

CZECH UNIVERSITY OF LIFE SCIENCES

FACULTY OF ECONOMICS AND MANAGEMENT

DEPARTMENT OF INFORMATION ENGINEERING



DISSERTATION THESIS

***EFFICIENT BUSINESS PROCESS REQUIREMENT ANALYSIS  
BASED ON THE TRANSITION FROM THE UML USE CASE  
METHOD TO THE BUSINESS OBJECT RELATION MODELING  
APPROACH: THE CASE STUDY OF CRITICAL GREENHOUSE  
INTEGRATED PEST MANAGEMENT BUSINESS PROCESSES***

**Author:** Ing. Athanasios Podaras

**Supervisor:** Prof. RNDr. Jiří Vaníček, CSc.

**MARCH 2010**

**PRAGUE**

*I would like to thank my supervisor Prof. RNDr. Jiří Vaniček, CSc. for his invaluable help throughout the entire period of my PhD study, and also for the derivation of the current dissertation thesis.*

*I would also like to thank, Prof. Ing. Ivan Vrana, DrSc., and Doc. Ing. Vojtech Merunka, PhD. for their generous contribution in terms of the completion of the current dissertation thesis, and their support throughout the entire period of my PhD Study. For the above stated reasons, I would also like to thank the Department of Information Engineering of the Faculty of Economics and Management in Prague.*

*Many thanks to Prof. Alexander Sideridis, head of the Informatics Laboratory of the Agricultural University of Athens in Greece.*

*Athanasios Podaras*

*30 March 2010*

## STATEMENT

I declare, that for the completion of the current dissertation thesis with the topic

*“ EFFICIENT BUSINESS PROCESS REQUIREMENT ANALYSIS BASED ON THE  
TRANSITION FROM THE UML USE CASE METHOD TO THE BUSINESS OBJECT  
RELATION MODELING APPROACH: THE CASE STUDY OF CRITICAL  
GREENHOUSE INTEGRATED PEST MANAGEMENT BUSINESS PROCESSES’*

i worked by myself, under the supervision of Prof. RNDr. Jiří Vaniček, CSc.

The utilized literature and also the related tools and methods are listed in the end of  
the PhD Thesis document.

Prague, 30 March 2010

Ing. Athanasios Podaras

# CONTENTS

<b>LIST OF FIGURES .....</b>	<b>7</b>
<b>LIST OF TABLES .....</b>	<b>8</b>
<b>1 INTRODUCTION – GOAL.....</b>	<b>10</b>
<b>2 INFORMATION TECHNOLOGY DEVELOPED APPLICATIONS FOR SOLVING INTEGRATED PEST MANAGEMENT (IPM) PROBLEMS. ....</b>	<b>18</b>
<b>2.1 NEST: A new expert system tool and its application to Pest Management.....</b>	<b>19</b>
2.1.1 Problems regarding computer based IPM .....	20
2.1.2 System’s characteristics.....	20
2.1.3 Implementation.....	23
<b>2.2 Information Technology for Creation and Use of the PC “Phytopharmacy” Database..</b>	<b>23</b>
2.2.1 Components of the proposed Information technology .....	24
2.2.2 Conceptual Model of the PC Database.....	25
<b>3 REQUIREMENT ANALYSIS AND PROJECT FEASIBILITY ROLE TO SYSTEM INTEGRATION PROCEDURE.....</b>	<b>27</b>
<b>3.1 Defining, validating and prioritizing initial requirements.....</b>	<b>27</b>
3.1.1 Defining requirements .....	27
3.1.2 Documenting requirements .....	28
3.1.3 Validating requirements .....	29
3.1.4 Prioritizing requirements .....	30
<b>3.2 Project feasibility.....</b>	<b>30</b>
3.2.1 Economic, technical and operational feasibility .....	30
3.2.2 Identification of risks.....	31
3.2.3 Identification of defects .....	32
<b>4 OBJECT-ORIENTED TOOLS FOR PERFORMING EFFICIENT AND DETAILED <i>BUSINESS PROCESS REQUIREMENT ANALYSIS</i>.....</b>	<b>33</b>
<b>4.1 Business Process Requirement Analysis with the Use Case Concept .....</b>	<b>33</b>
4.1.1 Use Case Analysis Parts .....	34
4.1.2 Use Case Scenarios .....	35
4.1.3 Use Case Diagrams .....	37
<b>4.2 Business Object Relation Modeling (BORM) .....</b>	<b>37</b>
4.2.1 Characteristics of the BORM approach.....	39
4.2.2 The advantages of BORM methodology to requirement analysis .....	41

<b>4.3</b>	<b>The “Use Case To BORM Transformation Algorithm” (UCBTA) .....</b>	<b>46</b>
4.3.1	Parallel comparison of Use Case and BORM Business Process Models .....	47
4.3.2	UCBTA <i>Representation</i> .....	56
<b>4.4</b>	<b>Advantages of the Use Case to BORM Transformation Algorithm regarding business process requirement analysis.....</b>	<b>58</b>
4.4.1	UCBTA: A pattern – oriented methodology .....	58
4.4.2	Mathematically defined approach .....	59
4.4.3	Process feasibility analysis from project commencement .....	60
4.4.4	Based on tested methodologies .....	60
4.4.5	UCBTA Transition Rules eliminate possible loss of data .....	61
<b>4.5</b>	<b>Formal definition of the UCBTA approach to business process requirement analysis....</b>	<b>62</b>
4.5.1	Mapping the Use Case approach to the BORM methodology.....	63
4.5.2	UCBTA Transition Rules.....	66
<b>4.6</b>	<b>Mathematical expression of the UCBTA Algorithm .....</b>	<b>72</b>
4.6.1	General characteristics of the Finite State Machine .....	73
4.6.2	Formal Definition of the Non – Deterministic Finite Automaton .....	74
4.6.3	Formal Mathematical Definition of the UCBTA Non – Deterministic Finite Automaton ..	75
<b>4.7</b>	<b>Business Process Requirement Analysis with the UCBTA_PROJECTS application .....</b>	<b>85</b>
4.7.1	Primary Goal .....	85
4.7.2	Delineation of the UCBTA_PROJECTS interface.....	86
<b>5</b>	<b>APPLIED UCBTA BUSINESS PROCESS REQUIREMENT ANALYSIS – CASE STUDY OF CRITICAL GREENHOUSE INTEGRATED PEST MANAGEMENT (IPM) PROCESSES.....</b>	<b>95</b>
<b>5.1</b>	<b>What is Integrated Pest Management? .....</b>	<b>96</b>
<b>5.2</b>	<b>What does effective Integrated Pest Management entail?.....</b>	<b>97</b>
<b>5.3</b>	<b>UCBTA - Case Study of the Greenhouse Integrated Pest Management practices .....</b>	<b>99</b>
5.3.1	Scouting (Monitoring) Record Keeping .....	100
5.3.2	Evaluation of pesticide’s effectiveness.....	125
<b>6</b>	<b>CONCLUSION .....</b>	<b>134</b>
<b>7</b>	<b>LITERATURE .....</b>	<b>137</b>
<b>7.1</b>	<b>Scientific Books, Journals, Papers .....</b>	<b>137</b>
<b>7.2</b>	<b>Author’s Literature.....</b>	<b>143</b>
<b>7.3</b>	<b>Internet.....</b>	<b>144</b>
<b>8</b>	<b>APPENDIX.....</b>	<b>146</b>

*The current dissertation thesis is devoted to my wife, my parents and my brother. I would like to thank them for their invaluable support and love.*

## LIST OF FIGURES

<b>FIGURE 2:1:</b> STRUCTURE OF NEST KNOWLEDGE BASE FOR DISEASES OF CROPS (MALIAPPIS, SIDERIDIS, MAHAMAN, "NEST: A NEW EXPERT SYSTEM TOOL AND ITS APPLICATION TO PEST MANAGEMENT", PROC. OF THE 3RD EUROPEAN CONFERENCE OF THE EUROPEAN FEDERATION FOR INFORMATION TECHNOLOGY IN AGRICULTURE, FOOD AND ENVIRONMENT (J. STEFFE, ED.), VOL. 2, 2001, PP. 421-426).....	23
<b>FIGURE 2:2:</b> COMPONENTS OF THE DEVELOPED INFORMATION TECHNOLOGY(ONKOV K., DIMOVA D.,"INFORMATION TECHNOLOGY FOR CREATION AND USE OF PC "PHYTOPHARMACY" DATABASE", PROC OF HAICTA 2006 INTERNATIONAL CONFERENCE ON "INFORMATION SYSTEMS IN SUSTAINABLE AGRICULTURE, AGROENVIRONMENT AND FOOD TECHNOLOGY", 20-23 SEPTEMBER 2006, VOLOS, GREECE.....	24
<b>FIGURE 2:3:</b> CONCEPTUAL MODEL OF THE PC PHYTOPHARMACY DATABASE(ONKOV K., DIMOVA D.,"INFORMATION TECHNOLOGY FOR CREATION AND USE OF PC "PHYTOPHARMACY" DATABASE", PROC OF HAICTA 2006 INTERNATIONAL CONFERENCE ON "INFORMATION SYSTEMS IN SUSTAINABLE AGRICULTURE, AGROENVIRONMENT AND FOOD TECHNOLOGY", 20-23 SEPTEMBER 2006, VOLOS, GREECE.....	25
<b>FIGURE 4:1:</b> ACTOR AND USE CASE DEPICTION.....	35
<b>FIGURE 4:2:</b> THE BORM STAGES (KNOTT ET AL., KNOWLEDGE BASED SYSTEMS. ,16, 2003: 77-89) ..	40
<b>FIGURE 4:3:</b> BORM INFORMATION ENGINEERING PROCESS (AGRIC. ECON. – CZECH, 52, 2006 (4): 165-172).....	40
<b>FIGURE 4:4:</b> FIG 9 : PROCESS – PARTICIPANT INTERACTION MODEL EXAMPLE (SOURCE:BORM – BUSINESS OBJECT RELATION MODELING , HTTP://WWW.CSE.MSU.EDU/ICRE2000/MERUNKA/BORM_HTML/INDEX.HTML.....	42
<b>FIGURE 4:5:</b> THE GENERAL SCHEMA OF THE UCBTA ALGORITHM.....	47
<b>FIGURE 4:6:</b> FLOWCHART OF THE UCBTA ALGORITHM.....	57
<b>FIGURE 4:7:</b> MAPPING OF THE USE CASE SCHEMA TO BORM SCHEMA.....	65
<b>FIGURE 4:8:</b> BORM ASPECT OF PROCESS A AFTER PRIMARY UCBTA TRANSITION.....	67
<b>FIGURE 4:9:</b> BORM ASPECT OF PROCESS A AFTER PRIMARY UCBTA TRANSITION.....	69
<b>FIGURE 4:10:</b> BORM ASPECT OF PROCESS B AFTER MIDDLE STEP UCBTA TRANSITION.....	70
<b>FIGURE 4:11:</b> BORM ASPECT OF PROCESS C AFTER MIDDLE STEP CONDITIONAL UCBTA TRANSITION.....	72
<b>FIGURE 4:12:</b> NON – DETERMINISTIC FINITE AUTOMATON SCHEMA OF THE UCBTA ALGORITHM.....	84
<b>FIGURE 4:13:</b> THE UCBTA_PROJECTCS_LOGIN WINDOW.....	87
<b>FIGURE 4:14:</b> THE UCBTA_PROJECTCS WINDOW.....	88
<b>FIGURE 4:15:</b> THE UCBTA_PROJECTCS WINDOW WITH THE PROJECT_ID COMBO BOX.....	89
<b>FIGURE 4:16:</b> THE USE CASE DATA MODEL WINDOW WITH ALL MAIN SUCCESS SCENARIO STEPS INCLUDED.....	90
<b>FIGURE 4:17:</b> USE CASE STEP DETAILS WINDOW WITH SUB STEPS OF EACH USE CASE MAIN SUCCESS SCENARIO STEP.....	92
<b>FIGURE 4:18:</b> THE BORM (PROCESS PARTICIPANT INTERACTION MODEL) WINDOW – BUSINESS PROCESS WORKFLOW IS INCLUDED.....	93
<b>FIGURE 5:1:</b> USE CASE DIAGRAM FOR THE "DAILY RECORD KEEPING FOR SCOUTING PURPOSES" PROCESS.....	108
<b>FIGURE 5:2 :</b> OBJECT RELATION DIAGRAM BASED ON THE DAILY RECORD KEEPING FOR MONITORING PROCESS.....	110
<b>FIGURE 5:3:</b> ENTERING USERNAME AND PASSWORD TO THE LOGIN WINDOW.....	111
<b>FIGURE 5:4:</b> ENTERING INITIAL DATA TO THE UCBTA PROJECTS WINDOW – DAILY SCOUTING AND RECORD KEEPING PROCESS.....	112
<b>FIGURE 5:5:</b> ENTERING MAIN SUCCESS SCENARIO DATA TO THE USE CASE DATA MODEL WINDOW – DAILY SCOUTING AND RECORD KEEPING PROCESS.....	113
<b>FIGURE 5:6:</b> BORM (PROCESS PARTICIPANT INTERACTION MODEL) DATA AUTOMATICALLY DERIVED – DAILY SCOUTING AND RECORD KEEPING PROCESS.....	114
<b>FIGURE 5:7:</b> USE CASE DIAGRAM FOR THE "SCOUTING ANALYSIS BASED ON WEEKLY RECORD KEEPING" PROCESS.....	121
<b>FIGURE 5:8:</b> ORD DIAGRAM FOR THE "SCOUTING BASED ON WEEKLY RECORD KEEPING" PROCESS ...	122
<b>FIGURE 5:9:</b> ENTERING INITIAL DATA TO THE UCBTA PROJECTS WINDOW – WEEKLY SCOUTING AND RECORD KEEPING PROCESS.....	123

