperwork by hand.

In order to deliver the previous services to its clients it has establish a standard Business Process, followed by specific steps/tasks. This Business Process is the Business Process for Service Request and Quality Control. Except form the previous DMC has establish a specific Business Process in order to satisfy customer requirements in the case of Maritime Compliance Capability (INCAP4), the Business Process for Compliance Monitoring, which also follows specific steps/tasks. A codification of them in relation to services is shown in the following tables:

Business Process		Leads to Tasks
Business Process Codification (processcode)	Business Process Name (processname)	Tasks (taskname)
BP1	Business Process for Service Request and Quality Control	<ol style="list-style-type: none"> 1. Service request initiates an offer from the Sales & Purchase dept 2. Both parties are discussing the offer and signing the final Contract 3. The requirement analyst is studying and analyzing the requirements of the client and then conclude to a project plan of development with the assistance of software engineers 4. Software engineers develop the request service and adjust their development to the given specification 5. Software engineers proceed to an integration test and if everything is working fine they deliver the protocol in order for the client to work with the module for a trial period 6. If anything is missing or malfunctioning then the company is obligated to review the development stage and the requirement analysis 7. The company offers consistent and permanent maintenance of the derived service to their clientele
BP2	Business Process for Compliance Monitoring	Tasks (manual and user tasks): <ul style="list-style-type: none"> ▪ Collect forms with vessel's status ▪ Collect forms with cargo status ▪ Collect forms with crew data ▪ Collect forms with history data
		Tasks (manual and user tasks): <ul style="list-style-type: none"> ▪ Insert to the form port's required information ▪ Prepare actual port forms ▪ Import forms to the system ▪ Save the complete forms for future use
		Service tasks (executed by the system): <ul style="list-style-type: none"> ▪ Compare imported data with existing of port
		Service tasks (executed by the system): <ul style="list-style-type: none"> ▪ Create alerts

Service (servicecode)				Is delivered by Business Process	
Main Service Codification (mservcode)	Main Service Name (servname)	Sub – Service Codification (subservcode)	Sub-Service Name (servname)	Business Process Code (processcode)	Business Process Name (processname)
SERV1	Danaos Enterprise Maritime Solutions (DEMS)	SERV1.1	Ship Management System	BP1	Business Process for Service Request and Quality Control
		SERV1.2	Commercial Operation Management System		
		SERV1.3	Financial Management System		
		SERV1.4	Optimal Routing System		
		SERV1.5	Integrated Communications Package		
		SERV1.6	Fleet Performance Monitoring System (WAVES)		
		SERV1.7	KPIs Monitoring System		
SERV2	Social Platform			BP1	Business Process for Service Request and Quality Control
SERV3	Mobile Apps			BP1	Business Process for Service Request and Quality Control
SERV4	Outsourcing			BP1	Business Process for Service Request and Quality Control
SERV5	E-Compliance System	SERV5.1	Port of Calls Application	BP1	Business Process for Service Request and Quality Control
				BP2	Business Process for Compliance Monitoring

In order to deliver this service Maritime Compliance Capability must have the ability and the capacity to ease the submission procedures of required compliance documents for the Port of Calls Application, which have an economic value. However Ability may be either internal or external. The same exist to Capacity. Thus a general codification for describing those object types may be according to the following tables:

Attributes	Ability	
	Ability Codification (abcode)	Ability Description (abdescr)
Internal Ability (inabcode)	INCAP1_INAB1
	INCAP1_INABn
	INCAP2_INAB1
	INCAP2_INABn
	INCAPn_INAB1
	INCAPn_INABn
External Ability (exabcode)	INCAP1_EXAB1
	INCAP1_EXABn
	INCAP2_EXAB1
	INCAP2_EXABn
	INCAPn_EXAB1
	INCAPn_EXABn

Attributes	Capacity	
	Capacity Codification (capaccode)	Capacity Description (capacdescr)
Internal Resource Set (incapaccode)	INCAP1_INRES1
	INCAP1_INRESn
	INCAP2_INRES1
	INCAP2_INRESn
	INCAPn_INRES1
	INCAPn_INRESn
External Resource Set (excapaccode)	INCAP1_EXRES1
	INCAP1_EXRESn
	INCAP2_EXRES1
	INCAP2_EXRESn
	INCAPn_EXRES1
	INCAPn_INRESn

As far for Maritime Compliance Capability (INCAP4) in order to produces its services it uses some Ability and some Capacity. From the previous description of services, Maritime Compliance Capability produces the E-Compliance System (SERV5) and thereafter the Port of Calls Application (SERV5.1). Thus a sample of data and a codification of the previous are shown in the following tables:

Internal Capability		Uses Ability		For Service
Codification (incapcode)	Capabilities Description (capdescr)	Codification (abcode)	Ability Description (abdescr)	Service Code (servcode)
INCAP4	Maritime Compliance Capability	INCAP4_INAB1	The ability to ease the submission procedures of required compliance documents for the Port of Calls Application	SERV5.1
INCAP4.1	Vessel Monitoring Capability	INCAP4.1_INAB1	The ability for vessel monitoring	SERV5.1
INCAP4.2	Port Regulation Monitoring Capability	INCAP4.2_INAB1	The ability for Port Regulations monitoring	SERV5.1
INCAP4.3	Regulation Inconsistences Reporting Capability	INCAP4.3_INAB1	The ability for regulating inconsistencies of reporting	SERV5.1

Internal Capability		Uses Capacity		For Service
Codification (incapcode)	Capabilities Description (capdescr)	Capacity Code (capaccode)	Capacity Description (capacdescr)	Service Code (servcode)
INCAP4	Maritime Compliance Capability	INCAP4_INRES1	The capacity to ease the submission procedures of required compliance documents for the Port of Calls Application	SERV5.1
INCAP4.1	Vessel Monitoring Capability	INCAP4.1_INRES1	The capacity for vessel monitoring	SERV5.1
INCAP4.2	Port Regulation Monitoring Capability	INCAP4.2_INRES1	The capacity for Port Regulations monitoring	SERV5.1
INCAP4.3	Regulation Inconsistences Reporting Capability	INCAP4.3_INRES1	The capacity for regulating inconsistencies of reporting	SERV5.1

The Internal Ability that uses the Maritime Compliance Capability and its sub capabilities in order to produce a service is made of a skill type. A codification of skill type is relation with ability is shown in the following tables:

Skill Type	
Codification (skillcode)	Skill Name (skillname)
SK1	IT skills in Microsoft Office (word, excel, access, power point, internet)
SK2	Daily user of Databases Management Systems
SK3	Familiarization in working at Cloud environment's
SK4	Successfully worked to strict deadlines
SK5	Bachelor Degree in Computer Software Engineering
SK6	Master Degree in Project Management
SKn

Internal Ability		Is made of Skill Type	
Codification (abcode)	Ability Description (abdescr)	Codification (skillcode)	Skill Name (skillname)
INCAP4_INAB1	The ability to ease the submission procedures of required compliance documents for the Port of Calls Application	SK1	IT skills in Microsoft Office (word, excel, access, power point, internet)
		SK2	Daily user of Databases Management Systems
		SK3	Familiarization in working at Cloud environment's
		SK4	Successfully worked to strict deadlines
		SK5	Bachelor Degree in Computer Software Engineering
		SK6	Master Degree in Project Management

The capacity that uses the Maritime Compliance Capability and its sub capabilities is made of Resources. Resources may be Physical, Legal, Procedural, Human, Technological, Financial and Datological. A general codification of resources and their relation with Capacity is shown in the following tables:

	Capacity		Is made of Resources		
	Capacity Codification (capaccode)	Capacity Description (capacdescr)	Resources Codification (rescode)	Resource Type Values (restype)	Resources Description (ph_descr or le_descr or pr_descr or hu_descr or te_descr or fi_descr or da_descr)
Internal Resource Set (inresourcesetcode)	INCAP1_INRES1	PH_INRES1.1	PH
			LE_INRES1.1	LE
			PR_INRES1.1	PR
			HU_INRES1.1	HU
			TE_INRES1.1	TE
			FI_INRES1.1	FI
			DA_INRES1.1	DA
	INCAP1_INRESn	PH_INRES1.n	PH
			LE_INRES1.n	LE
			PR_INRES1.n	PR
			HU_INRES1.n	HU
			TE_INRES1.n	TE
			FI_INRES1.n	FI
			DA_INRES1.n	DA
	INCAP2_INRES1	PH_INRES2.1	PH
			LE_INRES2.1	LE
			PR_INRES2.1	PR
			HU_INRES2.1	HU
			TE_INRES2.1	TE
			FI_INRES2.1	FI
			DA_INRES2.1	DA
	INCAPn_INRES1	PH_INRESn.1	PH
			LE_INRESn.1	LE
			PR_INRESn.1	PR
			HU_INRESn.1	HU
			TE_INRESn.1	TE
			FI_INRESn.1	FI
			DA_INRESn.1	DA
	INCAPn_INRESn	PH_INRESn.n	PH
			LE_INRESn.n	LE
			PR_INRESn.n	PR
			HU_INRESn.n	HU
			TE_INRESn.n	TE
			FI_INRESn.n	FI
			DA_INRESn.n	DA
External Resource Set (exresourcesetcode)	INCAP1_EXRES1	PH_EXRES1.1	PH
			LE_EXRES1.1	LE
			PR_EXRES1.1	PR
			HU_EXRES1.1	HU
			TE_EXRES1.1	TE
			FI_EXRES1.1	FI
			DA_EXRES1.1	DA
	INCAP1_EXRESn	PH_EXRES1.n	PH
			LE_EXRES1.n	LE
			PR_EXRES1.n	PR
			HU_EXRES1.n	HU
			TE_EXRES1.n	TE
			FI_EXRES1.n	FI
			DA_EXRES1.n	DA
	INCAP2_EXRES1	PH_EXRES2.1	PH
			LE_EXRES2.1	LE
			PR_EXRES2.1	PR
			HU_EXRES2.1	HU
			TE_EXRES2.1	TE
			FI_EXRES2.1	FI
			DA_EXRES2.1	DA
	INCAPn_EXRES1	PH_EXRESn.1	PH
			LE_EXRESn.1	LE
			PR_EXRESn.1	PR

	INCAPn_EXRESn	HU_EXRESn.1	HU
			TE_EXRESn.1	TE
			FI_EXRESn.1	FI
			DA_EXRESn.1	DA
			PH_EXRESn.n	PH
			LE_EXRESn.n	LE
			PR_EXRESn.n	PR
			HU_EXRESn.n	HU
			TE_EXRESn.n	TE
			FI_EXRESn.n	FI
			DA_EXRESn.n	DA

Finally DMC in order to deliver the Maritime Compliance Capability (INCAP4) has used a set of internal resources, which are Human, Technological & Legal. Those resources are described in the following table:

Internal Capability		Uses Capacity				
Codification (incapcode)	Capabilities Description (capdescr)	Capacity Code (capacccode)	Capacity Description (capacdescr)	Resources Codification (rescode)	Resource Type Values (restype)	Resources Description (resdescr)
INCAP4	Maritime Compliance Capability	INCAP4_INRES1	The capacity to ease the submission procedures of required compliance documents for the Port of Calls Application	HU_INRES4.1	Human	<ul style="list-style-type: none"> 2 Software Engineers from IT department 1 Project Manager
				TE_INRES4.1	Technological	<ul style="list-style-type: none"> RED Programming Language Reading and Learning Machinery of Word, Excel, PDF etc files Web Services for retrieval of data A database in order to store laws and regulations
				LE_INRES4.1	Legal	<ul style="list-style-type: none"> Laws and regulations in order to create a Database with rules and constraints

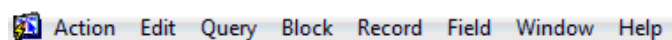
8.2 Inform the Application about a New Business Capability

Taking into account the Case Study discussed in the previous section we now will inform the application with the information about the INCAP4: Maritime Compliance Capability. A guide for this action will be the flow chart that discussed 7.3, since we are inserting a New Business Capability

Before we start it is necessary to discuss the **Main Menu** that exists in the Top of every Page in our Application. This is the following Menu:

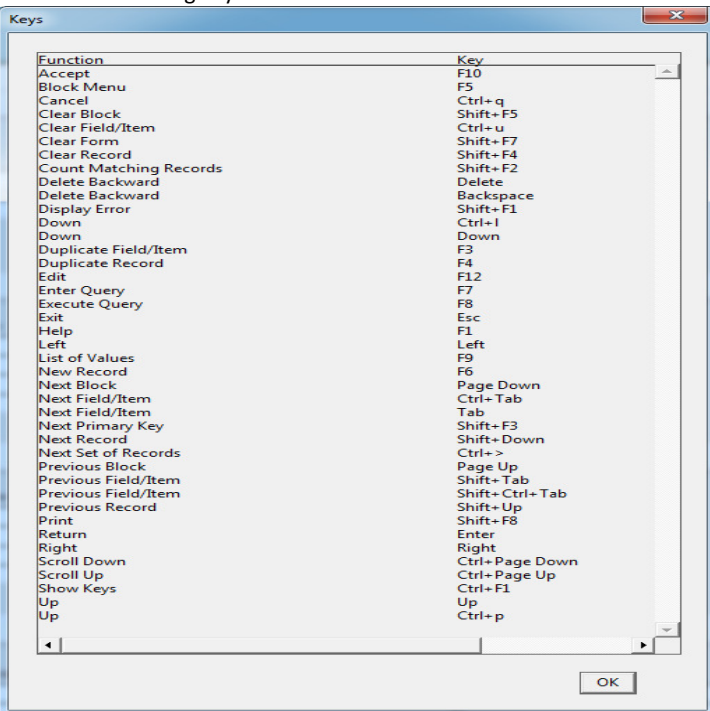


This menu contains the **Menu Items**:

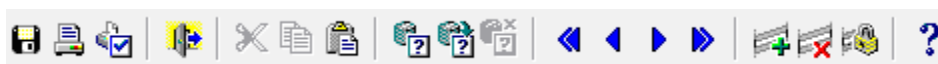


If we choose one of this items then a list of choices will be presented, by which we can implement specific actions. Those actions described thereafter:



















Action	Save	:	Saves any pending changes in the active form
	Clear All	:	Clears all the records of the current form
	Print	:	Prints the current window
	Print Setup	:	Sets up the printing choices
	Exit	:	Quits the current Application
Edit	Cut	:	Cuts the current field to the clipboard
	Copy	:	Copies the current field to the clipboard
	Paste	:	Pastes the contexts of the clipboard into current field
	Edit	:	Makes changes to a current field
	Display List	:	Appears a list of values in a current field
Query	Enter	:	Enters a new query by a specific record or for all records
	Execute	:	Executes a query by a specific record or for all records
	Cancel	:	Cancels an entered query
	Last Criteria	:	Enters a query according to the last used querying criteria
	Count Hits	:	Counts the number of records appeared in a query
	Fetch Next Set	:	Appears the next set of records
Block	Previous	:	Navigates to the previous block
	Next	:	Navigates to the next block
	Clear	:	Clears the records in the current block
Record	Previous	:	Navigates to the previous record
	Next	:	Navigates to the next record
	Scroll Up	:	Navigates to the first record
	Scroll Down	:	Navigates to the last record
	Insert	:	Inserts a new record
	Remove	:	Removes the current record
	Lock	:	Locks the current record
	Duplicate	:	Duplicates the current record
	Clear	:	Clears the current record
Field	Previous	:	Navigates to the previous field
	Next	:	Navigates to the next field
	Clear	:	Clears the current field
	Duplicate	:	Duplicates the current field
Window	Cascade	:	Displays any open windows in a "cascaded" or stair-stepped fashion
	Tile	:	Displays any open window in a tile fashion
	Arrange Icons	:	Specifies how to arrange icons
	Visible Window	:	Shows the current opened window

Help	Help	:	Shows the properties of the current block
	Keys	:	Shows the following keys:
			
	Display Error	:	Displays any encountered errors of the system during the operation of the application
	Debug:	:	Finds and resolves of defects that prevents correct operation of the application


The Main Menu also contains the following **Tool Bar**:

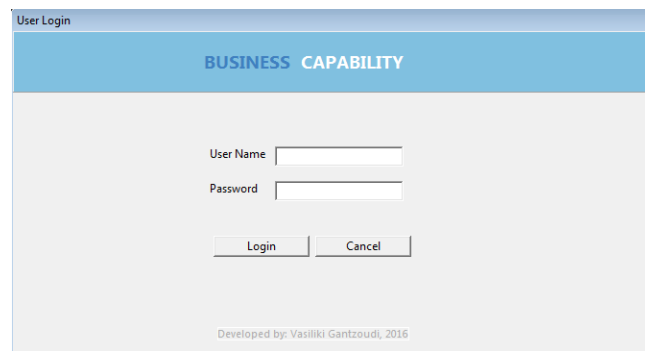


This Tool Bar is a collection of buttons that we may select in order to perform some of the actions described in the Menu Items directly. Those actions are:


	:	Save any pending changes in the active form		:	Cancel an entered query
	:	Set up the printing choices		:	Navigation to the previous block
	:	Print the current window		:	Navigation to the next block
	:	Quit the current Application		:	Navigation to the previous record
	:	Cut the current field to the clipboard		:	Navigation to the next record
	:	Copy the current field to the clipboard		:	Insert a new record
	:	Paste the contexts of the clipboard into current field		:	Remove the current record
	:	Enter a new query by a specific record or for all records		:	Lock the current record
	:	Execute a query by a specific record or for all records		:	Show the properties of the current block

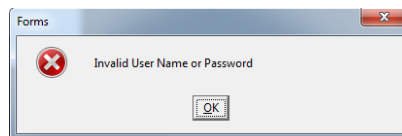


Thus for opening the Application we double click on the Business Capability Icon  , and then the following window appears:



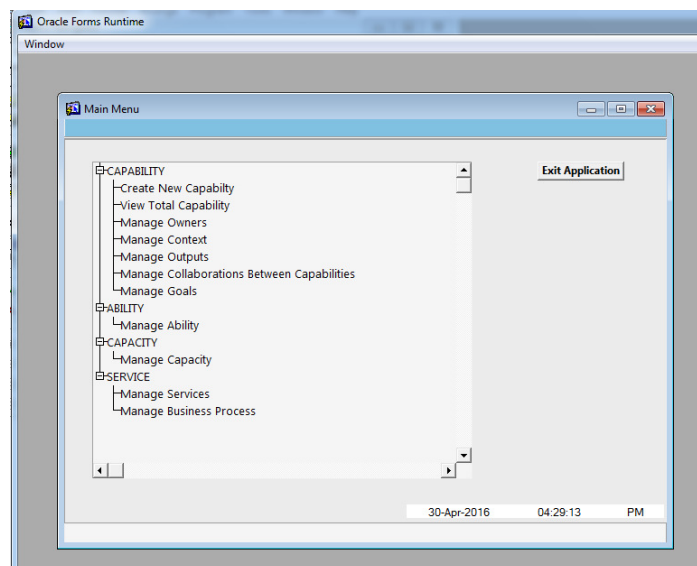
The 'User Login' window has a blue header with 'BUSINESS CAPABILITY'. It contains two input fields: 'User Name' and 'Password'. Below them are 'Login' and 'Cancel' buttons. At the bottom, it says 'Developed by: Vasiliki Gantzoudi, 2016'.

In this window we enter the User Name and the Password, and we press the button . If during the filling in we type an incorrect User Name or Password then the System appears the following message:

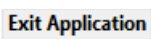


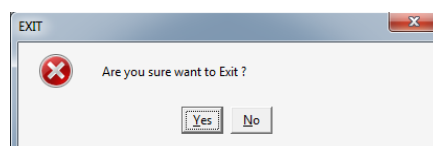
A small dialog box titled 'Forms' with a red 'X' icon. The text inside says 'Invalid User Name or Password'. There is an 'OK' button at the bottom.

Otherwise the Main Menu appears. With this menu we can navigate between the different Pages of the Application, by clicking on them.






The 'Main Menu' window shows a tree structure of application pages. The tree is expanded to show the following items: CAPABILITY (with sub-items: Create New Capability, View Total Capability, Manage Owners, Manage Context, Manage Outputs, Manage Collaborations Between Capabilities, Manage Goals), ABILITY (with sub-item: Manage Ability), CAPACITY (with sub-item: Manage Capacity), and SERVICE (with sub-items: Manage Services, Manage Business Process). There is an 'Exit Application' button in the top right corner. The status bar at the bottom shows '30-Apr-2016 04:29:13 PM'.

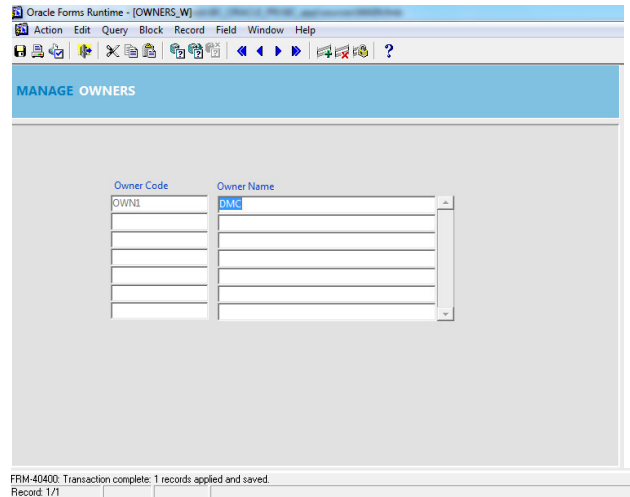
Also in this Page if we press the button  then this message appears:




A small dialog box titled 'EXIT' with a red 'X' icon. The text inside says 'Are you sure want to Exit ?'. There are 'Yes' and 'No' buttons at the bottom.

If we choose yes then we exit the from Business Capability Application, otherwise no action occurs.