<b>FIGURE 5:10:</b> ENTERING MAIN SUCCESS SCENARIO DATA TO THE USE CASE DATA MODEL WINDOW – WEEKLY SCOUTING AND RECORD KEEPING PROCESS .....	123
<b>FIGURE 5:11:</b> BORM (PROCESS PARTICIPANT INTERACTION MODEL) DATA AUTOMATICALLY DERIVED – WEEKLY SCOUTING AND RECORD KEEPING PROCESS .....	124
<b>FIGURE 5:12:</b> USE CASE DIAGRAM FOR THE “EVALUATING PESTICIDE EFFECTIVENESS FROM MONTHLY RECORD KEEPING” PROCESS .....	129
<b>FIGURE 5:13:</b> ORD DIAGRAM FOR THE “EVALUATING PESTICIDE EFFECTIVENESS THROUGH MONTHLY RECORD KEEPING” PROCESS.....	130
<b>FIGURE 5:14:</b> ENTERING INITIAL DATA TO THE UCBTA PROJECTS WINDOW – EVALUATION OF PESTICIDE EFFECTIVENESS PROCESS.....	131
<b>FIGURE 5:15:</b> ENTERING MAIN SUCCESS SCENARIO DATA TO THE USE CASE DATA MODEL WINDOW – EVALUATION OF PESTICIDE EFFECTIVENESS PROCESS .....	132
<b>FIGURE 5:16:</b> BORM (PROCESS PARTICIPANT INTERACTION MODEL) DATA AUTOMATICALLY DERIVED – EVALUATION OF PESTICIDE EFFECTIVENESS PROCESS .....	133

## LIST OF TABLES

<b>TABLE 2:1:</b> DISEASES INCLUDED IN NEST(MALIAPPIS, SIDERIDIS, MAHAMAN, "NEST: A NEW EXPERT SYSTEM TOOL AND ITS APPLICATION TO PEST MANAGEMENT", PROC. OF THE 3RD EUROPEAN CONFERENCE OF THE EUROPEAN FEDERATION FOR INFORMATION TECHNOLOGY IN AGRICULTURE, FOOD AND ENVIRONMENT (J. STEFFE, ED.), VOL. 2, 2001, PP. 421-426) .....	21
<b>TABLE 2:2:</b> INSECTS AND MITES INCLUDED IN NEST(MALIAPPIS, SIDERIDIS, MAHAMAN, "NEST: A NEW EXPERT SYSTEM TOOL AND ITS APPLICATION TO PEST MANAGEMENT", PROC. OF THE 3RD EUROPEAN CONFERENCE OF THE EUROPEAN FEDERATION FOR INFORMATION TECHNOLOGY IN AGRICULTURE, FOOD AND ENVIRONMENT (J. STEFFE, ED.), VOL. 2, 2001, PP. 421-426) .....	22
<b>TABLE 4:1:</b> APPROACHES FOR MODELING BUSINESS LOGIC( KNOTT ET AL., THE ROLE OF OBJECT – ORIENTED PROCESS MODELING IN REQUIREMENTS ENGINEERING PHASE OF INFORMATION SYSTEMS DEVELOPMENT, CONFERENCE PROCEEDINGS EFITA 2003, DEBRECEN, HUNGARY, 2003, P 300 – 307) .....	38
<b>TABLE 4:2:</b> REAL PROJECTS WERE BORM APPROACH TO BPM IS USED (KNOTT ET AL., KNOWLEDGE BASED SYSTEMS. ,16, 2003: 77–89) .....	41





# 1 Introduction – Goal

One of the most critical features of the current business status worldwide is the occurrence of the dramatic alterations in the field of information technology. This fact is obvious to the entire industrial community and consequently all companies strive to be utterly informed of those changes, since their financial development and market position, is strongly related to the technological rapid evolution which occurs nowadays.

What is implied by the author of the current dissertation thesis is that the IT strategy has to be performed with regard to the latest technological achievements and with the support of the best IT experts and consultants. The key factors that according to the author guarantee the most successful IT Strategy are the “*rapid data transfer*” and also the “*rapid business process implementation*”. Each of the above mentioned factors should be always taken into consideration by the system or software developers. According to many IT experts detailed requirement analysis leads to accurate, effective and efficient business process implementation. The author of the current document proposes the *detailed requirement analysis* as the key element of successfully integrated information systems or applications.

Many tools have been developed and proposed by IT experts, as great weapons in order to perform detailed and accurate requirement analysis. Before mentioning the most important proposed tools worldwide as an important part of system and software integration, it has to be stated that analytical discussions between IT experts and the end users of the final application have to be made, so that the analysts will be able to absorb the aimed utilization of the constructed product, insure that user requirements will be met, and that the long lasting bug fixing after user acceptance tests will be avoided. However, product utilization absorbance, from the part of IT analysts and developers is the first step for the integration of a promising application, which is based on analytical and extensive requirement analysis.

The following step is the choice of the proper tools for performing in depth requirement analysis in terms of business processes. Throughout the current

document, two prominent methods of business process requirement analysis are examined; the aforementioned methods are the *Use Case methodology* and also the *BORM methodology (Business Object Relation Modeling)*.

Both of the aforementioned tools, have been used in many real-life projects so far with success; the emergence though of remarkable gaps throughout the implementation of the above mentioned methodologies prompted the author to design and introduce an innovative and pattern based approach to requirement analysis. This new method is based on the author's concept of utilizing both of the aforementioned approaches in order to achieve the ideal business process requirement analysis.

The utilization of the Use Case Model solely can lead to business process requirement analysis mistakes since the business process flow is missing. If the Use Case Model is followed by i.e. the Sequence diagrams of UML the process flow is present but if the end user is not computer oriented then communication failure between the IT specialist and end user could emerge. On the other hand the BORM method is ideal for representing business process flow in a user friendly manner absorbable by users not oriented to programming concepts; but the formal process steps and sub steps as recorded by the Use Case method is missing from BORM. As a consequence the author decided to perform the construction of an algorithm, with which the successful transformation from Use Case model to the BORM model will be achieved, and the gaps that can emerge when not both of the above mentioned methodologies are utilized will be eliminated.

It can be thus stated that:

- the author's main goal is the construction of a new and innovative methodology for implementing efficient business process requirement analysis
- the method is based on two existing and tested approaches: the Use Case Model and the BORM model; the presence of both methods is necessary for

the construction of the new methodology so that the possible requirement analysis gaps caused by the unique presence of one of the two methods will be prevented

- the new approach is based on the transformation of the Use Case Model to the BORM approach
- the constructed by the author algorithm with which detailed, analytical, effective and efficient business process requirement analysis can be performed is defined as *the Use Case To BORM Transformation Algorithm*. Throughout the current document the aforementioned algorithm will be defined as *the UCBTA algorithm*
- the new approach designed by the author, as an algorithm has mathematical background; the UCBTA algorithm is based on the non – deterministic finite automaton theory
- specific Software Application is introduced which supports automatic transformation from Use Case to BORM. The application is entitled as UCBTA\_PROJECTS. The tool is designed in Microsoft Visual Studio 2005 programming environment and developed by Visual Basic 6.0 programming language
- the defined as UCBTA algorithmic approach to requirement analysis is implemented on a scientific Case Study from the field of Agriculture and precisely from the Greenhouse Integrated Pest Management (IPM) business process area. The IPM business process requirement analysis is selected by the author since little or no work has been done so far in this area; examples of IPM work are described throughout the second chapter of the document and the basic conclusion derived is that lack of requirement analysis is the main problem observed in the area of computer based IPM. Of course the author's ambition is the possibility to implement UCBTA business process requirement analysis on more Agricultural activities. The integrated tools are:

*A) NEST: a new expert system tool and its application to Pest Management, Agricultural University of Athens, Informatics Laboratory*

*B) Creation and Use of the PC "Phytopharmacy" Database, Agricultural University of Bulgaria, Department of Computer Science, Plovdiv*

After the definition of the IPM tools defined so far, where the poor business process requirement analysis is admitted as a huge problem for the construction of such applications, the presentation of the existing requirement analysis tools and methodologies comprises of the next important part of the current thesis. The presentation of first the Use Case Method and then the BORM (Business Object Relation Model) is important for the reader to comprehend their requirement analysis gaps – despite their utilization in real-life projects – and also why it their simultaneous utilization is important to cover the named gaps. At that point the author introduces the transition from the Use Case Model to BORM Model as the solution to these gaps; the implementation is performed by the construction of the Use Case To BORM Transformation Algorithm (UCBTA) which is utilized as the bridge that connects both applications for the above stated gap elimination.

The current dissertation thesis also includes a scientific and detailed presentation of the UCBTA algorithm. The above mentioned presentation is comprised of five basic parts which are considered by the author as crucial parts of any algorithmic definition,

The first part of the presentation focuses on the steps of which the UCBTA algorithmic method to business process requirement analysis is comprised. The steps are analyzed throughout the text of the current dissertation and also depicted at an analytical flowchart.