The INCAP4 has a pre-existing Owner with value “DMC”, which means that according to Basic Flow Chart we continue with the next step. However it is necessary to describe how we have inserted this Owner. So we suppose that INCAP4 has a New Owner with value “DMC”. Thus we are choosing from the Main Menu the Form “Manage Owners”. In the appeared Page we press the button Insert New Record (), we place the cursor in the field Owner Name and enter the value “DMC. Then we press the Save button () and finally the button Exit Form (). Worth mentioning that in this Page the system automatically creates the Owner Code in the logic that discussed in the previous section (meaning OWN1, OWN2, OWNn). Also when we press the Save button the message “FRM-40400: Transaction complete: 1 records applied and saved” appears in the bottom of the page, which confirms the success of this action.



Continuing with the flow chart the INCAP4 is an Internal Capability and also has a New Ability, which is “INCAP4_INAB1: The Ability to ease the submission procedures of required compliance documents for the Port of Calls Application”. This Ability is an Internal Ability, so we are choosing from the Main Menu the Form “Manage Ability” and we are going to the Tab Page “Internal Ability”. In this Tab Page we press the button Insert New Record () and then:

- We place the cursor in the field Internal Ability Code and enter the code with the logic that described in the previous section, meaning “INCAP4_INAB1”. When we are entering this code the system automatically converts the characters we use into Upper Case Characters.
- We place the cursor in the field Internal Ability Description and we enter the information “The Ability to ease the submission procedures of required compliance documents for the Port of Calls Application”.
- We place the cursor in the field Economic Value and we enter the information “15.000” (for this field the system show final values in a Format Mask 99,999.00).

- We place the cursor in the field Internal Ability Type, we click to display values and we choose the correct value, meaning the value “Internal”.
- We place the cursor in the field Skill Name and we enter the information about the Skills that Defines the Ability INCAP4_INAB1. Thus we enter the records:

1. «IT skills in Microsoft Office (word, excel, access, power point, internet)»
2. «Daily user of Databases Management Systems»
3. «Familiarization in working at Cloud environment's»
4. «Successfully worked to strict deadlines»
5. «Bachelor Degree in Computer Software Engineering»
6. «Master Degree in Project Management»

Skill Code	Skill Name
SK1	IT skills in Microsoft Office (word, excel, access, power point, internet)
SK2	Daily user of Databases Management Systems
SK3	Familiarization in working at Cloud environment's
SK4	Successfully worked to strict deadlines
SK5	Bachelor Degree in Computer Software Engineering
SK6	Master Degree in Project Management

- Then we press the Save Button (💾) and finally the button Exit Form (🚪).

Worth mentioning that the system automatically creates the Skill Code in the logic that discussed in the previous section (meaning SK1, SK2, SKn).

NOTE: If this New Ability was an External Ability, then we would choose the Tab Page “External Ability” and fill in the same fields with the same way we have done it in “Internal Ability” Tab Page.

We now continue with the next step. Thus the INCAP4 has a New Capacity, which is “INCAP4_INRES1: The Capacity to ease the submission procedures of required compliance documents for the Port of Calls Application”. This is an Internal Capacity, so we are choosing from the Main Menu the Form “Manage Capacity” and we are going to the Tab Page “Internal Capacity”. In this Tab Page we press the button Insert New Record (➕) and then:

- We place the cursor in the field Internal Capacity Code and we enter the code with the logic that described in the previous section, meaning INCAP4_INRES1. When

we are entering this code the system automatically converts the characters we use into Upper Case Characters.

- We place the cursor in the field Internal Capacity Description and we enter the information “The Capacity to ease the submission procedures of required compliance documents for the Port of Calls Application”.
- We place the cursor in the field Economic Value and we enter the information “25.000” (for this field the system show final values in a Format Mask 99,999.00).
- We place the cursor in the field Internal Capacity Type, we click to display values and finally we choose the correct value, meaning the value “External”.

The INCAP4_INRES1 has used three set of Resources, meaning Human, Technological and Legal. Thus for the first set of Resources:

- We place the cursor in the first record to the field Resource Type, we click to display values and we choose the value ‘Human’.
- We place the cursor in the field Resource Description and we enter the information:

- 2 Software Engineers from IT department
- 1 Project Manager

Also for the second set of Resources:

- We place the cursor in the next record to the field Resource Type, we click to display values and we choose the value ‘Technological’.

- We place the cursor in the field Resource Description and we enter the information:

- RED Programming Language,
- Reading and Learning Machinery of Word, Excel, PDF etc files,
- Web Services for retrieval of data,
- A database in order to store laws and regulations.

Finally for the third set of Resources:


- We place the cursor in the third record to the field Resource Type, we click to display values and we choose the value 'Legal'.

- We place the cursor in the field Resource Description and we enter the information: a) Laws and regulations in order to create a Database with rules and constraints.

- Finally we press the Save Button (💾) and finally the button Exit Form (🚪).

Worth mentioning that the system automatically creates the Resource Code in the logic that discussed in the previous section, meaning HU_INRESR4.1 for Human Resource Set, TE_INRESR4.1 for Technological Resource Set and LE_INRESR4.1 for Legal Resource Set.

NOTE: If this New Capacity was an External Capacity, then we would choose the Tab Page "External Capacity" and fill in the same fields with the same way we have done it in "Internal Capacity" Tab Page.

We now continue with the next Step. The INCAP4 delivers a New Service which delivered by a Specific Business Process. As we have already disused in the previous section DMC has established a specific Business Process in order to satisfy customer requirements in the case of Maritime Compliance Capability (INCAP4) the Business Process for Compliance Monitoring. So we are choosing from the Main Menu the Form “Manage Business Process”. In the appeared Page we press the button Insert New Record (), and then:

- We place the cursor in the field Process Name and we enter the information “Business Process for Compliance Monitoring”.
- We place the cursor in the field Task Name and we enter the information:

A. Tasks (manual and user tasks):

- Collect forms with vessel’s status
- Collect forms with cargo status
- Collect forms with crew data
- Collect forms with history data

B. Tasks (manual and user tasks):

- Insert to the form port’s required information
- Prepare actual port forms
- Import forms to the system
- Save the complete forms for future use

Service tasks (executed by the system):

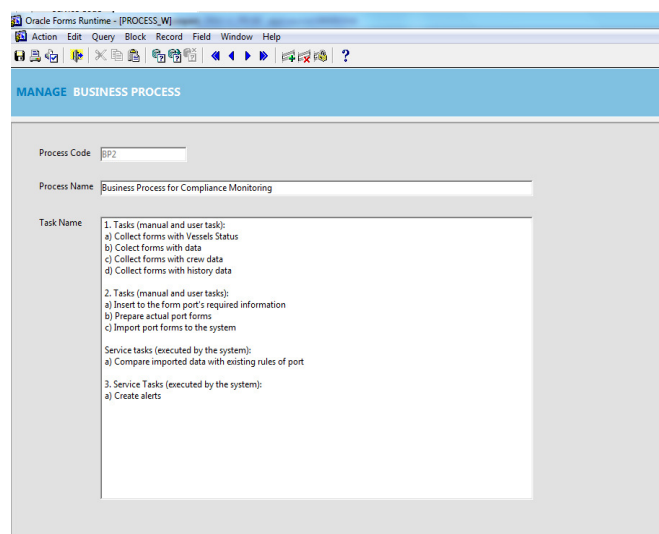
- Compare imported data with existing of port

C. Tasks (manual and user tasks):


- Create alerts


- Then we press the Save Button () and then the Exit Form Button ().


Worth mentioning that in this page the system automatically creates the Process Code in the logic that discussed in the previous section (meaning B1, B2, Bn).



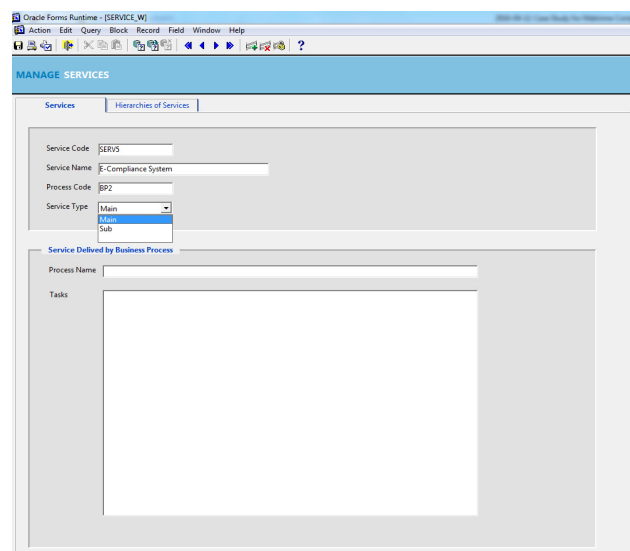
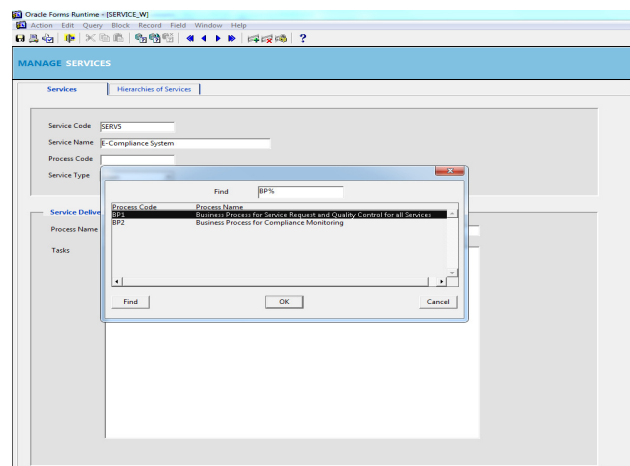
As we already mention this Business Process defines a Specific Service so we continue with step. Thus we are choosing from the Main Menu the Form “Manage Services” and we are going to the Tab Page “Services”. The INCAP4 capability delivers the Main Service “SERV5: E-Compliance System” which has as Sub Service the “SERV 5.1: Port of Calls Application”. Thus in this Tab Page we first insert the information about Main Service and then we insert the information about the Sub Service.

Thus for inserting the information about **Main Service SERV5** we press the button Insert New Record () and then:

- We place the cursor in the field Service Code and we enter the code with the logic that described in the previous section, meaning SERV5. After entering this code the system automatically converts the characters we use into Upper Case Characters.
- We place the cursor in the field Service Name and we enter the value “E-Compliance System”.
- Then we place the cursor in the field Process Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “B2: Business Process for Compliance Monitoring”.
- We place the cursor in the field Service Type, we click to display values and we choose the value “Main”.
- Finally we press the Save Button ().

Then we inserting the information **about the Sub Service SERV5.1**. Thus we press the button Insert New Record () and then:

- We place the cursor in the field Service Code and we enter the code with the logic

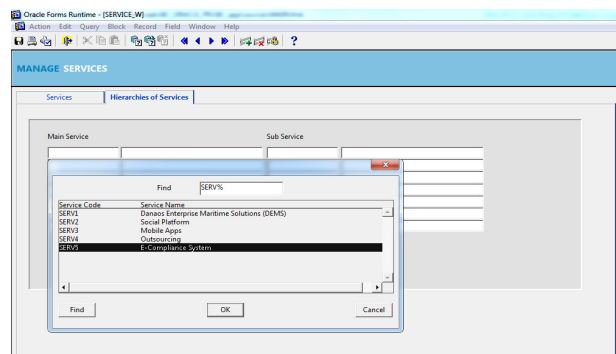


that described in the previous section, meaning SERV5.1. After entering this code the system automatically converts the characters we use into Upper Case Characters.

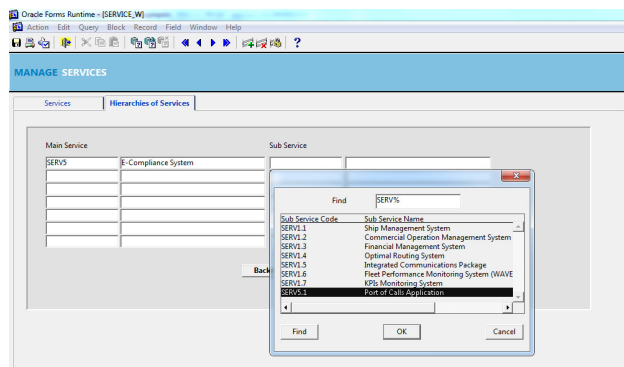
- We place the cursor in the field Service Name and we enter the value “Port of Calls Application”.
- Then we place the cursor in the field Process Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “B2: Business Process for Compliance Monitoring”.
- We place the cursor in the field Service Type, we click to display values and we choose the value “Sub”.

When we choose the value Sub in the field Service Type, then the Button **Hierarchies of Services** appears. So we press this button, we go to the Tab Page “Hierarchies of Services” and then:

- We place the cursor in the field Main Service, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5: E-Compliance System”. With this action the Main Service Code and the Main Service Description appears.



- We place we place the cursor in the field Sub Service, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1: Port of Calls Application”. With this action the Sub Service Code and the Sub Service Description appears.

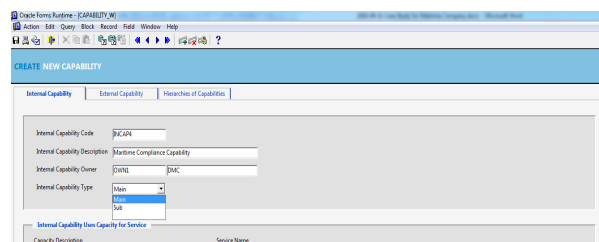
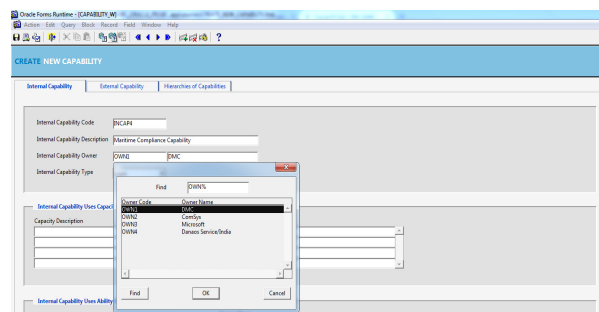


- Finally we press the Save Button (💾) and then the Exit Form Button (🚪).

We now continue with the next step. The Internal Capability is a Main Capability thus we are choosing from the Main Menu the Form “Create New Capability” and we are going to

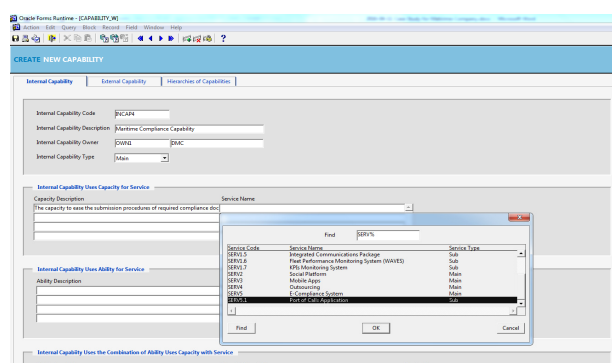
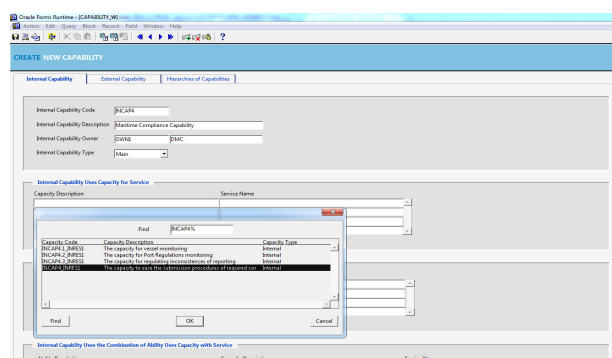
the Tab Page “Internal Capability” to enter the information. In this Tab Page we press the button Insert New Record (📄+) and then:

- We place the cursor in the field Internal Capability Code and we enter the code with the logic that described in the previous section, meaning INCAP4. After entering this code the system automatically converts the characters we use into Upper Case Characters.
- We place the cursor in the field Internal Capability Description and we enter the value “Maritime Compliance Capability”.
- Then we place the cursor in the field Internal Capability Owner, we click on the List of Values, we scroll down the list of values that appears and we select the value “OWN1: DMC”.
- We place the cursor in the field Internal Capability Type, we click to display values and we choose the value “Main”.



The Internal Capability “INCAP4: Maritime Compliance Capability” uses the Internal Capacity “INCAP4_INRES1: The Capacity to ease the submission procedures of required compliance documents for the Port of Calls Application” for delivering the Sub Service “SERV5.1: Port of Calls Application”. Thus:

- We Go to the block Internal Capability Uses Capacity For Service, we place the cursor in the field Capacity Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP1_INRES1”.



- In the same block we place the cursor in the field Service Name, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1”.


The Internal Capability “INCAP4: Maritime Compliance Capability” uses the Internal Ability “INCAP4_INAB1: The Ability to ease the submission procedures of required compliance documents for the Port of Calls Application” for delivering the Sub Service “SERV5.1: Port of Calls Application”. Thus:

- Then we go to the block Internal Capability uses Ability for Service, we place the cursor in the field Ability Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP1_INAB1”.
- In the same block we place the cursor in the field Service Name, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1”.

The Internal Capability “INCAP4: Maritime Compliance Capability” uses the combination of Internal Ability “INCAP4_INAB1: The Ability to ease the submission procedures of required compliance documents for the Port of Calls Application” uses Internal Capacity “INCAP4_INRES1: The Capacity to ease the submission procedures of required compliance documents for the Port of Calls Application” with the Sub Service “SERV5.1: Port of Calls Application”. Thus:

- We go to the block Internal Capability Uses the Combination of Ability Uses Capacity with Service, we place the cursor in the field Ability Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP1_INAB1”.
- In the same block we place the cursor in the field Capacity Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4_INRES1”.
- In the same block we place the cursor in the field Service Name, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1”.
- Finally we press the Save Button (💾).

However the INCAP4 Internal Capability is decomposed into Sub Capabilities. Those Capabilities are the “INCAP4.1: Vessel Monitoring Capability”, the “INCAP4.2: Port Regulation Monitoring Capability” and the “INCAP4.3 Regulation Inconsistence Reporting Capability”. So we have to insert each of them separately.

However we will only describe how we insert the first of them. Thus for the first Sub Capability we are going to the Tab Page “Internal Capability”, press the button Insert New Record () and then:

- We place the cursor in the field Internal Capability Code and we enter the code with the logic that described in the previous section, meaning INCAP4.1. After entering this code the system automatically converts the characters we use into Upper Case Characters.
- We place the cursor in the field Internal Capability Description and we enter the value “Vessel Monitoring Capability”.
- Then we place the cursor in the field Internal Capability Owner, we click on the List of Values, we scroll down the list of values that appears and we select the value “OWN1: DMC”.
- We place the cursor in the field Internal Capability Type, we click to display values and we choose the value “Sub”.

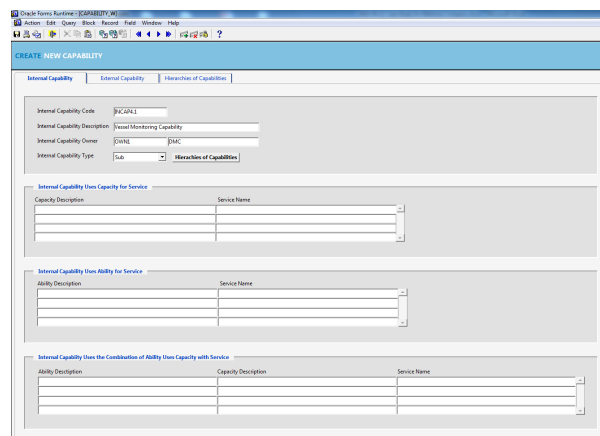
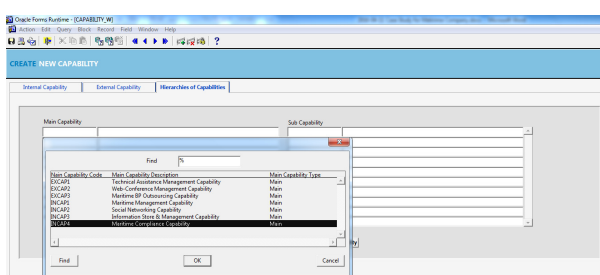
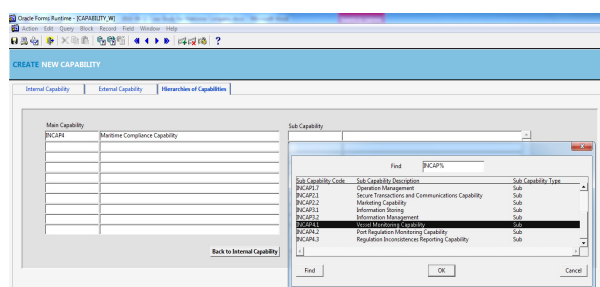
When we choose the value Sub in the field Internal Capability Type, then the button

Hierarchies of Capabilities

appears. So

we press this button, we go to the Tab Page “Hierarchies of Capabilities” and then:

- We place the cursor in the field Main Capability, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”. With this action the Main Capability Code and the Main Capability Description appears.
- We place we place the cursor in the field Sub Capability, we click on the List of Values, we scroll down the list of values

that appears and we select the value “INCAP4.1: Vessel Monitoring Capability”. With this action the Sub Capability Code and the Sub Capability Description appears.

- Then we press the button [Back to Internal Capability](#) and return to the Tab Page “Internal Capability” in order to continue filling in the fields. Worth mentioning that when we press this button then a commit action implement for the inserted records in this Tab Page.

The Internal Sub Capability “INCAP4.1: Vessel Monitoring Capability” uses the Internal Capacity “INCAP4.1_INRES1: The Capacity for Vessel Monitoring” for delivering the Sub Service “SERV5.1: Port of Calls Application”. Thus:

- We Go to the block Internal Capability Uses Capacity For Service, we place the cursor in the field Capacity Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4.1_INRES1”.
- In the same block we place the cursor cursor in the field Service Name, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1”.

The Internal Sub Capability “INCAP4.1: Vessel Monitoring Capability” uses the Internal Ability “INCAP4_INAB1: The ability for Port Regulations monitoring” for delivering the Sub Service “SERV5.1: Port of Calls Application”. Thus:


- Then we go to the block Internal Capability uses Ability for Service, we place the cursor in the field Ability Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4.1_INAB1”.
- In the same block we place the cursor in the field Service Name, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1”.