The second part of the algorithm's presentation is, which is considered as the completion of the first algorithmic part is its mathematical definition. The mathematical background of the UCBTA construction is the Non – Deterministic Finite State Automaton. Many scientists are used to the mentioned mathematical methodology as a method to define computer processes on a machine level. The author's concept is to utilize this mathematical approach on a business level and depict with specified mathematical symbols the algorithm's mathematical substance.

The third part of the algorithm's presentation is comprised of the advantages that the UCBTA approach to business process requirement analysis. Throughout this part it is explained why the defined algorithm is considered to be a scientific upgrade to this part of the information system development. Practical main profit according to the author of the current dissertation thesis is that analysts can avoid time loss to thorough

requirement analysis if they follow the defined UCBTA steps, which is a pattern oriented approach.

The fourth part of the UCBTA delineation includes the official transition rules according to which the Use Case Model is transformed to the BORM Model. The Use Case Main Success scenario and the sub steps are mapped to states, activities, flows and communications which are the basic elements of the BORM approach. The schematic depiction of the Use Case model in BORM is performed via Object Relation Diagrams. The defined transition rules are utilized by the author so that possible loss of data during the transition will be avoided.

Apart from the UCBTA Transition Rules which guarantee a secure transition from the Use Case Model to BORM without data loss, a windows-based and user friendly application is constructed for the support of the above described transition. The software is designed and developed by the author and it is named as UCBTA\_PROJECTS.

The fifth and final part of the UCBTA algorithm presentation is its application on a specific Case Study. Due to his agricultural and IT background, the author of the current thesis will implement the current algorithm on specific Greenhouse Integrated Pest management business processes. Detailed requirement analysis based on the analyzed algorithm is performed throughout the final chapter of the thesis.

The UCBTA approach to business process requirement analysis is presented as an Object – Oriented methodology since it is based on the Use Case and BORM, who both stem from the object concept. The reason for selecting an object approach to implement Greenhouse IPM business process requirement analysis is that the Agricultural scientific field includes data whose nature is by itself object – oriented. Taxonomies of plants, insects, diseases and pests are identical to object – oriented notations such as classes, objects, subclasses and data sets, due to their hierarchical data character.

*The primary goal of the current document is the delineation of a new, modern and detailed object oriented business process requirement analysis methodology, based on the transformation algorithm from Use Case Model to the BORM Model, and also the proposal of a Case Study requirement analysis, related to the pattern based amelioration of specific Greenhouse Integrated Pest Management processes.*

*The following thesis summary demonstrates a short business process requirement state of the art modeling tools, IPM state of the art applications that have been so far integrated, solutions that can be provided for requirement analysis improvement by using scientific tools and methodologies and finally an IPM case study of how the requirement analysis amelioration will positively affect Agriculture processes:*

**Goal:** Improvement of business process requirement analysis procedure by using a new and innovative Object - Oriented Methodology.

**Case Study area:** Agriculture: Computer based Greenhouse Integrated Pest Management (IPM) business process improvement by using appropriate Software.

**Existing Computer Based Greenhouse IPM tools:**

- NEST: A new expert system tool and its application to Pest Management, Agricultural University of Athens, Informatics Laboratory
- Creation and Use of the PC “Phytopharmacy” Database, Agricultural University of Bulgaria, Department of Computer Science, Plovdiv.

**Most famous existing Object - Oriented business process requirement analysis tools and methods (both tested in real – life projects):**

- Use Case approach
- BORM Approach

**Gaps of existing methods:**

Use Case Method:

- 1) Business Process Workflow is absent
- 2) It is followed by Sequence diagrams for workflow depiction but is too focused on programming concepts and cannot be absorbed by non IT experts.

BORM Method:

Business Process Workflow is present but the workflow activities are not supported by a formal and tested method such as Use Case.

**Gap Covering Solution:**

- Utilization of both Methods when implementing Business process requirement analysis
- Transformation of the Use Case Model to BORM Model with the Use Case To BORM Transformation Algorithm (UCBTA)



### **UCBTA Algorithm characteristics and advantages:**

- 1) Pattern oriented solution
- 2) Based on two tested methodologies (Use Case and BORM)
- 3) Has mathematical background (non-deterministic finite automaton)
- 4) Process feasibility analysis from project commencement
- 5) UCBTA transition rules prevent loss of data after transformation is completed
- 6) A specific Software Application is introduced which supports automatic transformation from Use Case to BORM. The application is entitled as UCBTA\_PROJECTS

### **UCBTA Case Study – IPM Business Process Requirement Analysis:**

Three important IPM business processes are selected in order to implement the UCBTA approach to business process requirement analysis.

## **2 Information technology developed applications for solving Integrated Pest Management (IPM) problems**

The rapid emergence of the information technologies in the majority of the scientific fields, forces the scientists who belong the rest of those fields, where new technologies have not still been introduced, to co-operate with IT experts in order to integrate software applications or to design information systems from scratch, in order to perform efficient practices. Efficient practices require the rapid and cost effective possible operation of them. It has also been concluded, that there are fields of science, for which many research works with regard to information technology has been done, but no practical implementations so far exist.

A remarkable scientific effort for the development of the System Integration with regard to the effective automatic operation of many agricultural activities is a fact that it can not be ignored. Agricultural experts are witnesses of the application of effective automated approaches to precision agriculture and farming; characteristic example that can be mentioned is the presence of intelligent machines and robots utilized to collect fruits, or automatic Greenhouse water supply programmed systems. Experiments have proved that intelligent system utilization in the above mentioned circumstances dramatically increased production.

The utilization of G.I.S. applications in forestry and agricultural production, comprise of another remarkable reference to the effective and efficient technological effort in the field of agriculture. In forestry, the basic problem on which scientists still focus, is the provision of a drastic solution regarding the prevention of future the disastrous fires as those that emerged in recent years worldwide, and played an important part to the global environmental disaster. In general, a number of famous Universities worldwide strive to perform scientific research with emphasis to the environmental amelioration.

An agricultural field though, for which computer science has still a lot to offer, and surprisingly enough poor information technology research has been performed is the Greenhouse Integrated Pest Management (IPM). Despite the fact that a lot of

Greenhouse processes are automatically implemented via expert systems, Pest Management lacks automation in many aspects. Monitoring processes as far as pest control practices in many cases are implemented without even the utilization of applications such as excel worksheets to keep track of the implemented pesticides, not to mention the existence of a database for implementing the above mentioned greenhouse activity.

The concrete part of the present document refers to scientific so far implemented effort, by IT experts for the construction of automated systems, designed and integrated in terms of Integrated Pest Management activities. It has to be stated that in all cases, despite the success of the application or the system, experts underline the emergence of serious problems due to the limited communication between IT experts and end users. Moreover, the need for the detailed business process requirement analysis related to IPM is obvious and it stems from the fact that many theoretical work has been done so far, but few practical integrated applications have appeared so far.

### ***2.1 NEST: A new expert system tool and its application to Pest Management***

The delineated application was integrated by the Agricultural University of Athens in Greece, and the Informatics Laboratory of the University. The people involved in the creation of the above NEST tool are Prof. Sideridis A. and his research assistants Dr. Maliapis M. and Dr. Mahaman B. The constructed system is rule-based. The modular construction of the Knowledge base (KB) is used by the system; the structure of the Knowledge base was developed for pest and disease management actions and for three greenhouse crops, namely *tomato*, *aubergine*, and *pepper*. The reason for the selection of the concrete crops is that a common core of knowledge can be identified in their cultivation process.

### **2.1.1 Problems regarding computer based IPM**

The rational Integrated Pest Management (IPM) in Greenhouses requires the elimination of losses that are caused by the fast reproduction rate of the pests. The high quality services offered by computer technology nowadays are strongly and agricultural experts with regard to pest control practices.

*Even though a lot of research work has been done in the field of IPM, and even if a lot of knowledge has been acquired, practical implementation of IPM research results is far to be completed. One of the reasons is the information gap between research scientists, extension agents and farmers. This lack of information leads to the misunderstanding of the Pest status, non-identification of the Pest and non-accuracy regarding recommended measures of control [52].*

The above stated *information gap* stems from poor analysis of functional requirements since they capture the intended behavior of the system [88]. For the above mentioned reasons, agricultural experts co-operated with computer experts in order to develop a system that will support decision making in pests's control.

### **2.1.2 System's characteristics**

#### **2.1.2.1 Architecture**

System uses client server architecture. The client part is related to the Graphical User Interface, the Knowledge Representation module, the Interference engine and the Knowledge Base Maintenance module. The server part handles the data storage using the capabilities of the Database Management System (DBMS) [52].

The following tables represent diseases included in NEST (Table1.) and insects and mites included in the system's database (Table 2.). Earlier versions of the system and its historical evolution are described in ([76], [77]).

Diseases/Disorders	Tomato	Pepper	Aubergine
1. <i>Botrytis cinerea</i>	x	x	x
2. <i>Phytophthora infestans</i>	x		
3. <i>Alternaria solani</i>	x	x	x
4. <i>Cladosporium fulvum</i>	x		
5. <i>Fusarium spp.</i>	x	x	x
6. Verticillium Wilt	x	x	x
7. <i>Phytophthora Capsici</i>		x	
8. Anthracnose	x		x
9. Damping Off	x	x	x
10. <i>Septoria lycopersici</i>	x		
11. Stem Rot	x		
12. Corky Root	x		
13. Powdery Mildew	x	x	x
14. Sclerotinia	x	x	x
15. Bacterial Cancer	x	x	
16. Bacterial Leaf Spot	x	x	
17. Bacterial Speck	x	x	
18. CMV	x	x	x
19. TMV	x	x	x
20. Leaf curl virus	x	x	
21. Nitrogen deficiency	x	x	x
22. Phosphorus deficiency	x	x	x
23. Magnesium deficiency	x	x	x
24. Potassium deficiency	x	x	x
25. Iron deficiency	x	x	x

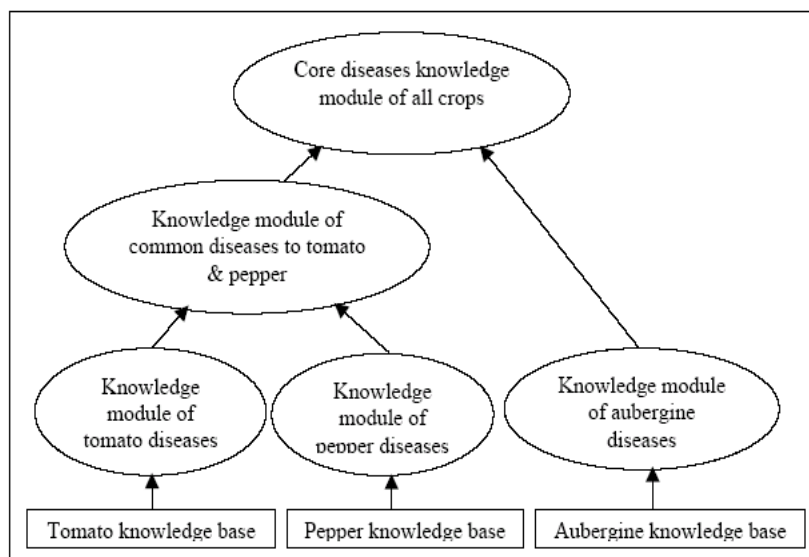
**Table 2:1 Diseases included in Nest(Maliappis, Sideridis, Mahaman, "NEST: A New Expert System Tool and its application to pest management", Proc. of the 3rd European Conference of the European Federation for Information Technology in Agriculture, Food and Environment (J. Steffe, Ed.), Vol. 2, 2001, pp. 421-426)**

INSECTS, MITES, NEMATODES	Tomato	Pepper	Aubergine
1. Aphids	X	X	X
2. Leafminers	X		X
3. Whiteflies	X	X	X
4. Tomato fruit worm	X	X	X
5. <i>Agriotes segetum</i>			X
6. Thrips	X	X	X
7. Colorado P. Beetle	X		X
8. Eggplant fleabeetle			X
9. <i>Spodoptera littoralis</i>			X
10. <i>Forficula spp</i>		X	
11. <i>Zonosemata electa</i>		X	
12. Mites	X	X	X
13. Nematodes	X	X	X

**Table 2:2 Insects and Mites included in Nest(Maliappis, Sideridis, Mahaman, "NEST: A New Expert System Tool and its application to pest management", Proc. of the 3rd European Conference of the European Federation for Information Technology in Agriculture, Food and Environment (J. Steffe, Ed.), Vol. 2, 2001, pp. 421-426)**

### 2.1.2.2 Knowledge base Structure

The approach used by the NEST for the knowledge representation is an adoption of the approach delineated in Knowledge for incorporation of time in rule-based knowledge bases [51]. The final KB for each crop is an assembly of several KB modules, which is much applicable to the development of a diagnostic expert system tool for the mentioned crops, since they have many common diseases.



**Figure 2:1: Structure of NEST Knowledge Base for diseases of crops (Maliappis, Sideridis, Mahaman, "NEST: A New Expert System Tool and its application to pest management", Proc. of the 3rd European Conference of the European Federation for Information Technology in Agriculture, Food and Environment (J. Steffe, Ed.), Vol. 2, 2001, pp. 421-426)**

### **2.1.3 Implementation**

According to the developers of the concrete application, all database operations have been implemented through SQL calls embedded into C++ code. Microsoft Visual C++ was the chosen development tool. Moreover, for data entry capabilities of the visual forms in Microsoft Visual C++ were used. Finally for the DBMS support Microsoft Jet Database Engine was utilized; the connection with the Engine was performed with the Open Database Connectivity technology (ODBC).

## ***2.2 Information Technology for Creation and Use of the PC "Phytopharmacy" Database***

The delineated application was integrated by Agricultural University of Bulgaria, Department of Computer Science, Plovdiv, and specifically by Dr. Onkov K. and his

research assistant Dimova D. The goal for the development of the concrete PC database is the storage and the structure of data with regard to the cultures, pests and permitted pesticides for use in a given country. According to the authors the constructed database *meets the requirements* of phytopharmacists, agronomists, farmers and economists and facilitates their decision-making [58].

### 2.2.1 Components of the proposed Information technology

The main components of the described information technology are,

- the data source
- the basic operations for the creation of the “Phytopharmacy” database
- the external data views addressed to different user groups [58]

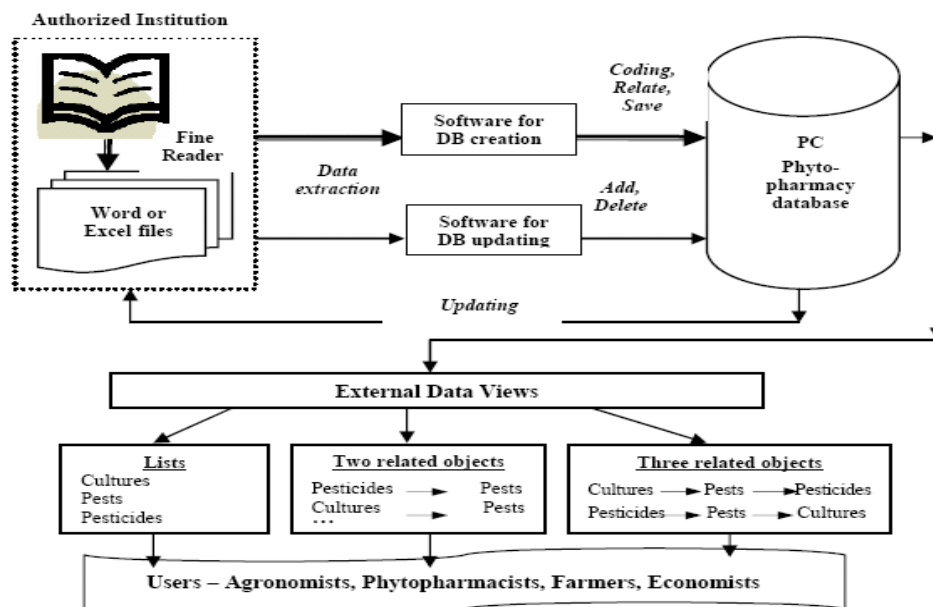


Figure 2:2: Components of the developed Information Technology(Onkov K., Dimova D.,“Information technology for creation and use of PC “Phytopharmacy” database”, Proc of HAICTA 2006 International Conference on “Information Systems in Sustainable Agriculture, Agroenvironment and Food Technology”, 20-23 September 2006, Volos, Greece



The *external data views* easy access to the data and relations between them stored in the “Phytopharmacy” database in order to support the decision-making by a wide range of agricultural specialists. Three types of external data views are developed; *lists* which present cultures, pests and pesticides and also characteristics of the pesticides. Moreover the second type of external data views is the so called two related objects and finally the three related objects. External data views are implemented by the graphical forms and user friendly software [58].

### 2.2.2 Conceptual Model of the PC Database

The software developed for the creation of the “Phytopharmacy” database performs the following basic operations:

- finds text objects (words or phrases) in the input files and extracts them;
- applies the coding system for the creation of relationships between text objects - cultures, pests and pesticides;
- saves the relational objects and their characteristics (active ingredients concentrations, etc) in accordance with the design of the blank database.

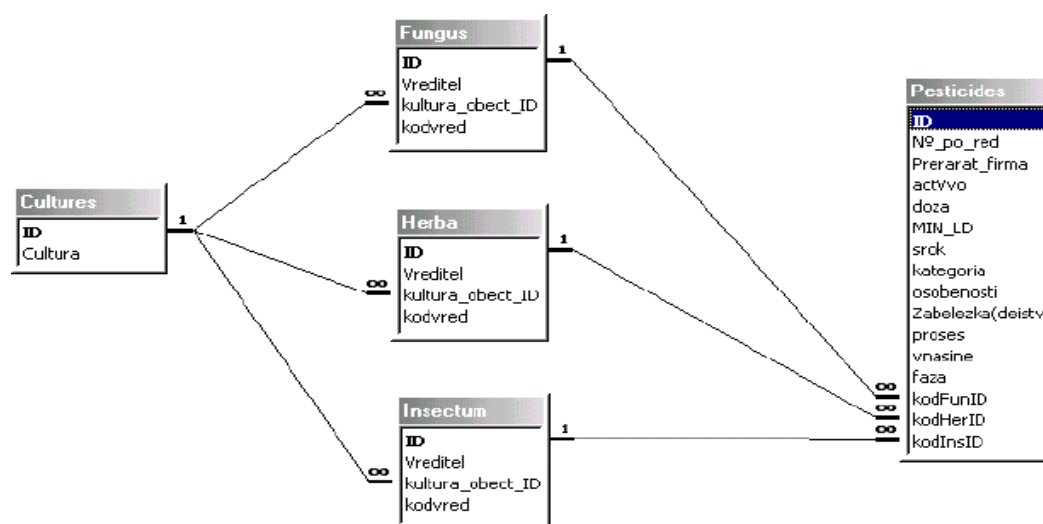


Figure 2:3: Conceptual model of the PC Phytopharmacy Database(Onkov K., Dimova D.,“Information technology for creation and use of PC “Phytopharmacy” database”, Proc of HAICTA 2006 International Conference on “Information Systems in Sustainable Agriculture, Agroenvironment and Food Technology”, 20-23 September 2006, Volos, Greece

According to the authors, the reasons for updating the data, is that the data included within the application should be in accordance to the rules and the laws of the Bulgarian government with regard to permission and prohibition of concrete pesticides. In the case that a pesticide is permitted it must be deleted and replaced/updated by a legal one. It is noted that the database type of the PC Database lays on the Relational Data Model. The reference book of permitted products for plant protection and fertilizers in Bulgaria [57] is used as a primary data source.

The developed database and user oriented software will facilitate decision-making by phytopharmacists and agronomists as well as farmers, who do not have sufficient knowledge of agricultural chemicals and pests.

### 3 Requirement analysis and project feasibility role to system integration procedure

#### 3.1 *Defining, validating and prioritizing initial requirements*

Primary task of the project teams when they apply ICT Management to any organization is to define, validate, document and prioritize initial requirements for the concrete application. Without this foundation the application which will be built will not fully meet the requirements of its users. Let us analyze one by one the above mentioned activities.

##### 3.1.1 **Defining requirements**

When defining the initial requirements for an application, it is conspicuous to get them from authorized sources; for **technical requirements** project team should work with technical experts; for **environmental requirements** the project team should work with people who understand the environment of the organization for which the application is developed; moreover, for the definition of **user requirements** the team should cooperate mainly with the user community. Harrison [38] shows that teams composed of people with wide range of skills consistently outperform homogeneous teams. Common approaches used for obtaining requirements from each of the mentioned source are:

- **Interviewing**: by interviewing users, technical experts and experts who can realize the business environment of the organization, the project team broadens its interpretation of the application. Furthermore, during the interview it can be decided who will be invited in the JAD or the CRC modeling and also new needs can be directly identified

- **CRC Modeling**: this is an approach in which a group of subject matter experts analyze their own needs for a system. CRC modeling typically starts with **brainstorming**, a technique in which people express whatever ideas they come up with the application
- **Use cases**: a use case describes a way in which a real-world actor – a person, an organization or an external system – interacts with the application. Specific examples of a use case are the **use-case scenarios** ([86],[91]) and the **use-case diagrams** [88]. In the fourth chapter of the essay examples of use-case scenarios and use-case diagrams are utilized for the description of the business processes of a concrete organization
- **User interface prototyping**: the goal of the user interface prototyping is to interpret the needs by showing people the possible design of the user interface of an application. In the case of the user interface prototype users are actively involved
- **Join Application Development**: Join Application Development is an organized meeting led by a facilitator and is arranged for gathering requirements and for designing a part of the whole application. The facilitator is responsible for organizing, running and summarizing the results of the meeting. Main advantage of the JAD is the cooperation of many people in order to define and document the requirements of the application. Conclusions are always more accurate when people are working as a team [6]

### 3.1.2 Documenting requirements

Having completed the task of defining requirements the next step is to document them. All the tools utilized for accurate definition are not of use if the project team does not document the conclusions drawn through these processes. The steps which should be followed for exact documentation of these presumptions are:

- Description of the needs
- Assignment of a unique number to each requirement
- Define priorities
- Name and describe the sources from which requirements are obtained
- Indicate any possible risks
- Document further sources of information about each requirement
- The outstanding objective of this process is the interpretation of those needs by the developers who will model the final application [6]

### 3.1.3 Validating requirements

Validation of the needs is another necessary process included in the ICT management's functions. Without this step, requirements can be misunderstood or not taken into consideration which might lead the project team to an incorrect modeling of the application. Three primary techniques used for validation of initial requirements are:

- **Use – case scenario testing:** a testing process in which users participate with ensuring that requirements are accurately defined
- **Prototype walkthroughs:** an analysis – testing process in which users work through a collection of use – cases to ensure that a specific prototype fulfills their requirements
- **User – requirement reviews:** a process through which the project team can control whether what is built meets the needs of the users and determine the objective of the project [6]

### 3.1.4 Prioritizing requirements

A tough reality when building an application is that time for every task to be integrated properly is limited. Thus, a supplementary role of ICT management is to prioritize the business needs and determine which of them must be met and which are not necessary to be fulfilled. The process utilized for this scope is called **requirements triage** and includes what we “must have”, what is “good to have” and what is “not indispensable”. Objective of this process is to limit the application to what can be truly delivered. [6]

## 3.2 *Project feasibility*

Another fundamental role of ICT management in the modern enterprises is the performance of the **feasibility study** for every application which is developed. The basic reason for this performance is the **justification** of the project which lays foundation for successful modeling of the application. In other words the project team has to develop an adaptation that makes sense from an economical, technical and operational point of view. Thus, economic, technical and operational feasibility must be performed in order to assert that this stage has been taken into consideration. Yet, for the ideal performance of this task, defects and risks of the adaptation have to be identified. Moreover, project management documents and project infrastructure should also be properly prepared.