The Internal Sub Capability “INCAP4.1: Vessel Monitoring Capability” uses the combination of Internal Ability “INCAP4_INAB1: The ability for Port Regulations monitoring” uses Internal Capacity “INCAP4.1_INRES1: The Capacity for Vessel Monitoring” with the Sub Service “SERV5.1: Port of Calls Application”.Thus:

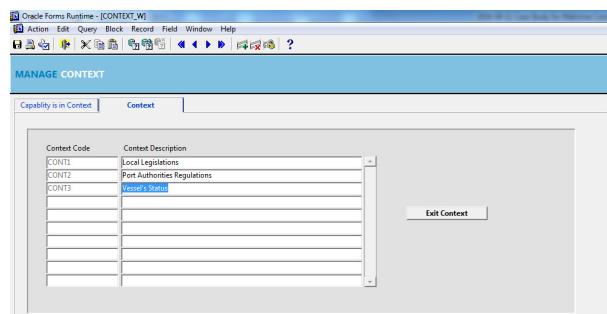
- We go to the block Internal Capability Uses the Combination of Ability Uses Capacity with Service, we place the cursor in the field Ability Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4.1_INAB1”.

- In the same block we place the cursor in the field Capacity Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4.1_INRES1”.
- In the same block we place the cursor in the field Service Name, we click on the List of Values, we scroll down the list of values that appears and we select the value “SERV5.1”.
- Finally we press the Save Button (💾) and then the Exit Form Button (🚪).

NOTE: If this New Capability was an External Capability, then we would choose the Tab Page “External Capability” and fill in the same fields with the same way we have done it in “Internal Capability” Tab Page. Also we would miss the previous steps described, except from the Step that concerns the existence of a New Owner.

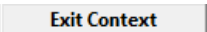
We now continue with the next Step. The INCAP4 is in a new Context and more especially in three different Contexts. Those are “Local Legislations”, “Port Authorities Regulations” and “Vessel’s status”. Since those contexts are new, we are choosing from the Main Menu the Form “Manage Context”, we are going to the Tab Page “Context”, we press the button Insert New Record (📄) (or in the Tab Page “Capability is in Context we press the button ”) and then:

- We place the cursor in the first record to the field Context Description and we enter the value “Local Legislations”.
- We place the cursor in the second record to the field Context Description and we enter the value “Port Authorities Regulations”.
- We place the cursor in the third record to the field Context Description and we enter the value “Vessel’s Status”.

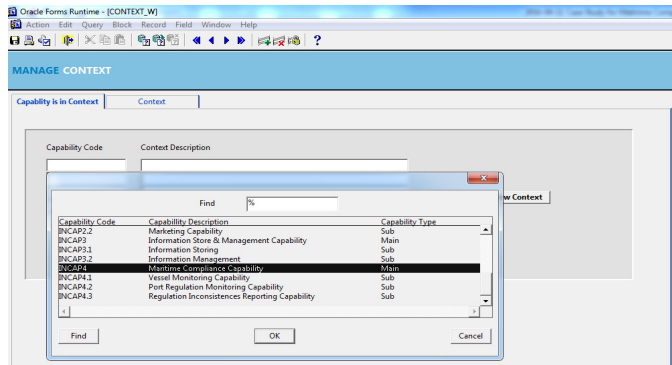


Context Code	Context Description
CONT1	Local Legislations
CONT2	Port Authorities Regulations
CONT3	Vessel's Status

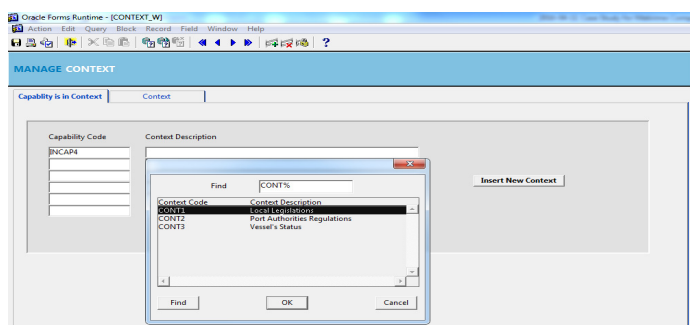
Worth Mentioning that in this page the system automatically creates the Context Code in the logic that discussed in the previous section (meaning CONT1, CONT2, CONTn). Thus:

- We then press the Save Button (💾) and then we push the button  in order to associate this records of context with the INCAP4.


- In the Tab Page “Capability is in Context” that appears, we place the cursor in the first record to the field Capability Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”. With this action the Main Capability Code appears.



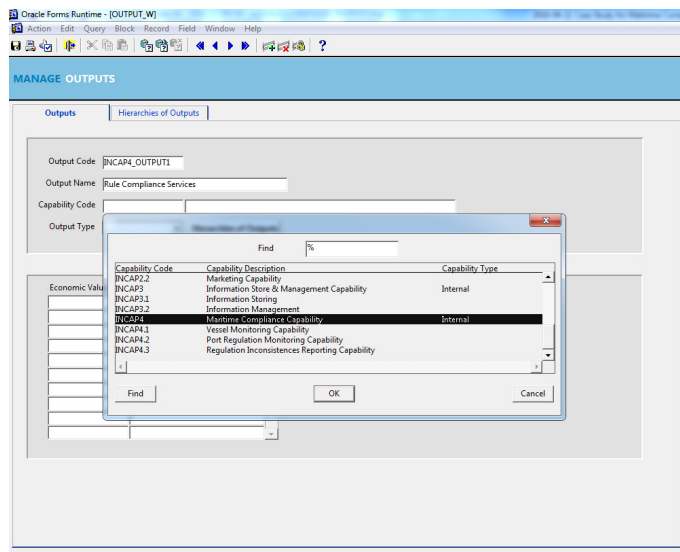
- Then we place the cursor in the field Context Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “CONT1: Local Legislations”. With this action the Context Description appears.



- We then place the cursor in the second record to the field Capability Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”.
- Then we place the cursor in the field Context Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “CONT2: Port Authorities Regulations”.
- Finally we then place the cursor in the third record to the field Capability Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”.
- Then we place the cursor in the field Context Description, we click on the List of Values, we scroll down the list of values that appears and we select the value “CONT3: Vessel’s Status”.
- Finally we press the Save Button (💾) and then the Exit Form Button (🚪).

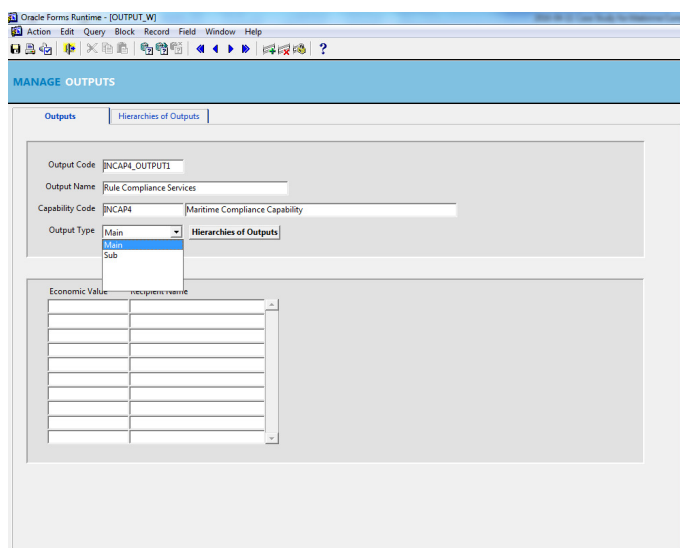
We now continue with the next Step. The INCAP4 delivers the Main Output “INCAP4_OUTPUT1: Rule Compliance Services”. Thus we are choosing from the Main Menu in the Form “Manage Outputs” and choosing the Tab Page “Output” to enter the information. In this Tab Page we press the button Insert New Record () and then:

- We place the cursor in the field Output Code and we enter the code with the logic that described in the previous section, meaning INCAP4_OUTPUT1. During the entering of this code the system automatically converts the characters we use into Upper Case Characters.
- We place the cursor in the field Output Name and we enter the value “Rule Compliance Services”.
- We place the cursor in the field Capability Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”.
- We place the cursor in the field Output Type, we click to display values and we choose the value “Main”.
- We place the cursor in the field Economic Value and enter the information “15.000” (for this field the system show final values in a Format Mask 99,999.00).



The screenshot shows the Oracle Forms Runtime window for the [OUTPUT_W] form. The title bar indicates 'Oracle Forms Runtime - [OUTPUT_W]'. The menu bar includes Action, Edit, Query, Block, Record, Field, Window, and Help. The toolbar contains various icons for navigation and actions. The main window is titled 'MANAGE OUTPUTS' and has two tabs: 'Outputs' and 'Hierarchies of Outputs'. The 'Outputs' tab is active, showing a form with the following fields: Output Code (INCAP4_OUTPUT1), Output Name (Rule Compliance Services), Capability Code (INCAP4), and Output Type (Main). A list of values for Capability Code is displayed, showing INCAP4: Maritime Compliance Capability selected. The list includes the following items:

Capability Code	Capability Description	Capability Type
INCAP2.2	Marketing Capability	Internal
INCAP3	Information Store & Management Capability	Internal
INCAP3.1	Information Storing	Internal
INCAP3.2	Information Management	Internal
INCAP4	Maritime Compliance Capability	Internal
INCAP4.1	Vessel Monitoring Capability	Internal
INCAP4.2	Port Regulation Monitoring Capability	Internal
INCAP4.3	Regulation Inconsistencies Reporting Capability	Internal



The screenshot shows the Oracle Forms Runtime window for the [OUTPUT_W] form. The title bar indicates 'Oracle Forms Runtime - [OUTPUT_W]'. The menu bar includes Action, Edit, Query, Block, Record, Field, Window, and Help. The toolbar contains various icons for navigation and actions. The main window is titled 'MANAGE OUTPUTS' and has two tabs: 'Outputs' and 'Hierarchies of Outputs'. The 'Outputs' tab is active, showing a form with the following fields: Output Code (INCAP4_OUTPUT1), Output Name (Rule Compliance Services), Capability Code (INCAP4), and Output Type (Main). The Economic Value field is empty. The list of values for Capability Code is displayed, showing INCAP4: Maritime Compliance Capability selected.

- We place the cursor in the field Recipient Name and enter the Value “Bulk Carriers S.A”
- Finally we press the Save Button (💾).

However the INCAP4_OUTPUT1 is decomposed into Sub Outputs. Those Outputs are the “INCAP4_OUTPUT1.1: Vessel Monitoring Services”, the “INCAP4_OUTPUT1.2: Port Regulation Monitoring Services” and the “INCAP4_OUTPUT1.3: Regulation Inconsistences Reporting Services”. So we have to insert each of them separately. However we will only describe how we insert the first of them. Thus for the first Sub Output we are going to the Tab Page “Output”, press the button Insert New Record (➕) and then:

- We place the cursor in the field Output Name and we enter the value “Vessel Monitoring Services”.
- We now place the cursor in the field Capability Code, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”.
- We place the cursor in the field Output Type, we click to display values and we choose the value “Sub”.

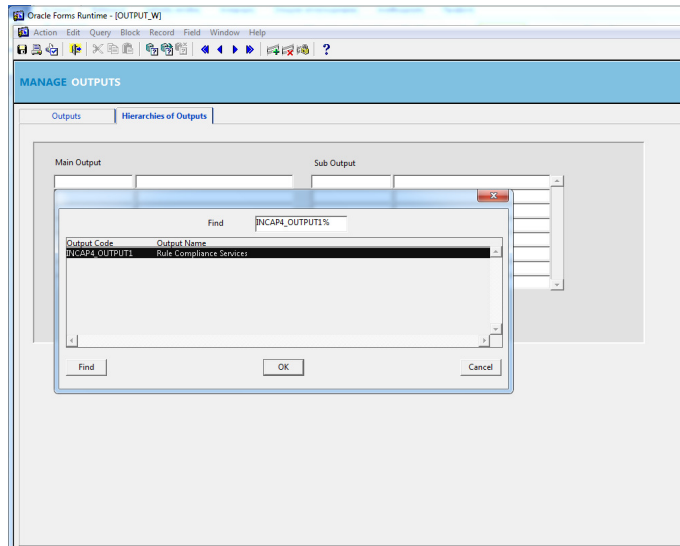
When we choose the value Sub in the field Output Type, then the button **Hierarchies of Outputs** appears.

So we press this button, we go to the Tab Page “Hierarchies of Outputs” and then:

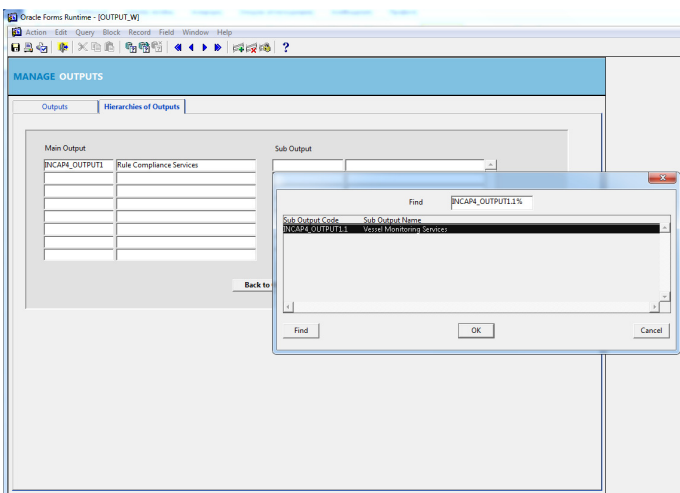
- We place the cursor in the field Main Output,

we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4_OUTPUT1”.

With this action the Main Output Code and the Main Output Description appears.



- We place we place the cursor in the field Sub Output, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4_OUTPUT1.1”. With this action the Sub Output Code and the Sub Output Description appears.




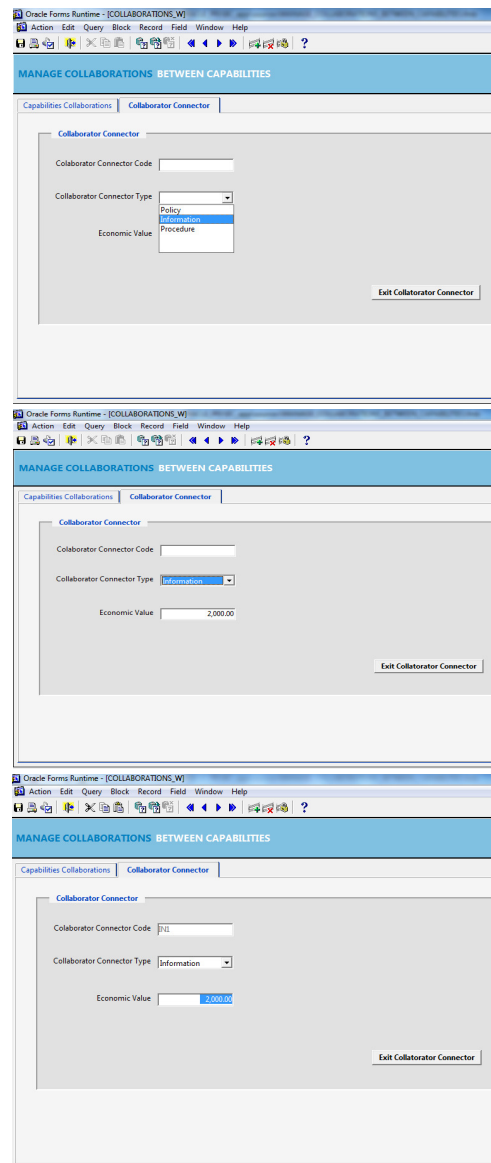
- Then we press the button **Back to Outputs**. When we press this button, then a commit statement occurs in the new record.
- Then we place the cursor in the field Economic Value and enter the information “5.000” (for this field the system show final values in a Format Mask 99,999.00).
- We place the cursor in the field Recipient Name and enter the Value “Bulk Carriers S.A”
- Finally we press the Save Button (💾) and then the Exit Form Button (🚪).


We now continue with the next step. The ICAP4.1: Vessel Monitoring Capability collaborates with INCAP1.5: Human Resource Management Capability through the Collaborator Connector “Information”, which costs 2.000. Since we have a new Collaborator Connector we are choosing from the Main Menu the Form “Manage Collaborations Between Capabilities”, we are going to the Tab Page “Collaborator Connector”, we press the button

Insert New Record () (or in the Tab Page “Capabilities Collaborations” we press the button

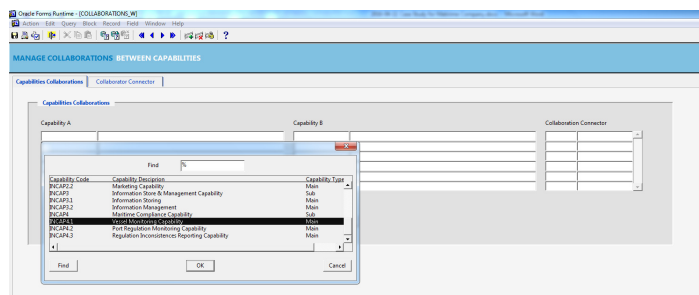
Insert New Collaborator Connector) and then:

- We place the cursor in the field Collaborator Connect Type, we click to display values and finally we choose the correct value, meaning the value “Information”.
- We place the cursor in the field Economic Value and enter the information “2.000” (for this field the system show final values in a Format Mask 99,999.00).
- Then we press the Save button (). When we are pressing this button we see that the system automatically creates the Collaborator Connector Code in the logic that discussed in the previous section (meaning IN1, etc.)
- Then we press the button **Exit Collatorator Connector**, in order to go to Tab Page “Capabilities Collaboration”.

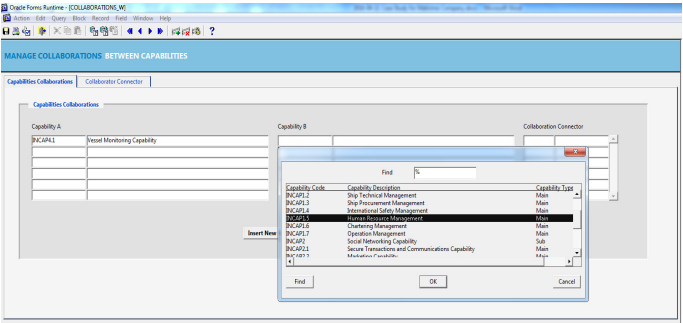


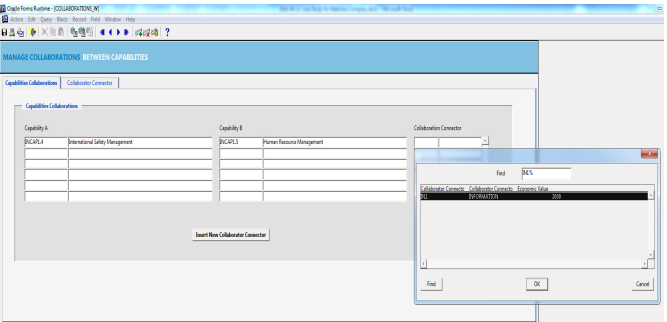
In the Tab Page “Capabilities Collaborations” we press the button Insert New Record () and then:

- We place the cursor in the first record on the field Capability A, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4.1: Vessel Monitoring Capability”. With this action the Capability Code and the Capability Description appears.



- We place the cursor in the first record on the field Capability B, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP1.5: Human Resource Management Capability”. With this action the Capability Code and the Capability Description appears.

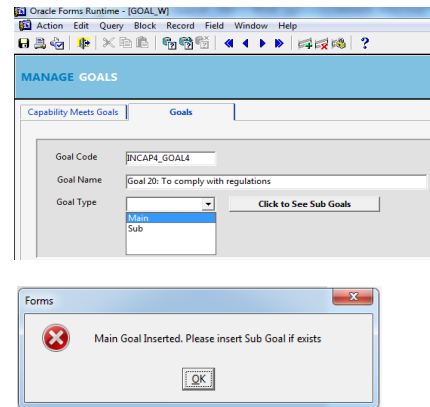

- We place the cursor in the field Collaborator Connector, we click on the List of Values, we scroll down the list of values that appears and we select the value “IN1”. With this action the Collaborator Connector Code and the Collaborator Connector Type appears.


- Finally we press the Save Button (💾) and then the Exit Form Button (🚪).

We now continue with the next steps. The INCAP4 Capability meets four new Main Business Goals, which are “INCAP4_GOAL1= Goal 9: To participate in research projects”, “INCAP4_GOAL2= Goal 10: To collaborate with academic research”, “INCAP4_GOAL3= Goal 13: To identify client’s needs” and “INCAP4_GOAL4= Goal 20: To comply with regulations”. For the previous Goals the last one is decomposed to three other Sub Goals, which are “INCAP4_GOAL4.1= Goal 41: To monitor vessel status”, “INCAP4_GOAL4.2= Goal 42: To be informed about the regulation of each port” and “INCAP4_GOAL4.3= Goal 43: To get alert when regulation are not met”. Since these Goals are new, we are choosing from the Main Menu the Form “Manage Goals” and we are going to the Tab Page “Goals”. Since the procedure of inserting this main goals is the same we will only describe the inserting of INCAP4_GOAL4 and its Sub Goals. Thus in this Tab Page we press the button Insert New Record (➕) and we fist insert the Information about the Main Goals as follows:

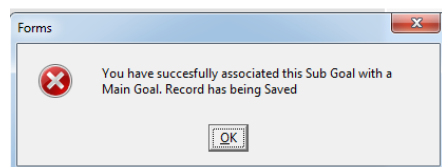
- We place the cursor in the field Goal Code and we enter the code with the logic that described in the previous section, meaning INCAP4_GOAL4. During the entering of this code the system automatically converts the characters we use into Upper Case Characters.
- We place the cursor in the field Goal Name and we enter the value “Goal 20: To comply with regulations”.
- We now place the cursor in the field Goal Type, we click on the List of Values and we select the value “Main”.