### 3.2.1 Economic, technical and operational feasibility

When assessing the economic feasibility of an implementation the developers are trying to answer the following question: ‘How well will this implementation pay for itself?’. The answer is given by the **cost and benefit analysis** which compares the

real costs of the application to its real economic profits. There exist two basic factors which should be taken into consideration when estimating the economic feasibility of an application; the **qualitative** factors and the **quantitative** factors. The former refer to costs or benefits for which it is very difficult to identify a monetary value while for the latter the identification of a monetary value can be easily accomplished.

In addition to economic feasibility project teams have to determine the technical feasibility of the implementation. In this case the basic question that has to be replied is: ‘Can we build this application?’ The way of doing this is to investigate the technologies which will be used in terms of the adaptation’s development. Sometimes when technologies are to be used in house they do not work together, thus the first task of the expert teams is to verify this through a mini – project or technical prototype.

Having determined the technical feasibility of an application, meaning that it is already built, the final step is to verify whether it is possible to ‘maintain and support it once it is in production’ In other words the **operational feasibility** should be assessed as well. If the expert team comes to the conclusion that the operation cannot be supported as it is, maintain and support infrastructure of the present status should be improved [6].

### **3.2.2 Identification of risks**

Another reason for performing the feasibility study is to identify and define any potential risks. Risk identification takes place when the technical and operational feasibility are determined. Possible risks which are usually met are:

- The use of new and not tested technology
- Inexperience of the organization with the delivered technology
- No project of this size was performed with the given technology
- The use of several unknown technologies

- Inability to support the application when in production
- Waiting for software products which are promised but not yet released

[6]

### **3.2.3 Identification of defects**

Since no application works ideally a process throughout which any possible defects are recognized is demanded. Feasibility study is exactly this process. While performing this procedure any kinds of gaps or mistakes are identified and if possible they are enhanced. Without this process the application will be integrated overtime for the reason that any changes to observed mistakes are not performed on time [5].

The present chapter of the dissertation thesis is based on IT experts' practical experience on software creation or even integration of an entire Information System. From the author's standpoint a short reference to already written ideas about the way requirement analysis and feasibility study should be utilized throughout system integration procedure is essential since they comprise of the main analyzed subject of the present work.

By the end of the dissertation thesis, after deriving the algorithmic approach to to busisness process requirement analysis, by transforming the Use Case approach to the BORM methodology, the author's central idea is the Case Stusy analysis for demonstrating practically the usefulness of both mentioned and analyzed parts of the entire system integration process.



## 4 Object-Oriented tools for performing efficient and detailed *business process requirement analysis*

### 4.1 *Business Process Requirement Analysis with the Use Case Concept*

Requirement Analysis is an important factor for the successful final product derivation. Considering the type of the product which is integrated, which can be either a *system* or a concrete *software application* within a system, the presupposition for the commencement of the analogous project, is the emergence of an initial demand by a simple or expert user that some new process or some set of processes has to be created. In other words, in the beginning of an IT project, the so called *feasibility study*, either in the case of an Information System [72], either in the circumstance that a new business process within a constructed system has to be introduced, is the key element by which it will be finally judged whether the final product is reasonably integrated or not.

Regarding the business process introduction within the derived system or software, the final product's success mainly relies on the proper *requirement analysis*. The most common technique, for requirements specification, proposed by the Object Oriented Approach to system development, and through which successful business requirement analysis has been successfully carried out in practice, is the so called *Use Case Analysis*.

Use Case Analysis is the primary form for gathering requirements for a new software program or task that must be completed. It is a concept associated to both business and software requirements. The concept of the Use Case was initiated by Ivar Jacobson [43], who later contributed to both the Unified Modelling Language (UML) as well as the Rational Unified Process (RUP).

### 4.1.1 Use Case Analysis Parts

The Use Case methodology in terms of requirements derivation, is comprised of the following parts:

- *Use cases.* A use case (**Fig. 4:1**), according to Jacobson, is a behaviourally related sequence of transactions, performed by a user, in a dialogue with a system. Another definition of the Use Case, describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse [86]
- *Actors.* An actor (**Fig. 4:1**) is a person, organization, or external system that plays a role in one or more interactions with the concrete system. Actors are drawn as stick figures
- *Associations.* Associations between actors and use cases are indicated in use case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use case. Associations are modelled as lines connecting use cases and actors to one another, with an optional arrowhead on one end of the line. The arrowhead is often used to indicating the direction of the initial invocation of the relationship or to indicate the primary actor within the use case. The arrowheads are typically confused with data flow and as a result their used is usually avoided
- *System boundary boxes (optional).* A rectangle drawn around the use cases, called the system boundary box, to indicate the scope of the system. Anything within the box represents functionality that is in scope and anything outside the box is not. System boundary boxes are rarely used; a circumstance where its utilization is recommended is when use cases will be delivered in each major release of a system
- *Packages (optional).* Packages are UML constructs that enable you to organize model elements (such as use cases) into groups. Packages are depicted as file folders and can be used on any of the UML diagrams, including both use case diagrams and class diagrams. Packages can be used only when diagrams

become unwieldy, which generally implies they cannot be printed on a single page, to organize a large diagram into smaller ones

Cockburn [20] identified three levels of detail in writing use cases:

Brief use case consists of a few sentences summarizing the use case. It can be easily inserted in a spreadsheet cell, and allows the other columns in the spreadsheet to record priority, duration, a method of estimating duration, technical complexity, release number, and so on.

Casual use case consists of a few paragraphs of text, summarizing the use case.

Fully dressed use case is a formal document based on a detailed template with fields for various sections; and it is the most common understanding of the meaning of a use case. Fully dressed use cases are discussed in detail in the next section on use case templates.



**Figure 4:1: Actor and Use Case depiction**

### **4.1.2 Use Case Scenarios**

A *scenario* is a brief narrative description with regard to a hypothetical use of a system. Scenarios include information about goals, expectations, motivations, actions and reactions. The act of capturing requirements with use cases is sometimes referred to as Scenario Defined Problem [23]. The description of the primary successful path through the use case is the so called *Main Success Scenario*. Main Success Scenario is also defined as the way the primary actor accomplishes in a straightforward manner

[1]. In this case, success is achieved after completing all the steps or scenarios that follow the main success scenario. One or more scenarios may be generated from each use case, corresponding to the way of achieving a concrete goal. Scenarios are neither predictions nor forecasts, but rather delineations of the way in which a system is used in the context of daily activity. Scenarios are frequently used as part of the system development process. Scenarios are written in plain language, with minimal technical details, so that stakeholders such as system integrators, business process specialists and technical experts can have a common understanding of the goal for developing the desired system. A stakeholder is an individual or department that is affected by the outcome of the use case [12] and might be called on to provide input, feedback, or authorization for the use case [50].

It can be briefly stated that a scenario:

- Tells who is using the system and what is to be accomplished
- Provides a realistic, fictional account of a user's constraints: when and where they are working, why they are using the system, and what they need the system to do for them
- Describes any relevant aspects of the context in which the user is working with the system, including what information the user has on hand when beginning to use the system
- Gives the user a fictional name, but it also identifies the user's role, such as student, faculty member, staff, or general public
- Indicates what the user regards as a successful outcome of using the system

It should be mentioned that the exact documentation of the Use Case Scenario is an important prerequisite for the creation of efficient business processes, and therefore an invaluable tool of the overall business process requirement analysis stage.

### 4.1.3 Use Case Diagrams

A use case diagram is a type of behavioral diagram defined by the Unified Modeling Language (UML) and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actors. Roles of the actors in the system can be depicted.

Use Case diagrams are formally included in two modeling languages defined by the OMG. Both the UML and SysML standards define a graphical notation for modeling use cases with diagrams. One complaint about the standards has been that they do not define a format for describing these use cases. Generally, both, graphical notation and descriptions are important as they are the documentation of the use case, showing the purposes an actor can use a system for.

The use case diagram shows the position or context of the use case among other use cases. As an organizing mechanism, a set of consistent, coherent use cases promotes a useful picture of system behavior, a common understanding between the customer/owner/user and the development team.

Interaction among actors is not shown on the use case diagram. If this interaction is essential to a coherent description of the desired behavior, perhaps the system or use case boundaries should be re-examined. Alternatively, interaction among actors can be part of the assumptions used in the use case.

## 4.2 *Business Object Relation Modeling (BORM)*

Business Object Relation Modeling originally started in 1993. It was intended to provide support for the construction of Object-Oriented software systems based on pure Object-Oriented languages and environments such as *Smalltalk* ([10], [41], [42])

and object databases. It has subsequently been realized that this method has significant potential in capturing knowledge of business processes ([9], [18], [36]) and process information which is demanded during the early stages of information system development cycle ([16] [24], [27], [46], [61]). BORM is an object-oriented and process-based analysis and design methodology, which has proved to be effective in the development of business systems [98].

All the above mentioned elements are related to business process requirement analysis. As it was already mentioned, the most common technique utilized worldwide for detailed requirement analysis is the Use Case modelling. Use Cases are often the foundation of most Object –Oriented development methods [43]. However, it has been stated by many IT experts, who strongly recommend the UML tools such as Use Case diagrams followed by the Sequence, Collaboration and State Transition Diagrams for the integration of efficient and effective requirement analysis, that the aforementioned tools are too oriented at the programming concepts and quite weak in terms of business logic and business process modelling. The above stated deficiencies of the Use Case analysis are highlighted by Fowler [34]. The appearance of many process modelling tools was an attempt of business analysts and information system integrators to overcome the above stated issue. The emerged business process oriented methodologies are demonstrated at the following table (*Table 4:1*). An important remark related to these modelling tools, is that they are all based on theoretical and mathematical concepts such as Petri Nets, Flowchart Diagrams and State Machine, which comprise of the roots of the computer science.

approach	theory behind	advantages	disadvantages
EPC - Aris	Petri Nets	very popular in Europe, perfectly supported by Aris CASE Tool, easy and comprehensible method for domain experts	weak relation at subsequent software development techniques, slow analysis, low expressiveness of large models
UML Activity Diagram	flowchart	industry standard, supported by many CASE tools	too software-oriented, difficult to understand by domain experts
UML sequence and state-chart diagram	state machine	industry standard, supported by many CASE tools	too software-oriented, difficult to understand by domain experts
Workflow Diagrams	flow chart	easy and comprehensible method for domain experts, perfectly supported by many business CASE Tools	not very popular in Europe where Aris takes the dominant place, weak relation at subsequent software development techniques

**Table 4:1: Approaches for modeling business logic( Knott et al., The role of Object – Oriented Process Modeling in Requirements Engineering phase of Information Systems Development, Conference Proceedings EFITA 2003, Debrecen, Hungary, 2003, p 300 – 307)**

### 4.2.1 Characteristics of the BORM approach

BORM is based on the spiral model for the development life cycle ([55], [97], [98]). Each loop of the Object Oriented Spiral model is comprised of the following stages (*Fig. 4:2*):

- Strategic analysis
- Initial analysis
- Advanced analysis
- Initial design
- Advanced design
- Implementation
- Testing

The first three stages are referred to as *expansion stages*; expansion terminates with the detailed analysis conceptual model, which fully describes the solution to the problem from the requirements point of view [78]. The remaining stages are the so called *consolidation stages*; they are strongly related to the process of developing from the ideas that stem from the expansion stages to a practically implemented application or system. The BORM approach to requirement analysis is characterised by a smooth transition from between Object Oriented analysis and design, contrary to other methodologies, since the above mentioned conceptual model is transformed gradually into the finalized software design (*Fig. 4:3*)

## Spiral Software Development Life Cycle in BORM

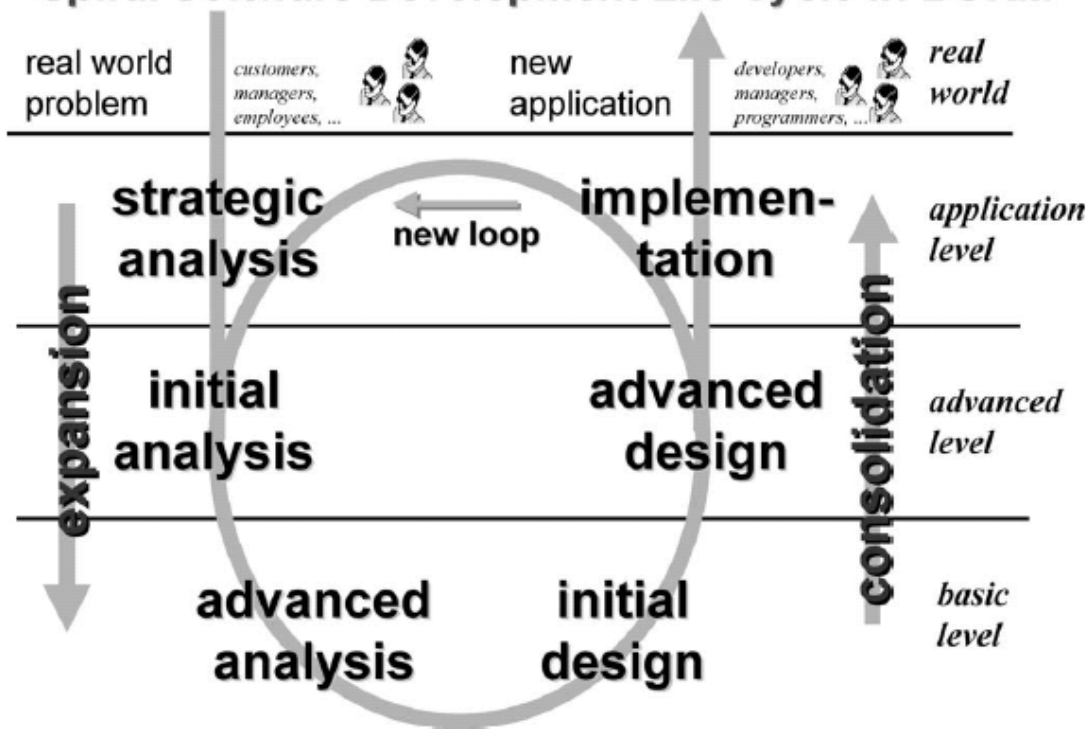


Figure 4:2 The BORM Stages (Knott et al., Knowledge Based Systems, 16, 2003: 77-89)

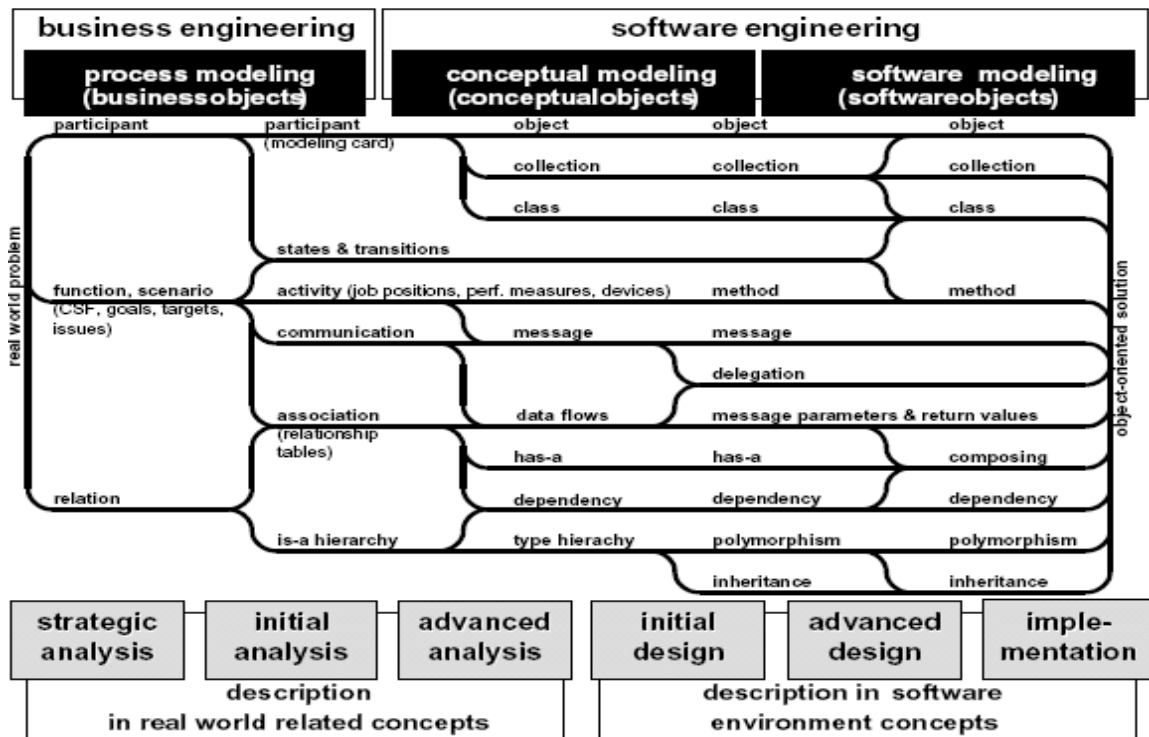


Figure 4:3: BORM Information Engineering Process (AGRIC. ECON. – CZECH, 52, 2006 (4): 165-172)



In BORM, every object is considered to be a machine of states and transitions. The definition of each state is based on the object data associations, while the transitions are defined by their behavior, to transform the object from an initial state to a final state. As a result, it can be underlined that BORM objects have similar characteristics of Mealy- type automaton [24]. All the above stated characteristics, are clearly demonstrated by the *Process – Participant Interaction Model (Fig 4:4)*.

#### 4.2.2 The advantages of BORM methodology to requirement analysis

The most significant advantage of the BORM methodology is that it is a practically implemented tool in many real – life projects. The following table (**Table 4:2**) is comprised of the list of the existing applications where the BORM methodology was indeed the key approach to process oriented requirement analysis.

Project	NSF	NS	NPD	NO	ANS	ANA
National agrarian chamber (analysis and design of software for fruit market public information system)	4	7	7	6	4	4
Hospital complex (BPR of organization structure)	6	12	12	8	10	12
TV and radio broadcasting company (BPR and company transformation for open market)	4	9	9	14	8	8
Regional electricity distribution company (customer information system analysis)	12	19	19	23	12	12
Regional electricity distribution company (failure handling information system analysis and prototype implementation)	19	31	34	27	13	14
Regional gas distribution company (BPR of all company)	28	81	97	210	11	12
Regional gas distribution company (BPR of all company)	23	60	63	120	12	12

NSF, number of system functions; NS, number of scenarios; NPD, number of process diagrams; NO, number of objects (participants) (only objects having activities, objects realizing data flows in processes are not included here); ANS, average number of states per object; ANA, average number of activities (in BPR projects, each activity includes about 4–6 additional business related entities (goal, job position, success factor, etc.) per object.

**Table 4:2: Real Projects where BORM approach to BPM is used (Knott et al., Knowledge Based Systems. ,16, 2003: 77–89)**

The second advantage of the concrete analyzed object - oriented approach is that throughout the above mentioned projects, the conclusions derived and the impressions made by the end users, is that the interaction of the various system components is clearly and thoroughly analyzed. The graphical representation of the processes, with the help of the Process – Participant Interaction Model is the key factor to the simplicity of the BORM approach. Example of Process – Participant Interaction Model is demonstrated in **Fig 4:4**, and also throughout the following sections of the present document.

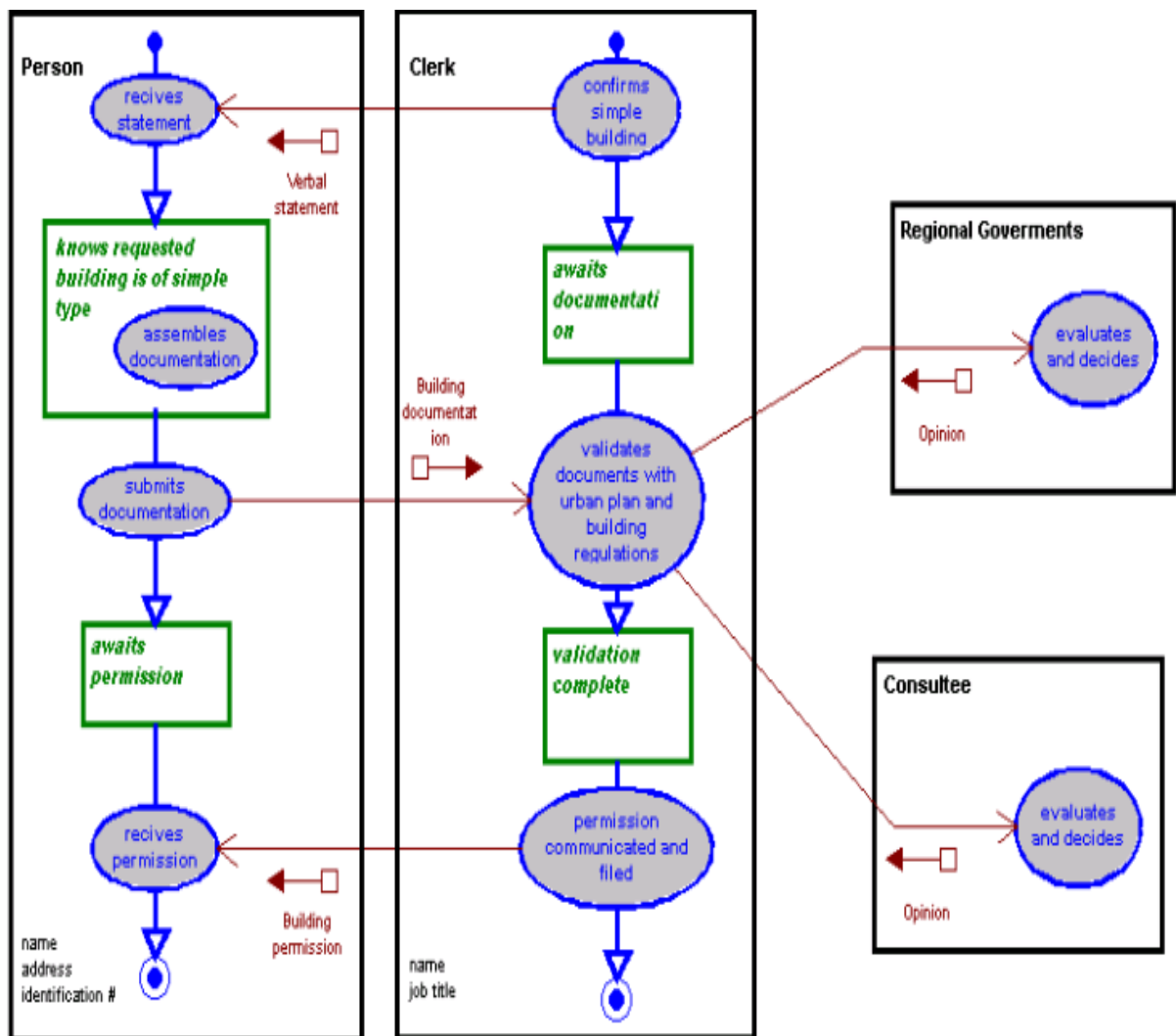
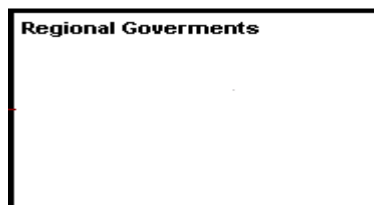