When selecting the value “Main” the following message appears, which says that a commit action took place in the database for the inserted records. Thus we press ok.

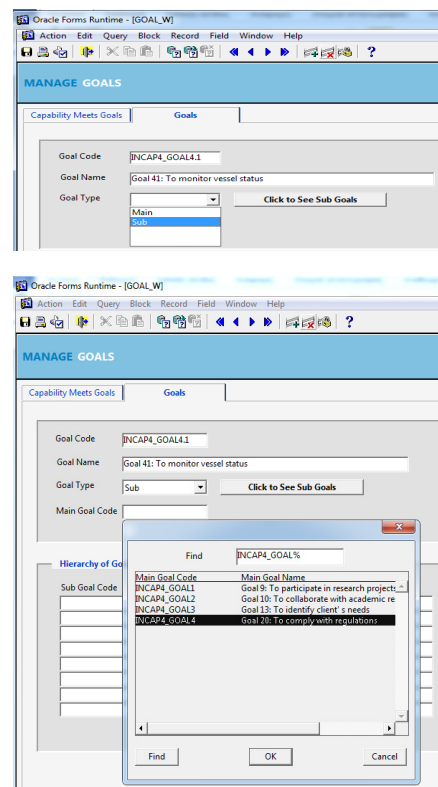


Then we insert the information about Sub Goals that related with the Main Goal INCAP4_GOAL4. Since the procedure of inserting every Sub Goal is the same we will only describe the first one, meaning the INCAP4_GOAL4.1. Thus:

- We place the cursor in the field Goal Code and enter the code with the logic that described in the previous section, meaning INCAP4_GOAL4.1.
- Then we place the cursor in the field Goal Name and we enter the value “Goal 41: To monitor vessel status”.
- We now place the cursor in the field Goal Type, we click on the List of Values and we select the value “Sub”.
- After selecting the value “Sub” a Main Goal Code field appears and a List of Values in that field. We scroll down the list of values and select the value “INCAP4_GOAL”.
- Then the following message appear:

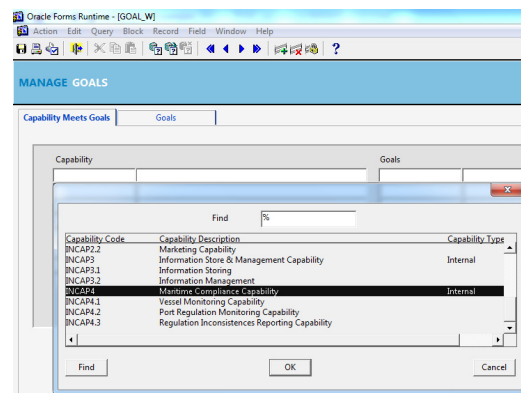


This message says that the record has being saved in the database.

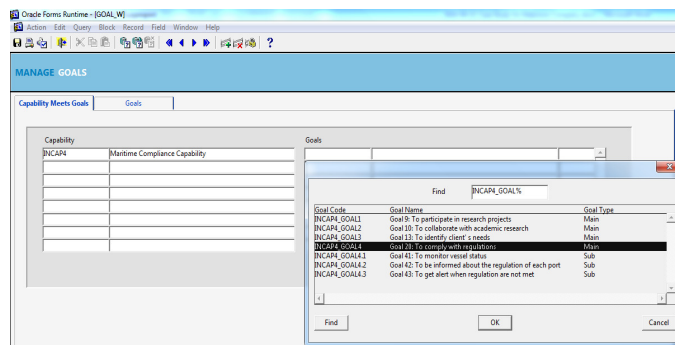


Then we go to the Tab Page “Capability Meets Goal” in order to associate the INCAP4 capability with the Goals. Thus we click on the Tap Page “Capability Meets Goal” and then:

- We place the cursor in the first record on the field Capability, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4: Maritime Compliance Capability”. With this action the Capability Code and the Capability Description appears.



- We place we place the cursor in the first record on the field Goals, we click on the List of Values, we scroll down the list of values that appears and we select the value “INCAP4_GOAL4”. With this action the Goal Code and the Goal Name appears.




The same procedure exists for the association with other goal, however we will not descibed it, in order to avoid redundancy.



- Thus finally we press the Save Button (💾) and then the Exit Form Button (🚪)

8.3 Executing Queries

In the previous section we describe by using a real data the procedure of inserting a new Business Capability. In this section we will describe how a user can perform queries in specific cases.

The actions of managing queries are available in every Form of the Application, through the Main Item and the Tool Bar, that are visible in the top of them. We have already described each of them in the previous section. For executing queries in the Total of Forms we simply insert a value of interest or a combination of values, in the associated field or fields and then we press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute). However except from the using of the previous case, in some Forms we have the ability to query and see specific information about an inserting value in different way.

More specific if we assume that we want to see all the information about a specific Main Output e.g INCAP4_OUTPUT1: Rule Compliance Services, and also which Sub Outputs are associated with this Main Output, then we are doing the following:

- We are going from the Main Menu in the Form “Manage Outputs” and we click to open it.
- In the Page that appears we are going to the Tab Page ‘Outputs’, and from the Tool Bar we press the button Enter Query  (or from the Menu Item of this Page we choose Query → Enter).
- Then we are entering in the field Output Code the value “INCAP4_OUTPUT1” and from the Tool Bar we press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute).

When we are pressing this button then the system automatically bring all the information about this Output as follows:

Oracle Forms Runtime - [OUTPUT_W]

MANAGE OUTPUTS

Outputs | Hierarchies of Outputs

Output Code: INCAP4_OUTPUT1


Output Name: Rule Compliance Services

Capability Code: INCAP4

Output Type: Main

Economic Value	Recipient Name
15,000.00	Bulk Carrier S.A.

Now if we want to see which are the Sub Outputs of this Main Output we leave the results of this query as it is and then:

- We click on the Tab Page “Hierarchies of Outputs”. After entering in this Tab Page, the cursor is already in the first field, meaning the “Main Output”.
- Then from the Tool Bar we press the button Execute Query .

With this last action the system brings automatically the information about the hierarchy of this specific Output as follows:

Oracle Forms Runtime - [OUTPUT_W]


MANAGE OUTPUTS

Outputs | Hierarchies of Outputs

Main Output	Sub Output
INCAP4_OUTPUT1	Rule Compliance Services
INCAP4_OUTPUT1.1	Rule Compliance Services
INCAP4_OUTPUT1.2	Rule Compliance Services
INCAP4_OUTPUT1.3	Rule Compliance Services

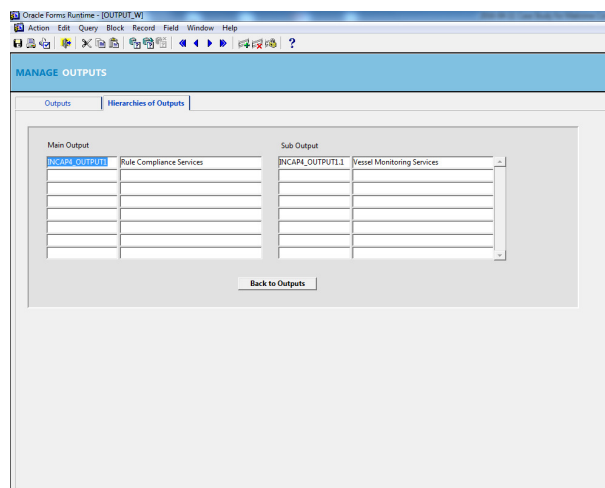
Back to Outputs

Now if we reverse the query, meaning that we want to see all the information about a specific Sub Output e.g INCAP4_OUTPUT1.1: Vessel Monitoring Services, and also in which Main Output it refers, then we are doing the following:


- We are going from the Main Menu in the Form “Manage Outputs” and we click to open it.
- In the Page that appears we are going to the Tab Page ‘Outputs’. Here let’s assume that we want to bring all the records about Outputs. Thus we press the button Execute Query  (or from the Menu Items of this Page we choose Query → Execute) to bring all the records.
- Now the cursor is already on the field “Output Code”, thus we navigate with the help of keyboard Up and Down Arrows, until we find the value “INCAP4_OUTPUT1.1”. When we choose this value all the information about it appears.

Now since this Output is a Sub Output we see that in this Page the button “Hierarchy of Outputs” appears. Thus:

- If pressing this button, then the system goes us to the Tap Page “Hierarchies of Outputs” and saw us automatically in which Main Output this Output refers to as follows:





Finally if we want to see all the information about Hierarchies of Outputs, meaning all the Main Outputs and the Sub Outputs that contains:

- We are going from the Main Menu in the Form “Manage Outputs” and we click to open it.
- We click in the Tab Page “Hierarchies of Outputs”.
- Then we press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute) to bring all the records.

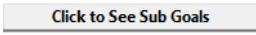
The results are as follows:

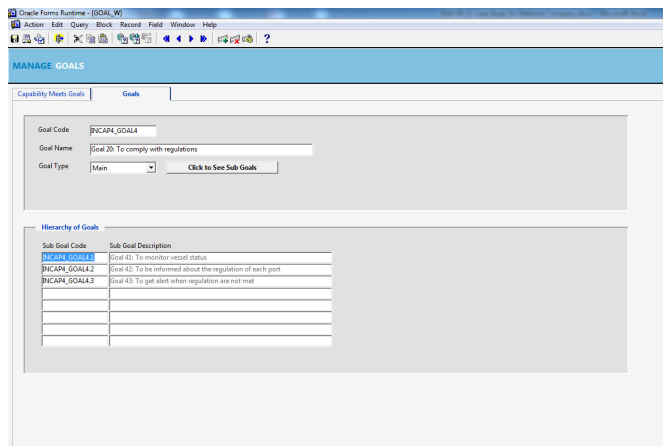
NOTE: Similar queries as the previous described can be executed in the Forms “Create New Capability” and “Manage Services”.

Another case is when we want to see all the information about a Main Goal e.g. INCAP4_GOAL4 and also the Sub Goals that is decomposed to. For seeing this kind of information we are doing the following:

- We are going from the Main Menu in the Form “Manage Goals” and we click to open it.
- We click in the Tab Page “Goals”.
- Then we press the button Enter Query  (or from the Main Menu of this Page we choose Query → Enter) and we enter in the field Goal Code the value “INCAP4_GOAL4”.
- We then press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute).



When we are pressing this button then the system automatically bring all the information about this Goal as follows:

- In this state of the system if you press the button , then the Sub Goals appears as follows:

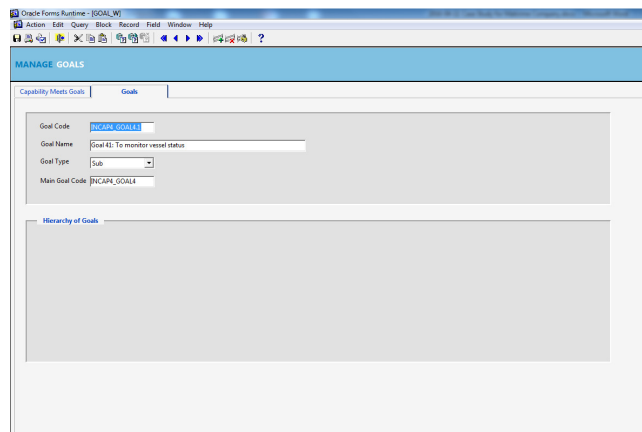


Sub Goal Code	Sub Goal Description
INCAP4_GOAL4.1	Goal 41: To monitor vessel status
INCAP4_GOAL4.2	Goal 42: To be informed about the regulation of each port
INCAP4_GOAL4.3	Goal 43: To get alert when regulations are not met

Now if we reverse our query, meaning to search all the available information about a Sub Goal e.g. INCAP4_GOAL4.1 and also in which Main Goal it refers, we are doing the following:



- We are going from the Main Menu in the Form “Manage Goals” and we click to open it.
- We click in the Tab Page “Goals”.
- Then we press the button Enter Query  (or from the Main Menu of this Page we choose Query → Enter) and we enter in the field Goal Code the value “INCAP4_GOAL4.1”.
- We then press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute).

When we are pressing this button then the system automatically bring all the information about the Sub Goal and also a “Main Goal” field appears, which says in which Main Goal this Sub Goal is part of. This follows thereafter:

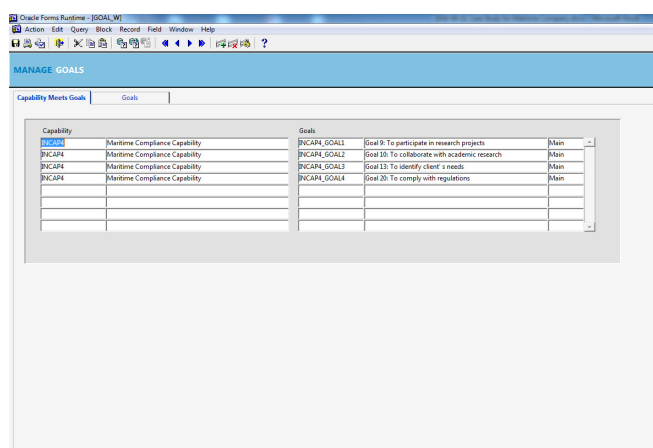


Sub Goal Code	Sub Goal Description

Since we are in this Form let's execute a query to see whose Goals meet the INCAP4 Business Capability. For executing this kind of query we are doing the following.

- In the same Form we are clicking on the Tab Page "Capability Meets Goals".
- Then we press the button Enter Query  (or from the Main Menu of this Page we choose Query → Enter) and we enter in the field Capability the value "INCAP4".
- We then press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute).

The last action causes the following results:

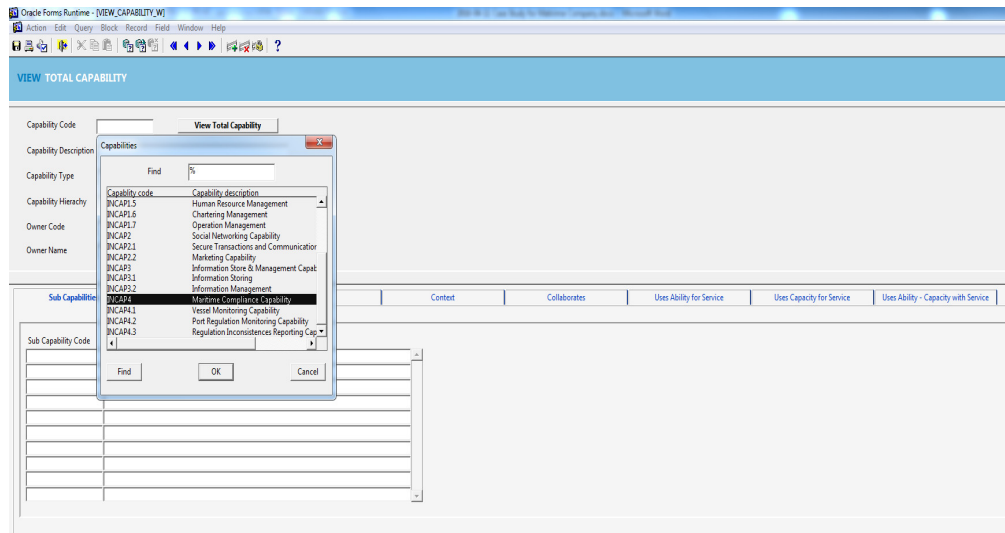


Capability	Goal
INCAP4	Goal 9: To participate in research projects
INCAP4	Goal 10: To collaborate with academic research
INCAP4	Goal 13: To identify client's needs
INCAP4	Goal 20: To comply with regulations

NOTE: Similar queries as the last one described, can be executed in the first Tab Pages of the Forms "Manage Context" and "Manage Collaborations between Capabilities". Also in more general it can be executed in every available field included in the Forms of our Application.

Finally our Application contains a Form which has been designed only for executing queries about Business Capability. This is the "View Total Capability" Form. Thus if for example we want to see all the Information about the INCAP4 Business Capability, then:


- We are going from the Main Menu in the Form "View Total Capability" and we click to open it.
- Then we place the cursor in the first record on the field Capability, we click on the List of Values, we scroll down the list of values that appears and we select the value "INCAP4: Maritime Compliance Capability". With this action the Capability Code and the Capability Description appears.





- We then press the button **View Total Capability**.

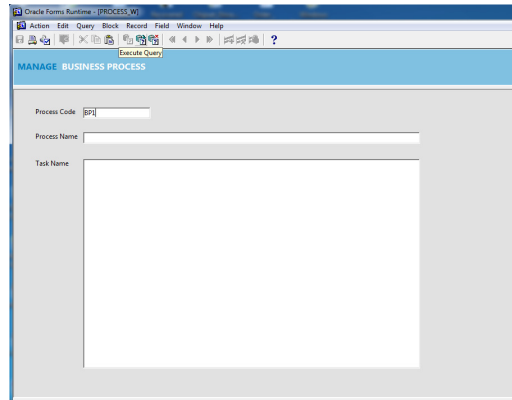
When we are pressing this button then the system automatically bring all the available information about INCAP4.

8.4 Removing Current Records in Specific Forms


The action of removing current records is available in every Form of the Application through the Main Menu that is visible in the top of them. More specific if a user wants to remove a specific record in a Form, he simply select this record with the cursor and then either presses the button , or from the Menu Items he choose Record → Remove. In every form a specific message appears before this action take place.

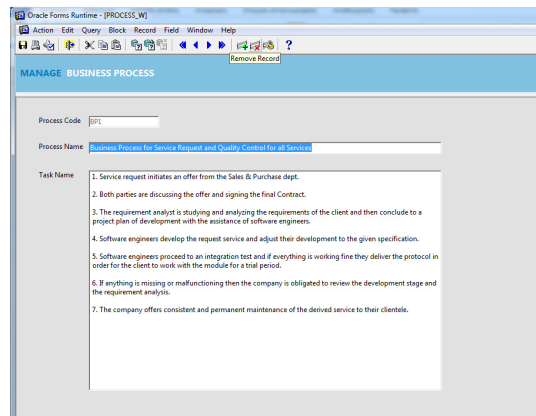
For example let's assume that we want to remove a specific Business Process e.g. BP1. Thus:

- We are going from the Main Menu in the Form “Manage Business Process” and we click to open it.
- Then we press the button Enter Query  (or from the Main Menu of this Page we choose Query → Enter) and we enter in the field Process Code the value “BP1”.
- We then press the button Execute Query  (or from the Main Menu of this Page we choose Query → Execute).

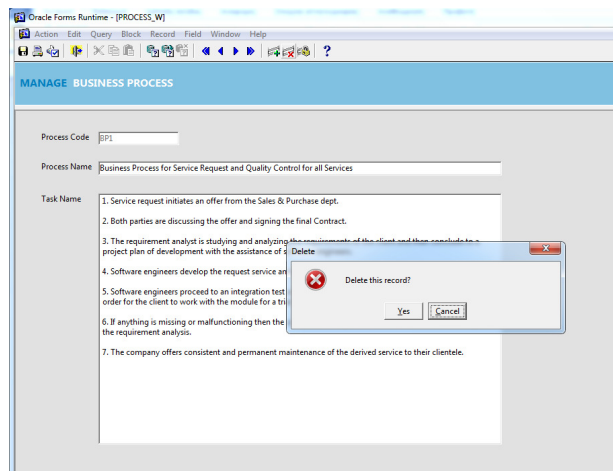


When we are pressing this button then the system automatically bring all the information about this Business Process.

- We leave the cursor in the field “Process Name” and then we press the button .



When we are pressing this button a message appears as follows:




- We then choose the “Yes” button from the two options.

NOTE: This is the general process of removing a record in every Form of our Application.

However in our Application in some cases when a user is removing a specific record then the system automatically removes some others records that associated with it. This actually refers in the cases of removing a record that refers to a top element in a hierarchy.

More specific if we want to remove a Main Business Capability then the system removes not only the Information about this Business Capability, but also all the information about its Sub Capabilities. Thus let's assume that we want to remove the INCAP4 Business Capability. Firstly we will execute a query to see which his Sub Capabilities are. Thus following the procedure that was described in the previous section, we will see that his Sub Capabilities are:

Thus in this Page we press the button **Back to Internal Capability** and:

- We leave the cursor in the field “Internal Capability Code” in the current record and then we press the button .

When we are pressing this button a message appears as follows:

- In this message we press the Yes button.

Now if you execute a query to see all the records in the Tab Page Internal Capability, we will see that not only the record for INCAP4 has being deleted, but also the records for INCAP4.1, INCAP4.2 and INCAP4.3. Also if we go to the Tab Page “Hierarchies of Goals” and execute a query, none of these records exists.

NOTE: The same process as previous exists in the Forms “Manage Outputs”, “Manage Goals” and “Manage Services”.

By describing the way we remove current records in specific Forms we fulfill the chapter of this dissertation. Next Chapter deals with the conclusions from the total work of this dissertation.

8.5 Chapter Summary

In this Chapter we have presented how a user may interact with the maritime application in different cases. Thus in order the previous scope to be achieved, we have taken a specific case study from the maritime domain field, the Danaos Management Consultants (DMC), who is a software and services company specializing in maritime IT.

For a start we have presented a sample of data description from the previous case study, which was focusing in a specific Business Capability that this Company has, the INCAP4: Maritime Compliance Capability. Those data was given in a form of tables and in the top of them is described each value separate in relation with the values types names of the physical database tables. Also a specific codification for this data is given. Finally a description for what they refer to.

Then we have presented the Main Menu that exists in the top of every page and contains specific Menu Items and a Tool Bar. In more specific for this Menu we have presented all the possible actions that can be executed by the user, when using it.

Thereafter we have taking into account the previous data, in order to show how a user may interact with the application in a case of informing the application about the information related with a new Business Capability. In this procedure we have guided by the Basic Flow Chart that was discussed in the previous Chapter, and we are also having described the procedure followed in order a user to login the system. Secondly we have presented the procedure followed in order a user to execute queries according to specific criteria and in this part a screen for Business Capability that has being designed for that purpose is presented. Finally we have presented how a user may remove specific record in specific forms of the application. This has to do with specific actions that the system automatically executes when removing the data that related with a case of hierarchies. Thus as we can see interacting with the application is an important part during the development process of the maritime application. That's because by this way we can test in depth the systems functionality, and also we can if the system works properly or not.

CHAPTER 9: Conclusions

This chapter deals with the conclusions of this dissertation. In more detail a brief summary of the total work is given, including some observations, the result of work and some directions for future work.

When searching in bibliography we can find numerous publications and researches about Capabilities. From the previous we observe that during the time Capabilities played an important role in Management Science, in order an organization to gain a competitive advance. Also the evolution of those into global strategies and thereafter into a notion that describes the whole elements of an organization, meaning the Business Capability notion, reinforces their importance.

Business Capabilities describes what an organization does, and this meaning has turned the researches in bibliography from the Management Theories, into the Business Informatics, and finally into the searching of using of them as a centric idea for the development of the digital enterprises of tomorrow. That's because when an organization focus in Business Capabilities is able to gain a competitive advance and thereafter to achieve growth.

Different approaches for modeling Business Capability can found in bibliography (Holman, 2006; Brits, Botha, & Herselman, 2007; Freitag, Matthes, Schu, & Nowobilska, 2011; Ulrich & Rosen, 2011), with the most complete this of (Ulrich & Rosen, 2011), since they provide a clear decomposition of Capability map hierarchy and a clear picture of the role of Business Capabilities in Enterprise Architecture. By using this approach an organization can achieve the alignment between Business and IT. In the meantime the extensive use of Internet and its variability led to the development of Capability Driven Development (CDD) method (Stirna, Grabis, Henkel, & Zdravkovic, 2012) that integrates organizational with IS development taking into account changes in the application context of a solution and also uses a Meta – model of Capability.

Thus in this dissertation we have introduced a new way for describing Business Capability in an empirical form. In more detail we have created a Database Management System for Business Capability and we have taken a Case Study from the Maritime Domain Field, the Danaos Management Consultant (DMC) in order to express the information about that. By this system all the information about Business Capability can be stored, related and viewed at any time by Managers. Thus Managers have an overall view of what the organization does and thereafter there are able to analyze their systems and processes more efficient. Also they can increase control, achieve better planning, make good decisions and are able to identify any requirements for change that may occur.

For creating this Database Management System we have chosen the Object Role Modeling technique instead of other entity-relationship and object oriented modeling methods. During the description of ORM technique, we observe that although there was no official standard for meta-modeling it uses a meta-modeling standard like this of OMG. In more specific the first level of ORM technique, meaning the Conceptual Schema Design Procedure, the information about the real word (UoD), can be considered as the M0 level in meta-modeling. Then this information, is depicted in a Conceptual Schema (in a drawing, in a conceptual model), which in turn can be considered as the M1 level in meta-modeling. Thereafter the building blocks or the meta-data that can be used to make this model are defined, meaning the abstract syntax of this modeling language. These blocks concerns the object types that can be used to present the model, the relations between the object types, the identifiers of the object types, the meaning of the object types (semantics) and the rules to combine the object types. Thus the previous level can be considered are the M2 level in meta-modeling. Finally the ORM is using a specific graphical notation, meaning the concrete syntax of this modeling language, which can be considered to be the M3 level. The same analysis as previous can be considered for the second level of analysis in ORM, meaning the Relational Schema Procedure, in which a second model is produced, which can be though as a meta-model since is a model that has being produced according to some other model.

Working with this method in the first level we have redesign a Conceptual Model for Business Capability definition. By this way we were able to understand in depth the Universe of Discourse of this project. That's because this data modeling method is descriptive enough, since it uses natural language and specific graphical notation to describe the specifications of the system implemented, which in fact has also helped us enable communication between stakeholders and to implement a system conforming to the information requirements. From the above we can understand that an important factor when designing a Conceptual Schema is the purpose of designing. This purpose usually specifies what kind of information must be depicted and the way is depicted. On the other hand when describing a specific UoD by a Conceptual Schema different patterns may be produced according to the way of thinking of the modeler. For example taking the previous initial of Conceptual Schema for Business Capability was an important help for producing an accurate and complete model for our Application. That's because this model became the basis for the designing of our model, since it has describe correctively the way most of the data required to be stored in the maritime application. On the other hand in the new Business Capability meta-model, some fact types have being depicted in a different way than the initial meta-model. The previous

has to do with the way of thinking of the modeling. However, the most important in both cases is not the way of depicting the different models by the modelers, but the prevention of missing important information about the under description of UoD. That's why we are also agree that in the CSPD procedure the most important part is the first step, where examples of data are express in term of elementary facts. This is a step where if the information about the UoD is not expressed in detail, then there is a big possibility to lead into a missing of data. Thus in this step it is important for the modeler to have a full access and permission by the company to the all available information about it.

Then this first level model became a guide for creating a second level model, the Relational Schema. Here we can say that the procedure of Relational mapping is easily understood and standardized. This means that if we have created an accurate and complete model in the first level of ORM, then the procedure of mapping into a Relational Schema is easily implemented. Otherwise, if from the Conceptual Schema we are missing some of the required information, then the procedure of mapping is flexible enough to express this information in a wright way, by the Normalization method. The results of this procedure were a specific number of tables, including all the required by the specification relationships and constraints.

Also this schema became the guide for creating the physical database schema in the next level. During the creation of the Database Schema we can say that it is important how we have designed the Relational Schema in order to create the physical database of our system. That's because everything is depicted in this schema, must then take the form of an object in the database and sometimes this is not feasible at once. This means we have to implement some extra procedural code like triggers, sequences etc, in order the information to be maintained with the right way. The last one presupposes a good knowledge of SQL language by the developer of this system. However in this stage as we observe some of the textual constraints that refers to unions of specific object (e.g. Business Capability (capcode) = IntenalCapabilty (incapcode) union ExternalCapability (excapcode)), has not being implemented yet.

Then taking into account the physical Database Schema we created the User Interface of the application. During the designing of the interface for the maritime application we can say that a good knowledge of SQL language was required. Also was required the using of UML language in order to describe the main windows, to give a hierarchy of forms and to describe the steps that a user follows in a case of a data entry. By creating the previous three kinds of diagrams we were able to understand how this application must be created,

which was an important factor in order the current user interface to be characterized by a specific quality criteria. Finally except from the previous in this step we were able to implement some of the constraints that were missing from the previous steps of ORM technique, which had to do with the union textual constraints. This means that all the process of creation the Database Management System was flexible enough, since every stage overlaps the other.

Finally we can say that interacting with the application is an important part during the development process of the maritime application. That's because by this way we can test in depth the systems functionality, and also we can if the system works properly or not.

Thus results of this dissertation include a proposal of ORM data modeling method for Business Capability description, in order to support the Capability Driven Development (CDD) approach. That's because his technique follows a complete procedures in the different levels of analysis for a specific UoD, and also it is flexible enough, since every level of analysis overlaps the other.

Also since the CDD approach lacks from empirical experience of application, we have taken the Case Study of Danaos Management Consultant (DMC), in order to show that it is feasible to implement it in a real case example.

Future work on this aspect would include the using of other data modeling methods or languages, for creating a similar Database Management System of Business Capabilities definition. By this way it would be possible to confirm, determine and define the appropriate data modeling method for supporting the CDD approach.

CHAPTER 10: Appendix

10.1 Table 1: Literature Review for Business Capability

10.2 Table 2: ORM 2 Graphical Notation

10.3 SQL Script of BC Tables

10.4 SQL Script of Total View

10.1 Table 1: Literature review for Business Capabilities

Author	Year	Title	Research aim/ objectives	Theoretical perspective/ framework	Method (empirical/ theoretical)	Main findings	Source/ Journal
Ulrich Holman	2006	A Business – Oriented Foundation for Service Orientation	<p>To answer in three main queries about Service Oriented Architectures.</p> <ul style="list-style-type: none"> How do we prevent Service-Oriented Architectures from following the architecture mistakes of the past; How do we ensure that the chosen implementation architecture relates to the actual desired state of business; How do we prolong the life expectancy of the implementation in an ever-changing environment; 	Enterprise Architecture and Service Oriented Architecture (SEA)		<p>1) The introduction of a more stable foundation, focusing in “What” a business actually does (Business Capabilities).</p> <p>2) The introduction of a framework for Business Capability Model implementation.</p>	Microsoft Corporation (https://msdn.microsoft.com/en-us/library/aa479368.aspx)
Denise Cook	2007	Business – Capability Mapping: Staying Ahead of the Joneses	To support the idea that Business – Capability mapping enables adaptive, sleek architectures that can respond quickly to changes in today’s competitive business landscape.	Enterprise Architecture	A Case Study of Phone Company	<p>1) Business – Capability mapping promotes a strong relationship between business mode and the technical infrastructure that supports the business requirements.</p> <p>2) Business – Capability mapping aligning the Technical Architecture to the Business Architecture.</p> <p>3) Business – Capability mapping provides a clear road map to SOA.</p>	Microsoft Corporation (https://msdn.microsoft.com/en-us/library/bb402954.aspx)

J. Brits, G.H.K. Botha, M.E. Herselman & Tshhwane	2007	Conceptual Framework for Modeling Business Capabilities	To provide a conceptual approach to analyze an organization and a foundation that would support the architecture of an agile organization by illustrating Business Capabilities.	Enterprise Architecture	Context analysis and qualitative research combined with a systems approach in the development of a model.	4) The development of a conceptual framework to construct Business Capabilities. 5) A production of a Business Capability model. 6) A production of two feed-back loops (Organizational Feedback Loop and Innovative Feedback Loop).	Informing Science and IT Education Joint Conference
Len Greski	2009	Business Capability Modeling: Theory and Practice	To provide a theory and practice about Business Capability.	Business Strategy & Business Architecture		1) A definition of Business Capability 2) A simple technique for modeling Business Capability. 3) The reasons for using the model of Business Capability, by an organization in order to make decisions.	Architecture & Governance Magazine (Volume 5, Issue 7)
Len Greski	2009	Business Capability Modeling: Building Hierarchy	To provide a framework about Building the Hierarchy and Identifying Key Relationships during the stages of modeling Business Capability and to provide some practical considerations about the previous.	Business Strategy & Business Architecture		1) A framework for Building Hierarchy 2) A framework for Identifying Key Relationships.	Architecture & Governance Magazine (Volume 5, Issue 11)
Wolfgang Keller	2009	Using Capabilities in Enterprise Architecture Management	To explain a few of the basic mechanisms behind capability based modeling in pattern form.	Enterprise Architecture Management & IT Planning		1) A Pattern Roadmap to illustrate capabilities. 2) Capabilities make enterprise architectures more effective and make an organization to have profit.	Whitepaper – Version of 2009, Lochham, Germany.
Jeff Scott	2009	Business	To state Business Capability Models as a new approach to	Business		1) Capability models provide a focal point for	Architecture &


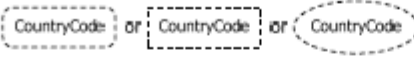

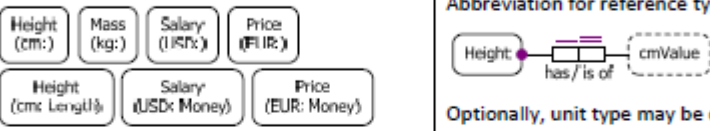

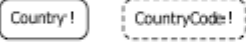


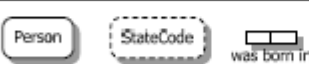


		Capability Maps: The missing Link Between Business Strategy and IT Action	close the gap between business interests and IT concerns, providing the right level of detail and consistency to facilitate an ongoing dialogue between business and IT leaders.	Strategy and Information Technology		<p>strategic dialogue.</p> <p>2) Companies are using capability maps to create value.</p> <p>3) Capability models are the core component of the overall business architecture framework.</p> <p>4) IT architects and planners can take capabilities as the starting point for discussion about IT investments.</p> <p>5) Capability models provide the Rosetta Stone through which business needs aligned IT action</p>	Governance Magazine (Volume 5, Issue 9)
Mike Rosen	2010	Business Processes starts with Capabilities	To introduce a discussion about how do we get to the heart of what business capabilities an enterprise needs? Also how business capabilities currently implemented in the future state of a more flexible, efficient and aligned Business/IT Solutions?	Business Architecture		<p>1) Analysis of the value streams leads to identification of the business capabilities.</p> <p>2) Business processes describe how the business performs, or implements, the given capability and how capabilities connect to deliver a desired outcome.</p> <p>3) Business Capabilities provide the link between two complex, disparate environments. The business and IT Architectures.</p>	A BPT trends column (www.bptrends.com)
Thiago Barroero, Giammario Motta & Giovanni Pignatelli	2010	Business Capabilities Centric Enterprise Architecture	To introduce a well-established and mature Business Capability Centric Approach at the Enterprise Architecture design.	Enterprise Architecture	Case study of TOGAF framework in a telecommunication organization.	3) The Business Capability Centric Extension (BCCE) of TOGAF, which provide a linkage between Business Architecture, Data Architecture, Application Architecture and Technology of the Enterprise Architecture.	Enterprise Architecture, Integration and Interoperability, IFIP Advances in Information and

							Communication Technology (Volume 326, p.p. 32-43), Springer
Andreas Freitag, Florian Matthes & Christopher Schultz	2011	A Method for Business Capability Dependency Analysis	To present a three-phase method to systematically identify dependences between capabilities and to other elements of the Enterprise Architecture	Enterprise Architecture Management	Existing literature of research approach and for Business Capability. Also a Case Study of multinational telecommunication company.	1) A Capability Dependency Analysis Method. 2) Business Capabilities are core elements of the business Architecture and a communication between business and IT. 3) Business Capabilities support strategic planning and innovation.	International Conference on IT-enabled Innovation in Enterprise (ICITIE2011), Sofia 2011.
William Ulrich & Michael Rosen	2011	The Business Capability Map: The “Rosetta Stone” of Business/IT Alignment	To discuss how capability mapping enables business analysis and business/IT alignment.	Enterprise Architecture		1) A Capability Mapping Framework. 2) A method of Incorporating Capability into Business Architecture and Enterprise Architecture 3) A method for Business/IT roadmap development. 4) The Business Capability provides the high-level foundation for alignment and bridges the Business/IT Chasm.	Cutter Consortium, Enterprise Architecture Vol. 14, No 2.
Michael Vaughan	2011	A Focused Approach to	To create a focused and specific Business Capability definition that reduces confusion and enables clarity in	Enterprise Architecture	Review and categorization	1) An Architecture model with specific characteristic that enables to a definition of	The 1 st International

		Business Capability	defining capabilities within an enterprise.		existing definitions from the literature. Uses resource – based theory and operation theories from the literature.	Business Capabilities.	Symposium on Business Modeling and Software Design (BMSD 11) at Hilton Sofia Hotel, Sofia, Bulgaria.
Rostamzadeh Bakhtiyari & Mohammad Adel	2012	Business Capability and its strategic impacts	To discuss the strategic impacts of Business Capability.	Strategic Management	Literature relevant to Business Capabilities, competitive advance and value creation.	1) Capabilities are one of the most strategically relevant artefacts of an organization. 2) Capabilities enable the organization to perform at level that required to success. 3) Capabilities endow competitive advance.	Australasian Conference on Information Systems (ACIS 2012), 3-5 December 2012, Deakin University, Geelong, VIC.
Michael Rosen	2012	Processes, Value Streams and Capabilities	To discuss the difference between processes, values streams and business capabilities.	Business Management and IT Management		1) Processes describes how something is done 2) Value Streams describes how value is delivered to a stakeholder 3) Capability describes what is done regardless of how, where, who or how cell it is performed. 4) Processes and Value Streams require capabilities and describe how those capabilities are used.	A BPT trends column, December 2012. (www.bptrends.com)

Janis Stirna, Janis Grabis, Martin Henkel & Jelena Zdravkokvic	2012	Capability Driven Development – An Approach to Support Evolving Organizations	To propose an approach that integrates organizational development with information system (IS) development taking into account changes in the application context of the solution	Enterprise modeling, Business & IT alignment	Literature relevant to Enterprise Modeling, Context Representation & Service Specification. Also a Case Study from energy efficiency domain.	1) A Capability Driven Development meta – model	The Practice of Enterprise Modeling, Lecture Notes in Business Information Processing (Volume 134, 2012, p.p.117-131)
Frank J. Frey, Carsten Hentrich & Uwe Zdun	2013	Capability – based Service Identification in Service – Oriented Legacy Modernization	To describe the Capability – Based Service pattern that identifies services and defines the service model based on a model of Business Capabilities, in Legacy Systems transforming into a SOA.	Software Architecture Patterns, SOA, Legacy Modernization.	A Literature review on legacy to SOA transformation and a chosen of a top-down transformation strategy.	1) A Capability – Based Service Pattern that: <ul style="list-style-type: none"> ▪ Provide a solution to identify services candidates during a preliminary analysis phase of a modernization program. ▪ Requires detailed modeling of services and business process during the execution of the program. ▪ Facilitates a durable alignment between business and IT. 	Proceeding of the 17 th European Conference on Pattern Languages of Programs – EuroPLOP (Germany 2012, 2013)

10.2 Table 2: ORM 2 Graphical Notation

Construct	Examples	Description/Notes
Entity Type		Named soft rectangle, named hard rectangle, or named ellipse. The soft rectangle shape is the default.
Value Type		Named, dashed, soft rectangle (or hard rectangle or ellipse).
Entity type with popular reference mode		Abbreviation for injective reference relationship to value type, e.g.
Entity type with unit-based reference mode		Abbreviation for reference type, e.g. Optionally, unit type may be displayed.
Entity type with general reference mode		Abbreviation for reference type, e.g.
Independent Object Type		Instances of the type may exist, without playing any elementary fact roles
External Object Type		This notation is tentative (yet to be finalized)
Predicate (unary, binary, ternary, etc.)		Ordered set of 1 or more role boxes with at least one predicate reading in mixfix notation. If shown, object placeholders are denoted by "...". If placeholders are not shown, unaries are in prefix and binaries are in infix notation.
Duplicate type or predicate shape		If an object type or predicate shape is displayed more than once (on the same page or different pages) it is shadowed.
Unary fact type		The smokes role may be played by instances of the Person object type
Binary fact type		By default, predicate readings (binary or longer) are read left-to-right or top-to-bottom. An arrow-tip is used to display a different reading direction. Role names may be displayed in square brackets beside their role. Forward and inverse readings for binaries may be shown together, separated by "/".

Construct	Examples	Description/Notes
Ternary fact type		<p>Role names may be added in square brackets.</p> <p>Arrow-tips are used to reverse the default left-right or top-down reading order.</p> <p>Reading orders other than forward and reverse are shown using named placeholders.</p>
Quaternary fact type		<p>The above notes for the ternary case apply here also.</p> <p>Fact types of higher arity (number of roles) are also permitted.</p>
Objectification (a.k.a. nesting)		<p>The enrolment fact type is objectified as an entity type whose instances can play roles. In this example, the objectification type is independent, so we can know about an enrolment before the grade is obtained.</p>
Internal uniqueness constraint (UC) on unaries		<p>These are equivalent (by default, predicates are assumed to be populated with sets, so no whole fact may be duplicated).</p>
Internal UC on binaries		<p>The examples show the 4 possible patterns:</p> <p>1:n (one-to-many); n:1 (many-to-one); m:n (many-to-many); 1:1 (one-to-one)</p>
Internal UC on ternaries. For n -aries ($n > 1$) each UC must span at least $n-1$ roles		<p>The first example has two, 2-role UCs: the top UC forbids ties; the other UC ensures that each team gets only place per competition (a dotted line excludes its role from the UC).</p> <p>The second example has a spanning UC (many-to-many-to-many).</p>
Simple mandatory role constraint		<p>The example constraint means that each person was born in some country.</p> <p>The mandatory role dot may be placed at either end of the role connector.</p>
Inclusive-or constraint (disjunctive mandatory role)		<p>The constraint is displayed as a circled dot connected to the constrained roles. The example constraint means that each visitor referenced in the model must have a passport or a driver licence (or both).</p>
Preferred internal UC		<p>A double bar on a UC indicates it underlies the preferred reference scheme.</p>

Construct	Examples	Description/Notes
External UC (double-bar indicates preferred identifier)		Here, each state is primarily identified by combining its country and state code. Each combination of country and state name also applies to only one state.
Object Type Value Constraint	<div> Gender (.nr) {M, F} </div> <div> Rating (.nr) {1, 2, 3, 4, 5, 6, 7} </div> <div> Rating (.nr) {1, 7} </div> <div> Grade (.code) {A+, F} </div> <div> Age (y:) {0, } </div> <div> NegativeInt {-1} </div> <div> PassScore (%) {50, 100} </div> <div> PositiveScore (%) {{0, 100}} </div> <div> NegativeTemperature (°C) {-273.15, 0} </div> <div> ExtremeTemperature (°C) {-100, -20, 40, 100} </div> <div> SQLchar {a, ..., z, A, ..., Z, 0, ..., 9, ' ', ...} </div>	<p><i>Enumerations</i></p> <p>Ranges are inclusive of end values by default. Round brackets are used to exclude an end value. Square brackets may be added to explicitly declare inclusion, e.g. the constraint on PositiveScore may also be specified as {{0..100}}.</p> <p>Multiple combinations are allowed.</p>
Role value constraint		As for object type value constraints, but connected to the constrained role. Here, an age of a person must be at most 140 years.
Subset constraint		<p>The arrow points from the subset end to the superset end (e.g. if a person smokes then that person is cancer prone).</p> <p>The role sequences at both ends must be compatible.</p> <p>A connection to the junction of 2 roles constrains that role pair.</p>
Join subset constraint		The constrained role pair at the superset end is projected from a role path that involves a conceptual join on Language. The constraint declares that if an advisor serves in a country then that advisor must speak a language that is often used in that country.
Exclusion constraint		<p>These constraints mean that no person is both married and widowed, and no person reviewed and authored the same book.</p> <p>Exclusion may apply between 2 or more compatible role sequences, possibly involving joins.</p>
Exclusive-or constraint		<p>An exclusive-or constraint is simply the conjunction of an inclusive-or constraint and an exclusion constraint.</p> <p>Also known as an xor constraint.</p>