Figure 4:4: Fig 9 : Process – Participant Interaction Model example (Source: BORM – Business Object Relation Modeling , [http://www.cse.msu.edu/ICRE2000/Merunka/borm\\_html/index.html](http://www.cse.msu.edu/ICRE2000/Merunka/borm_html/index.html))

The core notations which are depicted, and which comprise of the key elements of the Process – Participant Interaction Diagram are the following:

Participant is basic element of the defined process diagram. In the above modeled process, *clerk* is a participant and interacts with other participants i.e. *person*, *consultee* and *regional governments*.

Symbol:



Activity, comprises of a participant's action in order to interact with other objects. It actually demonstrates participants' behavior between them.

Symbol:



Communication between participants has two forms:

- Communication parameters which are data that could be part of the communication. Parameters with the opposite direction form the communication represent reply from the activity invited for communication.

Symbol:



- *Communication* represents connection of two activities between participants. It comprises of a message abstraction between objects.

Symbol:



*States and transitions* of the participant represent the so called *participant's role* throughout the process. The role is comprised of the following parts:

- *Beginning stage* that uses the symbol



- *End stage* which is represented by the symbol



- *State* which is a part of the process in which the participant expect some reaction and is represented by the following symbol



*awaits  
permission*

- *Transition* which is the mean through which the participant is transferred from the one state to the following. The activity is the key element by which transition is fired. Transition is symbolized as follows:



The third advantage stated by experienced Information System developers, is that requirement analysis is performed 3-4 times faster than in the occasion where ARIS and Rational Unified Process are utilized. This statement was made by Deloitte and Touche consultants, who have used all these methods. Furthermore, in a similar way Smalltalk and Java developers are also keen on the BORM approach, since they find attractive the fact that it exploits collection concept, and not just classes, contrary to UML.

Final mentioned advantages of the analyzed method are the following:

- Many Object – Oriented methods such as OMT ([29], [62]) or UML [14], refer to concepts such as quantifiers, links between classes, aggregations. The aforementioned concepts are considered to be extremely useful for software implementation since they are too ‘computer oriented’ and necessary for hybrid object – oriented programming languages such as C#, C++ and Java. On the other hand, in the case that stakeholders are not familiar with computer – oriented concepts, communication between IT experts and stakeholders cannot be achieved at the early stages of system development and throughout requirement analysis phase. BORM methodology on the other hand can be successfully utilized in this circumstance while it is business oriented, and it can be consequently absorbed by stakeholders and end users.
- The term, participant, is used to denote all entities in the problem domain that have a role in the process. A Process – Participant interaction model shows its

relevant activities and states. Participants viewpoint is referred to as the internal view of a process.

- The BORM use of representing a process activity trace enables the developer to easily identify who is involved in each activity and their particular responsibility to that activity.

### ***4.3 The “Use Case To BORM Transformation Algorithm” (UCBTA)***

Requirement analysis is a complicated process in terms of understanding and accuracy, as far as the end user’s demand is concerned. Practical IT project experience has proved that inaccurate requirement analysis can undoubtedly lead to improper application operation. Throughout the previous part of the current document, two excellent and practically functional approaches to requirement analysis were discussed and analyzed. Both of them, have been used in many real-life projects so far with success; it can be though stated that the discussed process of requirement analysis, in the case that the final application (system or software) is harder than usual to integrate, need to be ameliorated for achieving the desired result.

The idea of requirement analysis amelioration, based on the utilization of both of the aforementioned approaches is discussed in this part of the document. Precisely, the author of the parent document proposes the construction of an algorithm with which Use Case Requirement Analysis methodology, will be step – by – step transformed to the BORM Requirement Analysis approach. The steps and all the necessary components of the mentioned algorithm will be utterly defined so that its functionality will be clear enough even to readers with low algorithmic and mathematical background, but who can though absorb its goal, as far as both system and software integration procedure is concerned.

### 4.3.1 Parallel comparison of Use Case and BORM Business Process Models

From what is mentioned above, the *primary goal of the creation of the UCBTA Algorithm is the accurate and clear definition of the end users' business needs, in the case of the integration of a complicated information system or application.* In other words, the analysts will be able to defend their claim of an ideal requirement analysis to the end users, who may not be able to absorb computer based requirement analysis, but who can very easily understand business flows as far as their analyzed system is concerned.

The author's basic idea for the initiation of the construction of the discussed algorithm, is based on the comparison of all levels of both requirement analysis models, and on the effort to implement *level – to – level* parallelism and transformation from the Use Case approach to BORM methodology. Throughout the following parts of the parent document, the definition of the overall level – to – level transformation will be performed.

The general schema of the UCBTA Algorithm is illustrated in **Fig. 4:5**. As a starting point, the algorithm's input will be defined. Additionally and as the algorithm's construction proceeds, the step by step transformation will be in detail delineated. Finally, the completed final algorithm's output will be also explained. It should be stated though, that at present, the author's effort illustrates the algorithm's initial idea in a quite informal manner. The illustration of the analyzed approach is performed through the plain flowchart. The mathematical analysis of the defined algorithm is based on the finite state automaton [61]. The formal mathematical algorithm's description will be analyzed in future works.



**Figure 4:5 The General Schema of the UCBTA Algorithm**

#### 4.3.1.1 UCBTA *Input* – Process Definition

Considering the basic algorithmic notation, it has to be stated that the first element that has to be stated is the so called *input* of the algorithm. When it comes to requirement analysis, and especially as far as the Use Case Analysis is concerned, the key *Business Process* has to be defined. The analysts need to be aware of the functionality of the process, and to examine the so called *process feasibility*. In the case that the decided policy is that the concrete process must be included in the final application or system, then its precise definition has to be taken into consideration as the input element of the UCBTA algorithm.

#### 4.3.1.2 UCBTA 1<sup>st</sup> Part – Defining the Use Case

Having defined properly and in detail the process for which exact requirement analysis is to be performed, the analysts have to move on to the first step of the transformation algorithm. To be more precise, throughout the first step of the described transformation, the analyst still places oneself to the field of Use Case Analysis. Thus, the transformation path has not yet been explored.

The requested task throughout the first step of the algorithm is the *Use Case Definition* or *Use Case Name*. It is clear enough that a properly selected Use Case Name is the key element that will enable the ideal communication between experts and non-experts, meaning analysts and end users of the system or application. The Use Case Name should be defined according to the general UML rules [43].

Throughout the present part of the current dissertation thesis it has to be mentioned that the defined Use Case is identical or part of the aforementioned process of the input part of the UCBTA algorithm. Thus, if the letter *P* will be utilized instead of the word *Process* and simultaneously the symbol  $U_C$  will be used instead of the *Use Case*, then the relation between UCBTA elements will be the following:



$$U_C \subseteq P \quad (a)$$

#### 4.3.1.3 UCBTA 2<sup>nd</sup> Part – BORM general function definition

The following step of the UCBTA algorithm is the definition of the so called *BORM general function*. Some very important remarks that have to be mentioned, with regard to this second step of the process are the following:

- The Use Case defined throughout the first step of the overall process is a part of or equivalent to the BORM general function; in other words, a sort of parallelism is performed in terms of relationship between the two models, and it can be claimed that the BORM function actually stems from the Use Case definition.
- The concrete step is considered to be the first movement for the connection of the two Models; thus it is a step of major importance since it is regarded as the *transformation initiation*.

Taking into account the above mentioned remarks, the gravity of the analyzed second step of the algorithm can be realized.

By using concrete symbols in a similar way as it was performed throughout the previous paragraph and by using the symbol  $B_F$  to represent the BORM Function the following relation between Use Case and BORM Function will be valid:

$$U_C \subseteq B_F \quad (b)$$

According to the author of the current dissertation, the process which is the input UCBTA step is also part of the BORM Function. As a consequence the following relation will be valid as well:

$$P \subseteq B_F \quad (c)$$

The BORM function is actually comprised of several Use Cases and the process is a subtotal of the specific function as it will be seen during the final chapter of the present dissertation thesis where the UCBTA case study for the Greenhouse IPM domain will be thoroughly described.

#### **4.3.1.4 UCBTA 3<sup>rd</sup> Part – Considering Use Case Actors**

By defining the Use Case Actors, the algorithm proceeds with the initiation of a changeover path between the two models. The current UCBTA step is also presumed as a movement of major importance, due to the above mentioned characteristic. From this point, the algorithm's core philosophy, which suggests the changeover between the two models till the final output is produced, starts being discerned by the reader of the present document. The Use Case Actor definition is performed in accordance to the *Use Case definition*. The symbol utilized by the author of the current research is the  $U_A$ .

#### **4.3.1.5 UCBTA 4<sup>th</sup> Part – BORM Participant determination**

The fourth step of the UCBTA algorithm, which is characterized by a second changeover to the BORM business process model, presupposes the *actor* definition

through the Use Case model. The actors' names which are provided during the third step, are now utilized, without any title alterations, in order to perform the participants' definition with regard to the BORM model. The symbol that is used for the describing the BORM participant is  $B_P$ .

The successful algorithmic transformation of the Use Case model to the BORM model is undoubtedly threatened by loss of data. One of the author's initial concepts defined in order to prevent the loss of data during the transition procedure, is as it was above mentioned the Actors' names that are utilized in the BORM model as well in order to name the BORM participants. As a result the next valid relation of the UCBTA procedure is the following:

$$U_A = B_P \quad (d)$$

#### **4.3.1.6 UCBTA 5<sup>th</sup> Part – Use Case Main Success Scenario Statement – Initial step**

By determining the participants of the BORM model, which is an important element of the delineated algorithm, the transformation process proceeds with the main success scenario statement. The initiation of the process delineation occurs, not only in terms of Use Case Analysis but also in accordance to the BORM requirement analysis model. The Use Case Main Success Scenario is comprised of several parts or steps. The first Use Case step is equal to the BORM initiation. Main Success Scenario must be a subset of the above analyzed BORM General Function. Thus it has to be noted that the 5<sup>th</sup> UCBTA algorithm part comprises of a decision step.