Construct	Examples	Description/Notes
Equality constraint		<p>This constraint means that a patient's systolic BP is recorded if and only if his/her diastolic BP is recorded.</p> <p>An equality constraint may apply between 2 or more compatible role sequences, possibly involving joins.</p>
Derived fact type, and derivation rule		<p>A fact type is either asserted, derived, or semiderived.</p> <p>A derived fact type is marked with an asterisk "*". A derivation rule is supplied. A double asterisk "**" indicates derived and stored (eager evaluation).</p>
Semiderived fact type, and derivation rule		<p>A fact type is semiderived if some of its instances may be derived, and some of its instances may be simply asserted.</p> <p>It is marked by "*" (half an asterisk).</p> <p>"**" indicates semiderived and stored (eager evaluation for derived instances).</p>
Subtyping		<p>All subtypes are proper subtypes. An arrow runs from subtype to supertype. A solid arrow indicates a path to the subtype's preferred identifier (e.g. here, student employees are primarily identified by their employee number). A dashed arrow indicates the supertype has a different preferred identifier.</p>
Subtyping constraints		<p>A circled "X" indicates the subtypes are mutually exclusive. A circled dot indicates the supertype equals the union of the subtypes. The combination (xor constraint) indicates the subtypes partition the supertype (exclusive and exhaustive).</p>
Subtype derivation status		<p>A subtype may be</p> <ul style="list-style-type: none"> asserted, derived (denoted by "*"), or semiderived (denoted by "*"). <p>If the subtype is asserted, it has no mark appended and has no derivation rule.</p> <p>If the subtype derived or semiderived, a derivation rule is supplied.</p>

Construct	Examples	Description/Notes
Internal frequency constraint		<p>This constrains the number of times an occurring instance of a role or role sequence may appear in each population.</p> <p>Here: each jury has exactly 12 members; each panel that includes an expert includes at least 4 and at most 7 experts; each expert reviews at most 5 papers; each paper that is reviewed is reviewed by at least 2 experts; and each department and year that has staff numbers recorded in the quaternary appears there twice (once for each gender).</p>
External frequency constraint		<p>The example constraint has the following meaning. In this context, each combination of student and course relates to at most two enrolments (i.e. a student may enroll at most twice in the same course)</p>
Ring constraints	<p>E.g.</p>	<p>A ring predicate R is locally reflexive if and only if, for all x and y, xRy implies xRx. E.g. "knows" is locally but not globally reflexive.</p> <p>Reflexive, symmetric and transitive properties may also be enforced using semiderivation rather than by constraining asserted fact types.</p> <p>The example constrains the subtyping relationship in ORM to be both acyclic (no cycles can be formed by a chain of subtyping connections) and strongly intransitive (no object type A can be both a direct subtype of another type B and an indirect subtype of B, where indirect subtyping means there is a chain of two or more subtyping relationships that lead from A to B).</p> <p>Ring constraints may be combined only if they are compatible, and one is not implied by the other. ORM tools ensure that only legal combinations are allowed.</p>
Value-comparison constraints		<p>The example constraint verbalizes as: For each Project, existing enddate \geq startdate.</p>

Construct	Examples	Description/Notes
Object cardinality constraint		The example constraints ensure there is exactly one president and at most 100 senators (at any given time).
Role cardinality constraint		The example constraint ensures that at most one politician plays the role of president (at any given time).
Deontic constraints	<p> <i>Uniqueness</i> <i>Mandatory</i> <i>Subset, Equality, Exclusion</i> <i>Frequency</i> <i>Irreflexive</i> <i>Acyclic</i> <i>Asymmetric</i> <i>Asym-Intrans</i> <i>Intransitive</i> <i>Acyclic-Intrans</i> <i>Antisymmetric</i> <i>Symmetric</i> <i>Strongly Intransitive</i> etc. </p> <p>e.g.</p> <p>is a parent of</p>	<p>Unlike alethic constraints, deontic constraint shapes are colored blue rather than violet. Most include "o" for "obligatory". Deontic ring constraints instead use dashed lines.</p> <p>In the parenthood example, the alethic frequency constraint ensures that each person has at most two parents, the alethic ring constraint ensures that parenthood is acyclic, and the deontic ring constraint makes it obligatory for parenthood to be strongly intransitive.</p>
Textual constraints	<p> ¹ Each Employee who has Rank 'NonExec' uses at most one CompanyCar. ² Each Employee who has Rank 'Exec' uses some CompanyCar. </p>	First-order constraints with no graphic notation may be expressed textually in the FORML 2 language. These examples use footnoting to capture a restricted uniqueness constraint and a restricted mandatory role constraint.
Objectification display options: link fact types, and compact display.	<p>Enrolment !</p>	<p>Internally, link fact types connect objectified associations to their component object types. By default, display of link fact types is suppressed. If displayed, link predicate shapes use dashed lines instead of solid lines.</p> <p>Objectification object types may also be displayed without their defining components, using an object type shape containing a small predicate shape, as shown in this Enrolment example.</p>

10.3 SQL Script of BC Tables

```
create table Incapability
(
  incapcode varchar2(50) not null,
  capdesc varchar2(250) not null,
  ownercode varchar2(50) not null,
  incapttype number (1) default 1 not null,
  constraint Incapability_PK primary key (incapcode)
);

comment on column incapability.incapcode is 'Internal Capability Code';
comment on column incapability.capdesc is 'Internal Capability Description';
comment on column incapability.ownercode is 'Internal Capability Owner';
comment on column incapability.incapttype is 'Internal Capability Type: (1: Main, 0:Sub)';

create table Excapability
(
  excapcode varchar2(50) not null,
  capdesc varchar2(250) not null,
  ownercode varchar2(50) not null,
  excapttype number (1) default 1 not null,
  constraint Excapability_PK primary key (excapcode)
);

comment on column excapability.excapcode is 'External Capability Code';
comment on column excapability.capdesc is 'External Capability Description';
comment on column excapability.ownercode is 'External Capability Owner';
comment on column excapability.excapttype is 'External Capability Type: (1: Main, 0:Sub)';

create table Owner
(
  ownercode varchar2(50) not null,
  ownername varchar2(100) not null,
  constraint Owner_PK primary key (ownercode)
);

comment on column owner.ownercode is 'Owner Code';
comment on column owner.ownername is 'Owner Name';

alter table Incapability
add constraint Incapability_FK foreign key (ownercode) references Owner (ownercode);

alter table Incapability
add constraint Incapttype_Value_Constraint check (incapttype in (0,1));

alter table Excapability
add constraint Excapability_FK foreign key (ownercode) references Owner (ownercode);

alter table Excapability
add constraint Excapttype_Value_Constraint check (excapttype in (0,1));

create table Capabilityispartof
(
  mcapcode varchar2(50) not null,
  subcapcode varchar2(50) not null,
  constraint Capabilityispartof_PK primary key (mcapcode, subcapcode)
);

comment on column capabilityispartof.mcapcode is 'Top level Business Capability Code: Main Capabilities';
comment on column capabilityispartof.subcapcode is 'Low level Business Capability Code: Sub Capabilities';

alter table Capabilityispartof
add constraint Capabilityispartof_RC check (mcapcode<>subcapcode);
create table Contexts
(
  contcode varchar2(50) not null,
  contdesc varchar2(150) not null,
```

```

constraint Contexts_PK primary key (contcode)
);

comment on column contexts.contcode is 'Context Code';
comment on column contexts.contdesc is 'Context Description';

create table Capabilityisincontext
(
capcode varchar2(50) not null,
contcode varchar2(50) not null,
constraint Capabilityisincontext_PK primary key (capcode, contcode)
);

comment on column capabilityisincontext.capcode is 'Internal and External Capabiltiy Code';
comment on column capabilityisincontext.contcode is 'Context Code';

alter table Capabilityisincontext
add constraint Capabilityisincontext_FK foreign key (contcode) references Contexts (contcode)on delete cascade;

create table Goal
(
goalcode varchar2(50) not null,
goalname varchar2(250) not null,
goaltype number (1)default 1 not null,
constraint Goal_PK primary key (goalcode)
);

comment on column goal.goalcode is 'Goal Code';
comment on column goal.goalname is 'Goal Name';
comment on column goal.goaltype is 'Goal Type: (1: Main, 0:Sub)';

alter table Goal
add constraint Goaltype_Value_Constraint check (goaltype in (0,1));

create table Capabilitymeetsgoal
(
capcode varchar2(50) not null,
goalcode varchar2(50) not null,
constraint Capabilitymeetsgoal_PK primary key (capcode, goalcode)
);

comment on column capabilitymeetsgoal.capcode is 'Internal and External Capabiltiy Code';
comment on column capabilitymeetsgoal.goalcode is 'Goal Code: Strategic and Operational Goals Codes';

alter table Capabilitymeetsgoal
add constraint Capabilitymeetsgoal_FK foreign key (goalcode) references Goal (goalcode);

create table Goalispartof
(
mgoalcode varchar2(50) not null,
subgoalcode varchar2(50) not null,
constraint Goalispartof_PK primary key (mgoalcode, subgoalcode)
);

comment on column goalispartof.mgoalcode is 'Top level Business Goal Code: Strategic Goal Code';
comment on column goalispartof.subgoalcode is 'Low level Busines Goal Code: Operational Goal Code';

alter table Goalispartof
add constraint Goalispartof_FK1 foreign key (mgoalcode) references Goal (goalcode);
alter table Goalispartof
add constraint Goalispartof_FK2 foreign key (subgoalcode) references Goal (goalcode);

alter table Goalispartof
add constraint Goalispartof_RC check (mgoalcode<>subgoalcode);

create table Output
(
outputcode varchar2(50) not null,
outputname varchar2(250) not null,
capcode varchar2(50) not null,
outputtype number (1)default 1 not null,

```

```

constraint Output_PK primary key (outputcode)
);

comment on column output.outputcode is 'Output Code';
comment on column output.outputname is 'Output Name';
comment on column output.capcode is 'Internal and External Capability Code';
comment on column output.outputtype is 'Output Type: (1: Main, 0:Sub)';

alter table Output
add constraint Outputtype_Value_Constraint check (Outputtype in (0,1));

create table Outputisofvalue
(
outputcode varchar2(50) not null,
evalvalue number(20,2) not null,
recipientname varchar2(250) not null,
constraint Outputisofvalue_PK primary key (outputcode, evalvalue)
);

comment on column outputisofvalue.outputcode is 'Output Code';
comment on column outputisofvalue.evalvalue is 'Output Economic Value';
comment on column outputisofvalue.recipientname is 'Recipient Name';

alter table Outputisofvalue
add constraint Outputisofvalue_FK foreign key (outputcode) references Output (outputcode) on delete cascade;

create table Outputispartof
(
moutputcode varchar2(50) not null,
suboutputcode varchar2(50) not null,
constraint Outputispartof_PK primary key (moutputcode, suboutputcode)
);

comment on column outputispartof.moutputcode is 'Top level Output Code: Main Output';
comment on column outputispartof.suboutputcode is 'Low level Output Code: Sub Output';

alter table Outputispartof
add constraint Outputispartof_FK1 foreign key (moutputcode) references Output (outputcode) on delete cascade;

alter table Outputispartof
add constraint Outputispartof_FK2 foreign key (suboutputcode) references Output (outputcode) on delete cascade;

alter table Outputispartof
add constraint Outputispartof_RC check (moutputcode<>suboutputcode);

create table Collaborator
(
collabcode varchar2(50) not null,
connectortype varchar2(20),
evalvalue number (20,2) not null,
constraint Collaborator_PK primary key (collabcode)
);

comment on column collaborator.collabcode is 'Collaborator Connector Code';
comment on column collaborator.connectortype is 'The type of Collaborator Connector';
comment on column collaborator.evalvalue is 'The economic value of Collaborator Connector';

alter table Collaborator
add constraint Collaborator_Value_Constraint check (connectortype in ('POLICY','INFORMATION','PROCEDURE'));

create table Policy
(
collabcode varchar2(50) not null,
constraint Policy_PK primary key (collabcode)
);

comment on column policy.collabcode is 'Policy Collaborator Connector Code';

alter table Policy
add constraint Policy_FK foreign key (collabcode) references Collaborator (collabcode);

```

```

create table Information
(
collabcode varchar2(50) not null,
constraint Information_PK primary key (collabcode)
);

comment on column information.collabcode is 'Information Collaborator Connector Code';

alter table Information
add constraint Information_FK foreign key (collabcode) references Collaborator (collabcode);

create table Procedures
(
collabcode varchar2(50) not null,
constraint Procedures_PK primary key (collabcode)
);

comment on column procedures.collabcode is 'Procedure Collaborator Connector Code';

alter table Procedures
add constraint Procedures_FK foreign key (collabcode) references Collaborator (collabcode);

create table Collaborations
(
capcode1 varchar2(50) not null,
capcode2 varchar2(50) not null,
collabcode varchar2(50) not null,
constraint Collaborations_PK primary key (capcode1, capcode2, collabcode)
);

comment on column collaborations.capcode1 is 'Internal and External Capability Code';
comment on column collaborations.capcode2 is 'Internal and External Capability Code';
comment on column collaborations.collabcode is 'Collaborator Connector Code';

alter table Collaborations
add constraint Collaborations_FK foreign key (collabcode) references Collaborator (collabcode)on delete cascade;

create table Inability
(
inabcode varchar2(50) not null,
abdescr varchar2(250) not null,
evalue number (20,2) not null,
inabtype number (1) default 1 not null,
constraint Inability_PK primary key (inabcode)
);

comment on column inability.inabcode is 'Internal Ability Code';
comment on column inability.abdescr is 'Internal Ability Description';
comment on column inability.evalue is 'Internal Ability Economic Value';
comment on column inability.inabtype is 'Internal Ability Type: 1';

alter table Inability
add constraint Inabtype_value_constraint check (inabtype= '1');

create table Exability
(
exabcode varchar2(50) not null,
abdescr varchar2(250) not null,
evalue number (20,2) not null,
exabtype number (1) default 0 not null,
constraint Exability_PK primary key (exabcode)
);

comment on column exability.exabcode is 'External Ability Code';
comment on column exability.abdescr is 'External Ability Description';
comment on column exability.evalue is 'External Ability Economic Value';
comment on column exability.exabtype is 'External Ability Type= 0';

alter table Exability
add constraint Exabtype_value_constraint check (exabtype= '0');

```

```

create table Skill
(
    skillcode varchar2(50) not null,
    skillname varchar2(250) not null,
    abcode varchar2(50),
    constraint Skill_PK primary key (skillcode)
);

comment on column skill.skillcode is 'Skill Code';
comment on column skill.skillname is 'Skill Name';
comment on column skill.abcode is 'Internal and External Ability Code';

create table Process
(
    processcode varchar2(50) not null,
    processname varchar2(250) not null,
    taskname varchar2(1000),
    constraint Process_PK primary key (processcode)
);

comment on column process.processcode is 'Business Process Code';
comment on column process.processname is 'Business Process Name';
comment on column process.taskname is 'Tasks Name of Business Process';

create table Service
(
    servcode varchar2(50) not null,
    servname varchar2(250) not null,
    processcode varchar2(50),
    servtype number(1) default 1 not null,
    constraint Service_PK primary key (servcode)
);

comment on column service.servcode is 'Service Code';
comment on column service.servname is 'Service Name';
comment on column service.processcode is 'Business Process Code';
comment on column service.servtype is 'Service Type: (1: Main, 0:Sub)';

alter table Service
add constraint Service_FK foreign key (processcode) references Process (processcode);

alter table Service
add constraint Servtype_Value_Constraint check (servtype in (0,1));

create table Serviceispartof
(
    mservcode varchar2(50) not null,
    subservcode varchar2(50) not null,
    constraint Serviceispartof_PK primary key (mservcode, subservcode)
);

comment on column serviceispartof.mservcode is 'Top level Service Code: Main Service';
comment on column serviceispartof.subservcode is 'Low level Service Code: Sub Service';

alter table Serviceispartof
add constraint Serviceispartof_FK1 foreign key (mservcode) references Service (servcode) on delete cascade;

alter table Serviceispartof
add constraint Serviceispartof_FK2 foreign key (subservcode) references Service (servcode) on delete cascade;

alter table Serviceispartof
add constraint Serviceispartof_RC check (mservcode <> subservcode);

create table Incapacity
(
    incapaccode varchar2(50) not null,
    capacdescr varchar2(250) not null,
    evalvalue number(20,2) not null,
    incapactype number(1) default 1 not null,
    constraint Incapacity_PK primary key (incapaccode)
);

```

```

comment on column incapacity.incapaccode is 'Internal Resource Set/Internal Capacity Code';
comment on column incapacity.capacdescr is 'Internal Resource Set/Internal Capacity Description';
comment on column incapacity.evalue is 'Internal Resource Set/Internal Capacity Economic Value';
comment on column incapacity.incapactype is 'Incapacity Type= 1';

alter table Incapacity
add constraint Incapactype_value_constraint check (incapactype= '1');

create table Excapacity
(
    excapaccode varchar2(50) not null,
    capacdescr varchar2(250) not null,
    evalue number (20,2) not null,
    excapactype number(1) default 0 not null,
    constraint Excapacity_PK primary key (excapaccode)
);

comment on column excapacity.excapaccode is 'External Resource Set/External Capacity Code';
comment on column excapacity.capacdescr is 'External Resource Set/External Capacity Description';
comment on column excapacity.evalue is 'External Resource Set/External Capacity Economic Value';
comment on column excapacity.excapactype is 'Excapacity Type= 0';

alter table Excapacity
add constraint Excapactype_value_constraint check (excapactype= '0');
create table Resources
(
    rescode varchar2(50) not null,
    restype varchar2(10) not null,
    resdescr varchar (1000) not null,
    capaccode varchar2(50),
    constraint Resources_PK primary key (rescode)
);

comment on column resources.rescode is 'Resource Code';
comment on column resources.restype is 'Resource Type Values';
comment on column resources.resdescr is 'Resources Description';
comment on column resources.capaccode is 'Internal Resource Set/Internal Capacity and External Resource Set/Internal Capacity Code';

alter table Resources
add constraint Resources_Value_Constraint check (restype in ('PH','LE','PR','HU','TE','FI','DA'));

create table Usescapacityforservice
(
    incapcode varchar2(50) not null,
    capaccode varchar2(50) not null,
    servcode varchar2(50) not null,
    constraint Usescapacityforservice_PK primary key (incapcode, capaccode)
);

comment on column usescapacityforservice.incapcode is 'Internal Capability Code';
comment on column usescapacityforservice.capaccode is 'Internal Resource Set/Internal Capacity and External Resource Set/Internal Capacity Code';
comment on column usescapacityforservice.servcode is 'Service Code';

alter table Usescapacityforservice
add constraint Usescapacityforservice_FK1 foreign key (incapcode) references Incapability (incapcode)on delete cascade;

alter table Usescapacityforservice
add constraint Usescapacityforservice_FK2 foreign key (servcode) references Service (servcode)on delete cascade;

create table Usesabilityforservice
(
    incapcode varchar2(50) not null,
    abcode varchar2(50) not null,
    servcode varchar2(50)not null,
    constraint Usesabilityforservice_PK primary key (incapcode, abcode)
);

comment on column usesabilityforservice.incapcode is 'Internal Capability Code';
comment on column usesabilityforservice.abcode is 'Internal Ability and External Ability Code';

```

```

comment on column usesabilityforservice.servcode is 'Service Code';

alter table Usesabilityforservice
add constraint Usesabilityforservice_FK1 foreign key (incapcode) references Incapability (incapcode)on delete cascade;

alter table Usesabilityforservice
add constraint Usesabilityforservice_FK2 foreign key (servcode) references Service (servcode)on delete cascade;

create table Abilityusescapacitywithservice
(
abcode varchar2(50) not null,
capaccode varchar2(50) not null,
servcode varchar2(50) not null,
incapcode varchar2(50) not null,
constraint Abusescapwithserv_PK primary key (abcode, capaccode, servcode)
);
comment on column abilityusescapacitywithservice.abcode is 'Internal Ability and External Ability Code';
comment on column abilityusescapacitywithservice.capaccode is 'Internal Resource Set/Internal Capacity and External Resource Set/Internal Capacity Code';
comment on column abilityusescapacitywithservice.servcode is 'Service Code';
comment on column abilityusescapacitywithservice.incacode is 'Internal Capability Code';

alter table Abilityusescapacitywithservice
add constraint Abusescapwithserv_FK1 foreign key (incapcode) references Incapability (incapcode)on delete cascade;

alter table Abilityusescapacitywithservice
add constraint Abusescapwithserv_FK2 foreign key (servcode) references Service (servcode)on delete cascade;

create table Menu
s_num varchar2 (50) not null,
name varchar2(250),
menu_name varchar2 (250)
constraint Menu_PK primary key (s_num)
);

comment on column menu.s_num is 'Unique Identifier of Parent and Child Tree Nodes';
comment on column menu.name is 'Parent and Child Tree Nodes Names';
comment on column menu.menu_name is 'Child Tree Node Values';

create sequence OWNER_SEQ
minvalue 1
maxvalue 99999
start with 1
increment by 1
cache 20;

create sequence SKILL_SEQ
minvalue 1
maxvalue 99999
start with 1
increment by 1
cache 20;

create sequence PROCESS_SEQ
minvalue 1
maxvalue 99999
start with 1
increment by 1
cache 20;

create sequence CONTEXTS_SEQ
minvalue 1
maxvalue 99999
start with 1
increment by 1
cache 20;

create sequence PROCEDURE_SEQ
minvalue 1
maxvalue 99999
start with 1

```

```

increment by 1
cache 20;

create sequence INFORMATION_SEQ
minvalue 1
maxvalue 99999
start with 1
increment by 1
cache 20;

create sequence SKILL_SEQ
minvalue 1
maxvalue 99999
start with 1
increment by 1
cache 20;

```

10.4 SQL Script of Total View

```

create or replace view final_view as
select a.incapcode cd, a.capdesc dscr, a.ownercode, o.ownname, 'Internal' type1, a.incaptype typ, null capcode, null
capdesc, null servcode, null servname, 1 qry_flg
from incapability a, owner o
where a.ownercode = o.ownercode
UNION
select b.excapcode cd, b.capdesc dscr, b.ownercode, o.ownname, 'External' type1, b.excaptype typ, null capcode, null
capdesc, null servcode, null servname, 1 qry_flg
from excapability b, owner o
where b.ownercode = o.ownercode

/*****SUBCAPABILITIES*****/
UNION
select a.incapcode cd, a.capdesc dscr, null ownwercode, null ownwename, 'IN' type1, a.incaptype typ, cp.mcapcode, null
capdesc, null servcode, null servname, 2 qry_flg
from incapability a, capabilityispartof cp
where a.incapcode= cp.subcapcode
UNION
select b.excapcode cd, b.capdesc dscr, null ownwercode, null ownwename, 'EX' type1, b.excaptype typ, cp.mcapcode, null
capdesc, null servcode, null servname, 2 qry_flg
from excapability b, capabilityispartof cp
where b.excapcode= cp.subcapcode

/***** OUTPUTS *****/
UNION
select ot.outputcode cd, ot.outputname dscr, null ownercode, null ownname, 'IN' type1, ot.outputtype typ, ot.capcode
capcode, a.capdesc capdesc, null servcode, null servname, 3 qry_flg
from output ot, incapability a
where ot.capcode = a.incapcode
UNION

```

```

select ot.outputcode cd, ot.outputname dscr, null ownercode, null ownername, 'EX' type1, ot.outputtype typ, ot.capcode
capcode, b.capdesc capdesc, null servcode, null servname, 3 qry_flg
from output ot, excapability b
where ot.capcode = b.excapcode

/***** GOALS *****/
UNION
select g.goalcode cd, g.goalname dscr, null ownercode, null ownername, 'IN' type1, g.goaltyp typ, cmg.capcode capcode,
a.capdesc capdesc, null servcode, null servname, 4 qry_flg
from goal g, capabilitymeetsgoal cmg, incapability a
where g.goalcode = cmg.goalcode(+)
and cmg.capcode = a.incapcode(+)
and cmg.capcode(+) like 'IN%'
UNION
select g.goalcode cd, g.goalname dscr, null ownercode, null ownername, 'EX' type1, g.goaltyp typ, cmg.capcode capcode,
b.capdesc capdesc, null servcode, null servname, 4 qry_flg
from goal g, capabilitymeetsgoal cmg, excapability b
where g.goalcode = cmg.goalcode(+)
and cmg.capcode = b.excapcode(+)
and cmg.capcode(+) like 'EX%'

/***** CONTEXT *****/
UNION
select ctx.contcode cd, ctx.contdesc dscr, null ownercode, null ownername, 'IN' type1, null typ, cix.capcode capcode, a.capdesc
capdesc, null servcode, null servname, 5 qry_flg
from contexts ctx, capabilityisincontext cix, incapability a
where ctx.contcode = cix.contcode(+)
and cix.capcode = a.incapcode (+)
and cix.capcode(+) like 'IN%'
UNION
select ctx.contcode cd, ctx.contdesc dscr, null ownercode, null ownername, 'EX' type1, null typ, cix.capcode capcode, b.capdesc
capdesc, null servcode, null servname, 5 qry_flg
from contexts ctx, capabilityisincontext cix, excapability b
where ctx.contcode = cix.contcode(+)
and cix.capcode = b.excapcode (+)
and cix.capcode(+) like 'EX%'

/***** COLLABORATIONS *****/
UNION
select cl.capcode1 cd, a.capdesc dscr, cl.collabcode ownercode, col.connectortype ownercode, 'IN' type1, a.incaptype typ,
cl.capcode2 capcode, a1.capdesc capdesc, null servcode, null servname, 6 qry_flg
from collaborations cl, incapability a, collaborator col, incapability a1
where cl.capcode1 = a.incapcode
and cl.collabcode = col.collabcode
and cl.capcode2 = a1.incapcode
UNION
select cl.capcode1 cd, a.capdesc dscr, cl.collabcode ownercode, col.connectortype ownercode, 'IN' type1, a.incaptype typ,

```

```

cl.capcode2 capcode, b.capdesc capdesc, null servcode, null servname, 6 qry_flg
from collaborations cl, incapability a, collaborator col, excapability b
where cl.capcode1 = a.incapcode
and cl.collabcode = col.collabcode
and cl.capcode2 = b.excapcode
UNION
select cl.capcode1 cd, b.capdesc dscr, cl.collabcode ownercode, col.connectortype ownercode, 'EX' type1, b.excaptype typ,
cl.capcode2 capcode, a.capdesc capdesc, null servcode, null servname, 6 qry_flg
from collaborations cl, excapability b, collaborator col, incapability a
where cl.capcode1 = b.excapcode
and cl.collabcode = col.collabcode
and cl.capcode2 = a.incapcode
UNION
select cl.capcode1 cd, b.capdesc dscr, cl.collabcode ownercode, col.connectortype ownercode, 'EX' type1, b.excaptype typ,
cl.capcode2 capcode, b1.capdesc capdesc, null servcode, null servname, 6 qry_flg
from collaborations cl, excapability b, collaborator col, excapability b1
where cl.capcode1 = b.excapcode
and cl.collabcode = col.collabcode
and cl.capcode2 = b1.excapcode

/***** USES ABILITY FOR SERVICE *****/
UNION
select uafs.incapcode cd, a.capdesc dscr, a.ownercode, o.ownername, 'IN' type1, a.incaptype typ, uafs.abcode capcode,
ia.abdescr capdesc, uafs.servcode, s.servname, 7 qry_flg
from usesabilityforservice uafs, incapability a, owner o, inability ia, service s
where uafs.incapcode = a.incapcode
and a.ownercode = o.ownercode
and ia.inabcode = uafs.abcode
and uafs.servcode = s.servcode
UNION
select uafs.incapcode cd, a.capdesc dscr, a.ownercode, o.ownername, 'IN' type1, a.incaptype typ, uafs.abcode capcode,
ea.abdescr capdesc, uafs.servcode, s.servname, 7 qry_flg
from usesabilityforservice uafs, incapability a, owner o, exability ea, service s
where uafs.incapcode = a.incapcode
and a.ownercode = o.ownercode
and ea.exabcode = uafs.abcode
and uafs.servcode = s.servcode

/***** USES CAPACITY FOR SERVICE *****/
UNION
select ucfs.incapcode cd, a.capdesc dscr, a.ownercode, o.ownername, 'IN' type1, a.incaptype typ, ucfs.capaccode capcode,
ic.capacdesc capdesc, ucfs.servcode, s.servname, 8 qry_flg
from usescapacityforservice ucfs, incapability a, owner o, incapacity ic, service s
where ucfs.incapcode = a.incapcode
and a.ownercode = o.ownercode
and ic.incapaccode = ucfs.capaccode
and ucfs.servcode = s.servcode

```

UNION

```
select ucfs.incapcode cd, a.capdesc dscr, a.ownercode, o.ownname, 'IN' type1, a.incaptype typ, ucfs.capaccode capcode,
ec.capacdescr capdesc, ucfs.servcode, s.servname, 8 qry_flg
from usescapacityforservice ucfs, incapability a, owner o, excapacity ec, service s
where ucfs.incapcode = a.incapcode
and a.ownercode = o.ownercode
and ec.excapaccode = ucfs.capaccode
and ucfs.servcode = s.servcode
```

*/***** ABILITY USES CAPACITY WITH SERVICE *****/*

UNION

```
select aucs.abcode cd, ia.abdescr dscr, a.incapcode ownercode, a.capdesc ownname, 'IN' type1, null typ, aucs.capaccode,
ic.capacdescr capdesc, aucs.servcode, s.servname, 9 qry_flg
from abilityusescapacitywithservice aucs, inability ia, incapacity ic, service s, incapability a
where aucs.abcode = ia.inabcode
and aucs.capaccode = ic.incapaccode
and aucs.servcode = s.servcode
and aucs.incapcode = a.incapcode
```

UNION

```
select aucs.abcode cd, ia.abdescr dscr, a.incapcode ownercode, a.capdesc ownname, 'IN' type1, null typ, aucs.capaccode,
ec.capacdescr capdesc, aucs.servcode, s.servname, 9 qry_flg
from abilityusescapacitywithservice aucs, inability ia, excapacity ec, service s, incapability a
where aucs.abcode = ia.inabcode
and aucs.capaccode = ec.excapaccode
and aucs.servcode = s.servcode
and aucs.incapcode = a.incapcode
```

UNION

```
select aucs.abcode cd, ea.abdescr dscr, aucs.incapcode ownercode, a.capdesc ownname, 'IN' type1, null typ, aucs.capaccode,
ic.capacdescr capdesc, aucs.servcode, s.servname, 9 qry_flg
from abilityusescapacitywithservice aucs, exability ea, incapacity ic, service s, incapability a
where aucs.abcode = ea.exabcode
and aucs.capaccode = ic.incapaccode
and aucs.servcode = s.servcode
and aucs.incapcode = a.incapcode
```

UNION

```
select aucs.abcode cd, ea.abdescr dscr, aucs.incapcode ownercode, a.capdesc ownname, 'IN' type1, null typ, aucs.capaccode,
ec.capacdescr capdesc, aucs.servcode, s.servname, 9 qry_flg
from abilityusescapacitywithservice aucs, exability ea, excapacity ec, service s, incapability a
where aucs.abcode = ea.exabcode
and aucs.capaccode = ec.excapaccode
and aucs.servcode = s.servcode
and aucs.incapcode = a.incapcode;
```

CHAPTER 11: Bibliography

- (1997). *UML Summary: Version 1.1*. Rational Software, Microsoft, Hewlett-Packard, Oracle Sterling Software, MCI Systemhouse, Unisys, ICON Computing IntelliCorp, i-Logix, IBM, ObjecTime, Platinum Technology, Ptech Taskon, Reich Technologies, Softeam.
- FCO-IM: Fully Communication Oriented Information Modeling*. (2015). Retrieved 2015, from FCO-IM Foundation: www.fco-im.nl: <http://fco-im.nl/>
- Object Management Group*. (2015). Retrieved 2015, from Unified Modeling Language™ (UML): <http://www.omg.org/spec/UML/>
- Anand, P., Hunter, G., & Smith, R. (2005). Capabilities and Well-Being: Evidence Based on the Sen-Nussbaum Approach to Welfare. *Social Indicators Research*, 74, 9-55.
- Bachman, C. W. (1969). Data Structure Diagrams. 1(2), 4-10.
- Bakema, G., Zwart, J. P., & Lek, H. (2002). *Fully Communication Oriented Information Modeling (FCO-IM)*. FCO-IM Consultancy.
- Bakhtiyari, R., & Adel, M. (2012). Business Capability and its Strategic Impacts. *Australasian Conference on Information Systems*. Geelong, Australia: Deakin University.
- Barney, J. (1991). Firm Resources and Sustained Competitive Advance. *Journal of Management*, 17, 99-120.
- Barroero, T., Motta, G., & Pign, G. (2010). Business Capabilities Centric Enterprise Architecture. *Enterprise Architecture, Integration and Interoperability. IFIP Advances in Information and Communication Technology*. 326, pp. 32-43. Springer.
- Barton, D. L. (1992). Core Capabilities and Core Rigidities: A Paradox in Managing New Product Development. *Strategic Management Journal*, 13, 111-115.
- Beimborn, D., Martin, S. F., & Holman, U. (2005). Capability-oriented Modeling of the Firm. *Proceedings of the IPSI 2005 Conference*.
- Berzisa, S., Bravos, G., Gonzalez - Cardona, T., Czubayko, U., Espana, S., Grabis, J., et al. (2015). Capability Driven Development: An Approach to Designing Digital Enterprises. *Business & Information Systems Engineering*. 57, pp. 15-25. Springer.
- Berzisa, S., Espana, S., Grabis, J., Henkel, M., Jokste, L., Kampars, J., et al. (2013, September 1). Task 5.1 Result Report: State of Art in Relevant Methodology Areas . *Capability as Service in Digital Enterprises - Collaborative Project Number 611351* .
- Bevan, N. (1999). Quality in Use: Meeting User Needs for Quality. *Journal of System and Software*.
- Bommel, P., Hofstede, A., & Weide, T. (1993). Semantics and verification of object-role models. *Information Systems*, 16(5), 471-495.

- Bravos, G., Gonzslez, T., Grabis, J., Henkel, M., & Jokste, L. (2014). Capability Modeling: Initial Experiences. *Perspectives in Business Informatics Research - Lecture Notes in Business Information Processing*. 194, pp. 1-14. Springer.
- Bravos, G., Grabis, J., Henkel, M., Jokste, L., & Kampars, J. (2014). Supporting Evolving Organizations: IS Development Methodology Goals. *Perspectives in Business Informatics Research - Lecture Notes in Business Information Processing*. 194, pp. 158-171. Springer.
- Bravos, G., Loucopoulos, P., Stratigaki, C., & Valvis, D. (2014). An Empirical Evaluation of Capability Modelling using Design Rationale. *The 1st International Workshop on Capability-oriented Business Informatics (CoBI 2014)*. CEUR Workshop Proceedings (CEUR-WS.org).
- Brits, J., Botha, G., & Herselman, T. M. (2007). Conceptual Framework for Modeling Business Capabilities. *Proceedings of the 2007 Informing Science and IT Education Joint Conference*, 151-170.
- Chan, M., & Cheung, S. (1999). Testing Database Applications with SQL Semantics. *Proceedings of 2nd International Symposium on Cooperative Database Systems for Advanced Applications*, 363-374.
- Chen, C., Song, Y., & Zhu, W. (2007). Trends in Conceptual Modeling: Citation Analysis of the ER Conference Papers 1979-2005. In D. Torres-Salinas, & H. F. Moed (Ed.), *Proceedings of the 11th International Conference on the International Society for Scientometrics and Informetrics* (pp. 189-200). Madrid: CSIC.
- Chen, P. P.-S. (1976, March). *The entity-relationship model*. Retrieved February 2015, from <http://www.minet.uni-jena.de/dbis/lehre/ws2005/dbs1/Chen.pdf>
- Connolly, T., & Begg, C. E. (2005). *Database Systems: A Practical Approach to Design, Implementation and Management* (Fourth ed.). Addison Wesley.
- Cook, D. (2007, March). *Business-Capability Mapping: Staying Ahead of the Joneses*. Retrieved May 2015, from Microsoft Corporation MSDN Library: <https://msdn.microsoft.com/en-us/library/bb402954.aspx>
- Curland, M., & Halpin, T. (2010). The NORMA Tool for ORM 2. *Proceedings of the CAiSE Forum 2010*. 592. Tunisia: CEUR Workshop Proceeding (<http://ceur-ws.org/>).
- Cusick, K. (1997). The Systems Engineering Capability Maturity Model: Where to start? *Proceedings of The IEEE 1997 Aerospace and Electronics Conference (NAECON)*. 1, pp. 410-416. IEEE Xplore Digital Library.
- Cuyler, D., & Halpin, T. (2003). *Metamodels for Object-Role Modeling*. Retrieved 2015, from citeseerx.ist.psu.edu: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.4841&rep=rep1&type=pdf>

- Cuyler, D., & Halpin, T. (2003). *Metamodels for Object-Role Modeling*. Retrieved 2015, from <http://citeseerx.ist.psu.edu:>
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.4841&rep=rep1&type=pdf>
- Darke, P., & Shanks, G. (1995). Defining System Requirements: A Critical Assessment of the NIAM Conceptual Design Procedure. *Australasian Journal of Information Systems*, 2(2).
- Deneulin, S., & McGregor, A. J. (2010). The Capability Approach and the Politics of a Social Conception of Wellbeing. *European Journal of Social Theory*, 3(4), 501–519.
- Duhs, L. A. (2008). Sen's Economic Philosophy: Capabilities and Human Development in the Revival of Economics as a Moral Science . *School of Economics Discussion*.
- Elmasri, R., & Navathe, S. B. (2004). *Fundamentals to Database Systems*. (M. Hirsch, Ed.) Addison-Wesley.
- Embley, D. W., & Ling, T. W. (1989). Synergistic Database Design with an Extended Entity-Relationship Model. *Proceedings of the Eight International Conference on Entity-Relationship Approach*, (pp. 111-128). Toronto.
- Espana, S., Gonzalez, T., Grabis, J., Jokste, L., Juanes, R., & Valverde, F. (2014). Capability-driven development of a SOA platform: A Case Study. *Advanced Information Systems Engineering Workshops - Lecture Notes in Business Information Processing*, 178, 100-111.
- European Commission. (2013). *Capability as Service in Digital Enterprises*. Retrieved May 2015, from CORDIS - Community Research and Development Information Service: http://cordis.europa.eu/project/rcn/109917_en.html
- Finkelstein , C. (1981-C, June 8). Information Engineering: Part 5. *Computer World*, XV, pp. 31-40.
- Finkelstein , C. (1981-D, June 15). Information Engineering: Part 6 - Infomethod. The Information Engineering Development Plan. *Computer World*, XV, pp. InDepth 1 - 8.
- Finkelstein, C. (1981-A, May 25). Information Engineering: Part3 - Information Analysis. Developing A Corporate Model. *Computer World*, XV, pp. 29-36.
- Finkelstein, C. (1981-B, June 1). Information Engineering: Part 4 - Data Analysis and Database Design. *Computer World*, XV, pp. InDepth 1 - 12.
- Finkelstein, C. (2006). Information Engineering Methodology. In P. Bernus, K. Mertins, & G. Schmidt (Eds.), *Handbook on Architectures of Information Systems* (pp. 460-485). Springer Berlin Heidelberg.
- Fowler, M., & Scott, K. (1999). *UML Distilled Second Edition A Brief Guide to the Standard Object Modeling Language* (Second ed.). Addison Wesley Longman Inc.

- FP7 Collaborative Project with No 611351 . (2014, July). *CaaS - Capability as Service in Digital Enterprises*. Retrieved May 2015, from Slideshare.net:
http://www.slideshare.net/fp7_caas/fp7-capability-as-a-service-caas?next_slideshow=1
- Freitag, A., Matthes, F., Schu, C., & Nowobilska, A. (2011). *A Method for Business Capability Dependency Analysis*. Retrieved May 2015, from CiteSeerX:
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.363.9176&rep=rep1&type=pdf>
- Frey, F. J., Hentrich, C., & Zdun, U. (2013). Capability-based Service Identification in Service-Oriented Legacy Modernization. *Proceedings of the 17th European Conference on Pattern Languages of Programs (EuroPLOP)*. Kloster Irsee, Germany.
- Geisler, R., Klar, M., & Pons, C. (1998). Dimensions and dichotomy in metamodeling. *3FACS'98 Proceedings of the 3rd BCS-FACS conference on Northern Formal Methods* (pp. 10-10). ACLDL.
- Gregersen, H., & Jensen, C. S. (1999). Temporal Entity-Relationship Models: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 11(3).
- Greski, L. (2009-A). Business Capability Modeling: Building Hierarchy. *Architecture & Governance Magazine*, 5(11), pp. 1-4.
- Greski, L. (2009-B, September 14-15). Business Capability Modeling: Theory & Practice. *Architecture & Governance Magazine*, 5(7), pp. 1-4.
- Halpin, T. (2010). *ORM 2 Graphical Notation*. Retrieved June 2015, 30, from Object Role Modeling: The Official Site for Conceptual Data Modeling:
<http://www.orm.net/pdf/ORM2GraphicalNotation.pdf>
- Halpin, T. (1989). *A Logical Analysis of Information Systems: Static Aspects of the Data-Oriented Perspective*. University of Queensland, Department of Computer Science. Brisbane: www.orm.net.
- Halpin, T. (1991). A fact-oriented approach to schema transformation. In B. Thalheim, J. Demetrovics, & H. Gerhardt (Eds.), *MFDBS 91: 3rd Symposium on Mathematical Fundamentals of Database and Knowledge Base Systems Rostock* (pp. 342-356). Berlin: Springer Berlin Heidelberg.
- Halpin, T. (1995-A). *Conceptual Schema & Relational Database Design* (Second ed.). (K. Smith, Ed.) Sydney: Prentice Hall of Australia Pty Ltd.
- Halpin, T. (1995-B). Subtyping: conceptual and logical issues. (R. G. Ross, Ed.) *Database Newsletter*, 23, 3-9.
- Halpin, T. (1996). Business Rules and Object - Role Modeling. *Database Programming & Design*.

- Halpin, T. (1999, February). UML data models from an ORM perspective: Part 7. *Journal of Conceptual Modeling*.
- Halpin, T. (2001). *Informational Modeling and Relational Database: From Conceptual Analysis to Logical Design*. (J. Gray, Ed.) San Francisco: Morgan Kaufmann Publishers.
- Halpin, T. (2002). Part4 Visio-Based Database Modeling in Visual Studio.NET Enterprise Architect .
- Halpin, T. (2005-A). ORM2. In R. Meersman, Z. Tari, & P. Herrero (Eds.), *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops* (Vol. 3762, pp. 676-687). Springer Berlin Heidelberg.
- Halpin, T. (2005-B). *ORM 2 Graphical Notation - Technical Report ORM2-01*. Neumont University.
- Halpin, T. (2006). Object-Role Modeling (ORM/NIAM). In P. Bernus, K. Mertins, & G. Schmidt (Eds.), *Handbook on Architectures of Information Systems* (pp. 81-101). Springer Berlin Heidelberg.
- Halpin, T. (2007). Fact-Oriented Modeling: Past, Present and Future. In J. Krogstie, A. L. Opdahl, & S. Brinkkemper (Eds.), *Conceptual Modelling in Information Systems Engineering* (pp. 19-38). Berlin, Heidelberg, New York: Springer.
- Halpin, T. (Ed.). (2015). *Object Role Modeling: The Official Site for Conceptual Data Model*. Retrieved from ORM: <http://www.orm.net/>
- Halpin, T., & Morgan, T. (2008). *Information Modeling and Relational Databases* (Second ed.). Burlington: Morgan Kaufmann Publishers.
- Halpin, T., & Orlowska, M. (1992). Fact-oriented modelling for data analysis. *Information Systems Journal*, 2(2), 97-119.
- Halpin, T., & Proper, H. (1995). Database Schema Transformation & Optimization. *OOER'95: 14th International Conference on Conceptual Modeling*. 1021,, pp. 191-203. Springer.
- Halpin, T., Evans, K., Hallock, P., & Maclean, B. (2003). *Database Modeling with Microsoft Visio for Enterprise Architects*. (H. Lothlorien, Ed.) USA: Elsevier.
- Hay, D. C. (1999). *A Comparison of Data Modeling Techniques*. Essential Strategies Inc.
- Hinkelmann, K. (2011). *Modeling and Meta-Modeling*. Retrieved 2015, from <http://knut.hinkelmann.ch/>:
http://knut.hinkelmann.ch/lectures/EA2011/EA_2_Modeling_Meta-Modeling.pdf
- Hinkelmann, K. (2015, September). *Meta-Modeling and Modeling Languages*. Retrieved 2015, from <http://knut.hinkelmann.ch/>:
http://knut.hinkelmann.ch/lectures/EA2014-15/EA_4_Metamodeling.pdf

- Hoffer, J. A., Prescott, M. B., & McFadden, F. R. (2007). *Modern Database Management* (Eight ed.). (B. Horan, Ed.) New Jersey: Pearson Prentice Hall.
- Hofstede, A., & Weide, T. (1993). Expressiveness in conceptual data modelling. *Data & Knowledge Engineering*, 10(1), 65-100.
- Holman, U. (2006, February). *A Business-Oriented Foundation for Service Orientation*. Retrieved May 2015, from Microsoft Corporation MSDN Library:
<https://msdn.microsoft.com/en-us/library/aa479368.aspx>
- Jackson, M. (1996). *Academia.edu*. Retrieved May 2015, from
http://www.academia.edu/168883/Object_Role_Modelling_and_Conceptual_Database_Design
- Karagiannis, D., & Kühn, H. (2002). Metamodeling Platforms. In *E-Commerce and Web Technologies* (Vol. 2455, pp. 182-182). Springer.
- Keller, W. (2009). *Using Capabilities in Enterprise Architecture Management*. Retrieved May 2015, from Objectarchitects:
<http://www.objectarchitects.biz/ResourcesDontDelete/CapabilityBasedEAMWhitepaper.pdf>
- Kogalovsky, M., & Kalinichenko, L. (2009, September). Conceptual and ontological modeling in information systems. *Programming and Computer Software*, 35(5), 241-256.
- Lee, T. Y. (1999). Information Modeling from Design to Implementation. *Proceedings of the Second World Manufacturing Congress* (pp. 315-321). Gaithersburg: CiteSeerX.
- Leppanen, M. (2006). An Integrated Framework for Meta Modeling. In Y. Manolopoulos, J. Pokorný, & T. K. Se (Ed.), *10th East European Conference, ADBIS 2006*. 4152, pp. 141-154. Thessaloniki: Springer.
- Long, C., & Vickers-Koch, M. (1995). Using Core Capabilities to Create Competitive Advance. 24(1), 7-22.
- Loucopoulos, P., Bravos, G., Stratigaki, C., & Vavlis, D. (2013). *Deliverable 3.1: Capability Models for Business Compliance Controlling and Auditing (CaaS-Collaborating Project Number 611351)*. Athens.
- Mamayev, R. (2013). *Data Modeling of Financial Derivatives - A Conceptual Approach* (APress ed.). Springer.
- Mandel, T. (2002). Interface Design and Development. In *User/System Interface Design*. Proof.
- Mayer, R. J., Painter, M. K., & deWitte, P. S. (1992). *IDEF Family of Methods for Concurrent Engineering and Business Reengineering Applications*. Knowledge Based Systems Inc.
- Merson, P. (2009). *Data Model as an Architectural View*. Software Engineering Institute .

- Montali, M. (2011-2012). *Conceptual Modelling for Information Systems-Relational Mapping*. Retrieved 02 27, 2015, from Faculty of Computer Science Trilingual and Intercultural-KRDB Research Centre-Faculty of Computer Science-Free University of Bozen-Bolzano: <http://www.inf.unibz.it/~montali/1213/cmris/slides/4.relational-mapping.pdf>
- Mylopoulos, J. (1992). *Conceptual Modelling and Telos*. Retrieved February 2015, from University of Toronto: <http://www.cs.toronto.edu/~jm/2507S/Readings/CM+Telos.pdf>
- Mylopoulos, J. (1998, May). Information modeling in the time of the revolution. *Journal Information Systems*, 23(3-4), 127-155.
- National Institute of Standards & Technology. (1993). *Integration Definition For Information Modeling - IDEF1X*. Federal Information Processing Standards Publications.
- Nijssen, G., & Leunc, C. (1988). Relational Database System Design using the NIAM conceptual Schema. *Information Systems*, 13(2), 219-227.
- Nussbaum, M. S. (2000). *Women and Human Development: The Capability approach*. New York: Cambridge University Press.
- Oracle Corporation. (April 2011). *PL/SQL Developer 9.0 User's Guide*. Allround Automations.
- Oren, T., & Çetin, S. (1999, January 12-14). Quality Criteria for User/System Interfaces. *RTO Meeting Proceedings 38 – Modelling and Analysis of Command and Control*, 18-1, 18-8.
- Phahalad, C. K., & Hamel, G. (1990). The Core Competence of a Corporation. *Harvard Business Review*, 68(3), 79-91.
- Ponniah, P. (2007). *Data Modeling Fundamentals: A Practical Guide for IT Professionals*. New Jersey, Canada: A John Wiley & Sons, INC. Publications.
- Rapakrishnan, R., & Gehrke, J. (2003). *Database Management systems*. USA: McGraw-Hill.
- Rasdorf, W. J., & Abudayyeh, O. Y. (1992). NIAM Conceptual Data-Base Design in Construction Management. *Journal of Computing in Civil Engineering*, 6(1), 41-62.
- Rosen, M. (2010). *Business Processes Start with Capabilities*. Retrieved May 2015, from BPT trends column: <http://www.bptrends.com/publicationfiles/12-07-10-COL-BPM%20%26%20SOA--BusProcesses%20begin%20with%20Capabilities%201003%20v01--Rosen.pdf>
- Rosen, M. (2012). *Processes, Value Streams, and Capabilities*. Retrieved May <http://www.bptrends.com/publicationfiles/12-04-2012-COL-BA-ProcessesValueStreams%26Capabilities-Rosen.pdf>, 2015, from BP Trends Column.
- Roussopoulos, N., & Karagiannis, D. (2009). Conceptual Modeling: Past, Present and the Continuum of the Future. In A. T. Borgida, V. K. Chaudhri, P. Giorgini, & E. S. Yu

- (Eds.), *Conceptual Modeling: Foundations and Applications* (pp. 139-152). Berlin, Heidelberg: Springer.
- Rumbaugh, J., Jacobson, I., & Booch, G. (1999). *The Unified Modeling Language Reference Manual*. Massachusetts, Harlow, England, Menlo Park, California, Berkeley, Don Mills, Ontario, Sydney, Bonn, Amsterdam, Tokyo, Mexico City: Addison Wesley Longman Inc.
- Scott, J. (2009). Business Capability Maps: The Missing Link Between Business Strategy and IT Action . *Architecture & Governance Magazine*, 5(9), pp. 1-4.
- Sharron, A., & Evan, T. (2005). *Beginning Relational Data Modeling* (Second ed.). (T. Davis, Ed.) Springer.
- Shoval, P., & Zohn, S. (1991). Binary-Relationship Integration Methodology. *Data & Knowledge Engineering*, 6(3), 225-250.
- Sommerville, I. (2007). *Software Engineering* (8th Edition ed.). Addison-Wesley.
- Sprinkle, J., Rumpe, B., & Vangheluwe, H. (2010). 3 Metamodelling: State of the Art and Research Challenges. In *Model-Based Engineering of Embedded Real-Time Systems* (Vol. 61100, pp. 57-76). Springer Berlin Heidelberg.
- Stalk, G., Evans, P., & Shulman, L. E. (1992). Competing on Capabilities: The New Rules of Corporate Strategy. *Harvard Business Review*, 70(2), 57-69.
- Stirna, J., Grabis, J., Henkel, M., & Zdravkovic, J. (2012). Capability Driven Development – An Approach to Support Evolving Organizations. *The Practice of Enterprise Modeling. Lecture Notes in Business Information Processing*. 134, pp. 117-131. Springer.
- Stratigaki, C., Loucopoulos, P., & Nikolaidou, M. (2014). Designing a Meta Model as the Foundation for Compliance Capability. *The 1st International Workshop on Capability-oriented Business Informatics (CoBI 2014)*. CEUR Workshop Proceedings (CEUR-WS.org).
- Tallman, S., & Fladmoe-Lindquist, K. (2002). Internationalization, Globalization, and Capability-Based Strategy. *California Management Review*, 45(1), 116-135.
- Teece, D. J., Pisano, G., & Shuen, A. (1997). Dynamic Capabilities and Strategic Management. *Strategic Management Journal*, 18(7), 509-533.
- Tell, A. W. (2014). What Capability Is Not. *Perspectives in Business Informatics Research. Lecture Notes in Business Information Processing*, 194, pp. 128-142. Springer.
- Teorey, T. J., Yang, D., & Fry, J. P. (1986, June). A logical design methodology for relational databases using the extended entity-relationship model. *Journal ACM Computing Surveys (CSUR)*, 18(2), 197-122.
- Teorey, T., Lightstone, S., & Nadeau, T. (2006). *Database Modeling & Design: Logical Design* (Fourth ed.). (J. Gray, Ed.) San Francisco: Morgan Kaufmann, Elsevier.

- Thalheim, B. (1991). Foundations of Entity-Relationship Modeling. *Computer Science Department, University of Rostock*.
- Thalheim, B. (2011). The Theory of Conceptual Models, the Theory of Conceptual Modelling and Foundations of Conceptual Modelling. In D. W. Embley , & B. Thalheim (Eds.), *Handbook of Conceptual Modeling* (pp. 543-577). Heidelberg, Dordrecht, London, New York: Springer.
- Tickoo, S., & Raina, S. (2010). *Oracle 11g with PL/SQL Approach*. Dorling Kindersley Pvt Ltd.
- Tryfona, N., Busborg, F., & Borch, J. G. (1999). starER: A Conceptual Model for Data Warehouse Design.
- Ulrich, W., & Rosen, M. (2011). *The Business Capability Map: The “Rosetta Stone” of Business/IT Alignment*. Retrieved May 2015, from Cutter Consortium: <http://www.cutter.com/content-and-analysis/resource-centers/enterprise-architecture/sample-our-research/ear1102.html>
- Vaughan , M. (2011). A Focused Approach to Business Capability. *The 1st International Symposium on Business Modeling and Software Design*. Sofia, Bulgaria: Informatics Research Centre, Henley Business School, University of Reading, Whiteknights, Reading, UK.
- Wambler, S. (2015). *Data Modeling 101*. Retrieved May 2015, from Agile Data: <http://www.agiledata.org/essays/dataModeling101.html>
- Wernerfelt, B. (1984). A Resource-Based View of the Firm. *Strategic Management Journal*, 5, 171-180.
- Wikipedia: The Free Encyclopedia*. (n.d.). Retrieved May 2015, from Information Technology: https://en.wikipedia.org/wiki/Information_technology
- Wikipedia: The Free Encyclopedia*. (n.d.). Retrieved from https://en.wikipedia.org/wiki/Information_technology#cite_note-2
- Windows Enterprise Support Database Services. (2015). *Liberty University*. Retrieved May 2015, from <http://www.liberty.edu/media/1414/%5B6330%5DERDDataModeling.pdf>
- Wintraecken, J. (1990). *The NIAM Information Analysis Method: Theory and Practice*. Netherlands: Kluwer Academic Publishers.
- www.db-engines.com*. (n.d.). Retrieved from <http://db-engines.com/en/ranking/relational+dbms>
- www.onestopsoftwaretesting.com*. (n.d.). Retrieved from Tutorial Database Testing using SQL: <http://www.softwaretestingtimes.com/2010/04/tutorial-database-testing-using-sql-sql.html>

- www.softwaretestinghelp.com*. (n.d.). Retrieved 2016, from Software Testing Help:
<http://www.softwaretestinghelp.com/database-testing-process/>
- www.softwaretestinghelp.com*. (n.d.). Retrieved 2016, from Software Testing Help:
<http://www.softwaretestinghelp.com/database-testing-process/>
- www.tutorialspoint.com*. (n.d.). Retrieved 2016, from Database Testing Tutorial:
http://www.tutorialspoint.com/database_testing/
- www.zentut.com*. (n.d.). Retrieved from <http://www.zentut.com/sql-tutorial/>
- Zdravkovic, J., Pastor, O., & Loucopoulos, P. (2014). On the Capability Notion in Business Informatics. *The 1st International Workshop on Capability-oriented Business Informatics (CoBI 2014)*. CEUR Workshop Proceedings (CEUR-WS.org).
- Zdravkovic, J., Stirna, J., & Henkel, M. (2013). Modeling Business Capabilities and Context Dependent Delivery by Cloud Services. *Advanced Information Systems Engineering - Lecture Notes in Computer Science, 7908*, 369-383.
- Zhang, Y., Sreedharan, S., & Kambhampati, S. (2015, May 4-8). Capability Models and Their Applications in Planning. *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, 1151-1159.
- Σκουρλάς, Χ. (2001). *Υλοποίηση Εφαρμογών με Γλώσσα Σ:Λ* (1η Έκδοση ed.). Αθήνα: Εκδόσεις Νέων Τεχνολογιών.

Tutorials from the web:

- Tutorial 1: Creating a basic form with Oracle Form Builder 10G:
<https://www.youtube.com/watch?v=hs8jpRklez4>
- Tutorial 3: Creating a simple master-detail form:
<https://www.youtube.com/watch?v=emKtt6TRJrg>
- Tutorial 4: LOV and LOV buttons:
https://www.youtube.com/watch?v=ebEo_dx0imA#t=2.733446
- Tutorial 5: Alerts:
https://www.youtube.com/watch?v=4nw_Bmqdu7A&ebc=ANyPxKo18Nkmo2TplQFkdVp95O7jTVEKMpPeFnQ5ovce-YWQsL4I0DLQZIpYiIT0w6U9UAH8n1gtAXYuyDiBv_NcVXO_tdYSjmg
- Tutorial 6: Item Validation and Advanced Controls:
https://www.youtube.com/watch?v=VndMv1VrwoQ&ebc=ANyPxKo18Nkmo2TplQFkdVp95O7jTVEKMpPeFnQ5ovce-YWQsL4I0DLQZIpYiIT0w6U9UAH8n1gtAXYuyDiBv_NcVXO_tdYSjmg
- Forms 6i tutorial en espanol: <https://www.youtube.com/watch?v=qGhMPT4XZqs>
- Oracle Forms lesson 2:
<https://www.youtube.com/watch?v=Fum3MJm5yKU&list=PL26C3B8E96CFC7AC1>

- Oracle Forms Lesson 4 (create LOV):
<https://www.youtube.com/watch?v=bVSYAK2kyJk&list=PL26C3B8E96CFC7AC1#t=8.655532>
- Oracle Forms Lesson 5 (create LOV manual):
https://www.youtube.com/watch?v=oYkwm_40-n0&index=2&list=PL26C3B8E96CFC7AC1#t=1.306041
- Oracle Forms Lesson 6 part 1 (working with Alerts forms):
<https://www.youtube.com/watch?v=YgBPEVHwhKQ&index=2&list=PL26C3B8E96CFC7AC1#t=2.121625>
- Oracle Forms Lesson 6 part 2 (Working with Alerts forms):
<https://www.youtube.com/watch?v=6CTTIRSV00w&index=2&list=PL26C3B8E96CFC7AC1#t=28.340479>
- Oracle Forms Lesson 6 part 3 (Working with Alerts forms):
https://www.youtube.com/watch?v=vDEGrqu_loc&index=2&list=PL26C3B8E96CFC7AC1#t=28.340479
- Oracle Forms Lesson 7 part 1(Working with Multiple Forms):
<https://www.youtube.com/watch?v=tFYnBnKlqFw&index=2&list=PL26C3B8E96CFC7AC1#t=0.21125>
- Oracle Forms Lesson 7 part 2(Working with Multiple Forms):
<https://www.youtube.com/watch?v=3UEX2Wu2pN4&index=2&list=PL26C3B8E96CFC7AC1#t=0.21125>
- Oracle Forms Lesson 9 (Working with Parameters):
<https://www.youtube.com/watch?v=KaUjwAWsidA&index=2&list=PL26C3B8E96CFC7AC1#t=1.940083>
- Oracle part 3 Visual attribute:
<https://www.youtube.com/watch?v=yphPXBZxaw0&index=2&list=PL26C3B8E96CFC7AC1#t=0.40977>
- Oracle part 4 Lovs and Record Group By Wizard:
https://www.youtube.com/watch?v=J76z-_naFfI&list=PL26C3B8E96CFC7AC1&index=12
- Oracle part 5 Manual Lov and Record Group:
<https://www.youtube.com/watch?v=ZLyTXoSJc2c&index=13&list=PL26C3B8E96CFC7AC1>
- Oracle part 6 H and V menu bars:
https://www.youtube.com/watch?v=_Bzb63VWuW8&index=14&list=PL26C3B8E96CFC7AC1
- Oracle part 7 Stack Canvas:
<https://www.youtube.com/watch?v=kp9ldF6Mr78&index=15&list=PL26C3B8E96CFC7AC1>
- Oracle part 8 Tab Canvas:
<https://www.youtube.com/watch?v=eAR1ptuWGXm&index=16&list=PL26C3B8E96CFC7AC1>
- Oracle Forms 11g - When-Validate-Item Trigger:
<https://www.youtube.com/watch?v=zm-xRE5hCW4>

- Oracle Developer - Lecture 11 - Ustad Ahmad Shah:
<https://www.youtube.com/watch?v=KFIWh8ywMW4&list=PLjIWxiqmfBg1y0-MfsleKFUfAcnS3Ve41>
- Oracle Developer - Lecture 10 - Ustad Ahmad Shah – Pashto:
https://www.youtube.com/watch?v=j5_W7SffLoA&index=2&list=PLjIWxiqmfBg1y0-MfsleKFUfAcnS3Ve41
- Scope of triggers:
https://www.youtube.com/results?search_query=Oracle+Forms+11g
- Forms Builder / Insert:
https://www.youtube.com/watch?v=cZ0QZCJ0wcE&ebc=ANyPxKpV17npjCse0ITE25gR5C8ovuE1b03NxVvOdLxr8PkX964P8cn5L-1bhq_jhMNDKQtl_8YcSGtkD8Rnkk6kE7E0Stq8A&nohtml5=False
- Using Sequence to generate automated data:
<https://www.youtube.com/watch?v=NQ38hIPoSg4>
- Tutorial 2 - Using radio buttons:
https://www.youtube.com/watch?v=X5L3rHU5TBY&annotation_id=annotation_616781&src_vid=hs8jpRklez4&feature=iv
- Oracle Forms Training: Master Details with Tab Canvas:
<https://www.youtube.com/watch?v=ZmLRneqP51M>
- How to convert content canvases item to tab canvas:
<https://www.youtube.com/watch?v=KHqtbplh7c0>
- Oracle Forms 10g: How to enable/disable buttons or item with check box?:
<https://www.youtube.com/watch?v=DrUcSGypSdE>
- Oracle Forms 10g: How to create iconic button or animated button:
<https://www.youtube.com/watch?v=3w-PinuEKgw&nohtml5=False>
- Forms 6 Triggers When Button Pressed Trigger example:
<https://www.youtube.com/watch?v=XG4yAee2ed8>
- Oracle Forms Lesson 8 Part 2(search/Query Form):
<https://www.youtube.com/watch?v=kCkpBnXuOMI&ebc=ANyPxKp9zidP02MXgJWpHx4WfSGj1TwH82ChqMQYs8TzVm4WhfFOWFLRmDWF1Ty4bMJHD4g5MZ0MVNynFbjlwriL8jeFexyr0A&nohtml5=False>
- Oracle forms lesson 12: <https://www.youtube.com/watch?v=w4aSoNlrzU>
- Login Screen: <https://www.youtube.com/watch?v=mj1x5y1Qt-I>
- Create Login Forms Oracle 10g by Muhammad Nur E Alam from Bangladesh with Bangla: <https://www.youtube.com/watch?v=LPel6Gk0YZs>
- Part 19 Change Password: <https://www.youtube.com/watch?v=7j6kxvTqjig>
- Part 18 login validation type 1: <https://www.youtube.com/watch?v=IJKPiMcJUAQ>
- Hide Default Icon from Forms Runtime:
<http://oracledevelopersuite.blogspot.gr/2010/03/hide-default-icon-from-forms-runtime.html>
- Disable menu item of login form: <http://oracle.ittoolbox.com/groups/technical-functional/oracle-dev-l/how-to-enable-or-disable-the-default-menu-items-in-forms-6i-468143>
- How to create the Hierarchical Tree in oracle forms?:
<http://startapps.blogspot.gr/2009/12/how-to-create-hierarchical-tree-in.html>

- Create Tree:
<http://www.orafaq.com/forum/t/198850/>
http://www.orafaq.com/forum/m/608702/?srch=tree#msg_608702
http://www.orafaq.com/forum/m/604341/?srch=tree+call+form#msg_604341
http://www.orafaq.com/forum/m/416642/?srch=tree+call+form#msg_416642
- Oracle Lesson 17 (Open form & continuous changing of time on display item):
<https://www.youtube.com/watch?v=JErnZ00giww>