The intention is the completion of the Main Success Scenario which was initiated throughout the 5th part of the UCBTA procedure. The definition of a complete and precise main success scenario is of great importance as far as the Use Case transformation BORM is concerned. All Use Case steps must be recorded in a

detailed manner. BORM diagram construction will be based on the thoroughness of the *main success scenario*, and all analysts should bear in mind that possible loss of data during the transformation, will result to poor and unsuccessful requirement analysis.

It was mentioned that the precondition for proceeding to the following part of the algorithm, is that the Main Success Scenario must be a subset of the above analyzed BORM General Function. In the case that the mentioned precondition is not fulfilled the main success scenario statement is modified by the analyst.

The symbol  $U_{MSS}$  is used for representing the Use Case Main Success Scenario; thus its relation to the BORM General Function will be the following:

$$U_{MSS} \subseteq B_F$$

#### **4.3.1.7 UCBTA 6<sup>th</sup> Part – BORM Initiation Statement**

With the successful completion of the 5<sup>th</sup> step of the UCBTA algorithm, and having controlled the feasibility of the defined Use Case and the corresponding Main Success Scenario, the algorithmic chain involves an additional changeover between the two models, with the introduction of the so called *Initiation*. With respect to the parallelism performed between the two models, the Initiation statement is a task which relies on and stems from the Use Case Main Success Scenario. It can be also stated that the philosophy and the algorithmic role of both model elements is similar; consequently the *Use Case Main Success Scenario Initial step* definition, can be used for the definition of the BORM Initiation as well. The BORM Model completion requires the definition of the so called *Action* and the expected *Result*. However, further analysis with regard to the two last BORM elements will be performed throughout the last part of the current section

#### 4.3.1.8 UCBTA 7<sup>th</sup> Part – Defining Use Case Steps

The aforementioned Use Case Main Success Scenario, as far as the methodology of Use Case Analysis is concerned, should be followed by specified and concrete *Use Case Steps*. The achievement of the concrete specification is the ability of effective and efficient communication between expert IT analysts and the end users of the application or designed system, with respect to the system's or application's workflow. Thus, the importance of the accuracy of each workflow step is a critical and demanding task since its delineation has to be utterly absorbed even by stakeholders with limited or low IT background but who are able to understand the feasibility and the value of the described business process.

As it was mentioned throughout the 5<sup>th</sup> step of the UCBTA algorithm, all Use Case steps must be recorded in a detailed manner. and all analysts should bear in mind that possible loss of data during the transformation, will result to poor and unsuccessful requirement analysis. To prevent loss of data the author's concept, as far as Use Case Steps, is based on the following important principles:

- Literal transformation of the Use Case Steps of the Main Success Scenario to the BORM activities, states and flows
- Analysis of complex use case steps to sub steps, including also states and flows and not only activities.

The Use Case Steps are symbolized as  $u_1, u_2, u_3 \dots u_n$  and the corresponding sub steps as  $u_{1A}, u_{1B}, u_{2A}, u_{2B}, \dots u_{nA}, u_{nB}$ . Throughout the *UCBTA Transition Rules* section of the current document, it will be absorbed by the reader how the above mentioned notations are utilized for the smooth and without data loss transition from the Use Case model to the BORM business process requirement analysis methodology.

#### 4.3.1.9 UCBTA 8<sup>th</sup> Part – BORM Action specification

When performing the expected requirement analysis, and taking into consideration the BORM methodology, an additional task of major importance to the above BORM parts analyzed, is that of the defined *Participant Action*. What should be seriously taken into account during requirement analysis based on the UCBTA algorithm is that the *Use Case Steps have to be included in the specified action*. Moreover, and in accordance to the algorithmic aspect, in the case that the named *BORM Action* of the involved participant does not follow the aforementioned rule, *it has to be altered or modified*. It can be noticed that the concrete statement comprises of *the second decision algorithmic step*, which is of major importance, considering that in any other case the action is not feasible and the entire model transformation process cannot be finished due to performance disability. In other words, the algorithm in any other case cannot be applied. Furthermore, if the BORM Action is symbolized as  $B_A$ , then the relation which involves the BORM Action and the Use Case Steps will be the following:

$$B_A = \{u_1, u_2, u_3, \dots, u_n\}$$

and

$$u_1 = \{u_{1A}, u_{1B}, \dots\}, \quad u_2 = \{u_{2A}, u_{2B}, \dots\}, \quad u_n = \{u_{nA}, u_{nB}, \dots\}$$

#### 4.3.1.10 UCBTA 9<sup>th</sup> Part – Design the Use Case Diagram

Moving to the 9<sup>th</sup> part of the algorithmic process, presupposes the performance of an extensive and accurate study of the parts that refer to the Use Case Analysis and the parts which concern the BORM Requirement Analysis. The concrete part of the entire process, involves the schematic representation of the Use Case Analysis in which the Actors, the Use Cases and finally the associations between them are included. Moreover, the importance of the concrete step should be underlined, if it is considered as the first picture of the requirement analysis procedure, and the first diagrammatic

communication between IT expert analysts and stakeholders or end users of the integrated application or Information System.

#### **4.3.1.11 UCBTA 10<sup>th</sup> Part – Define BORM Data Flows**

With the completion of an analytical Use Case Diagram, the diagrammatic route of the final algorithmic output defined as *BORM Result* is initiated. As it was already mentioned, detailed dynamic modeling with respect to requirement analysis cannot be achieved, if the business process flow is missing. The Use Case analysis itself, is an important and tested Object Oriented UML in terms of requirement analysis, but the end user of the integrated application or the involved users of an entire Information System, always need to control that the process flow is in accordance to their initial expectations; for this reason the BORM requirement analysis involves the creation of a detailed depiction of the business process, which is the key element that misses even from the most extensive from the Use Case Analysis.

The BORM schematic representation though is not ideal, if the detailed analysis of the data transition between the BORM participants is not performed. Data flows between the process participants should be recorded according to the Use Case defined steps and sub steps. BORM data flow analysis with regard to Use Case defined steps comprises of the last step before the algorithmic expected final output is derived.

#### **4.3.1.12 BORM Diagram Construction (Object Relation Diagram)**

The UCBTA procedure reaches its expected schematic form; the design of the BORM diagram can now be successfully, efficiently and effectively implemented. Some important characteristics though of the aforementioned diagram have to be seriously considered by the business process analysts:

- When the construction of the desired BORM diagram begins a new non – deterministic finite automaton is initiated as well. Consequently a new algorithm starts and its final output comprises of the UCBTA output.
- The names of all *process states* of the diagram must be recorded in relation with the Use Case steps. Thus, the description utilized in order to keep track of all the steps of the process must be the same with that of the corresponding state depicted in the BORM diagram.
- The *data flow* names defined throughout the 10<sup>th</sup> part of the UCBTA algorithm should also stem from the Use Case recorded steps.

Example the business process depiction with the aid of the BORM diagram, as the target output stemming from the derivation of the UCBTA algorithm will be analyzed in the following chapter of the of the current document.

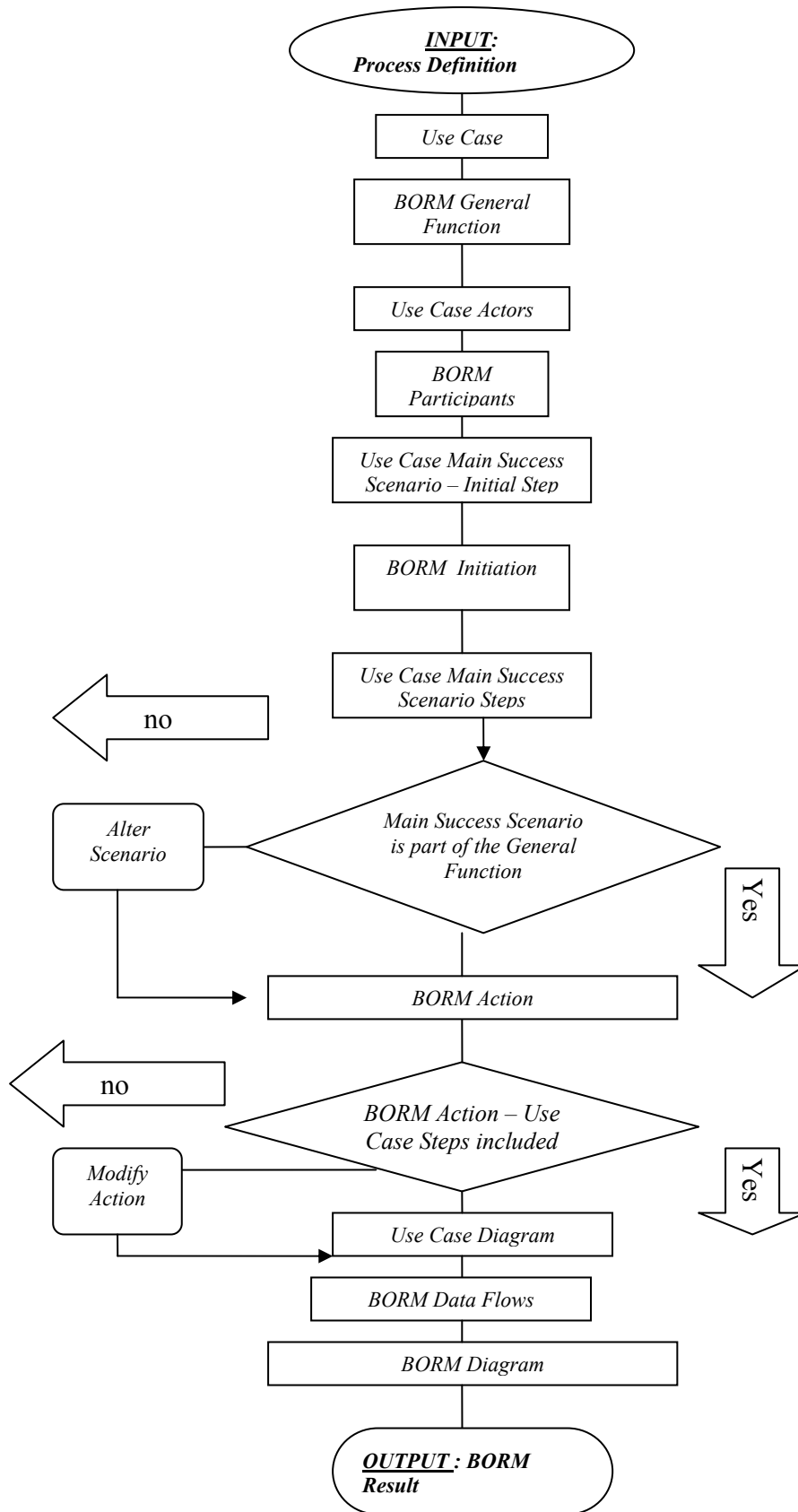
#### **4.3.1.13 UCBTA Output: BORM Result**

The entire algorithmic process reaches its final step; the BORM diagram as a sub algorithm of the UCBTA algorithm reaches its final output, which is depicted with the BORM activity. The BORM result is the actual goal of the defined process and is the cause of the business process requirement analysis derivation. Of course the BORM result is entitled with the same diction as the *Use Case final step*.

#### **4.3.2 UCBTA Representation**

The final part of the current section comprises of a detailed depiction of the entire process implemented in terms of the UCBTA algorithm derivation. The algorithmic schema is designed in detail (**Fig. 4:6**) from the transformation *input statement*, including all algorithmic steps and eventually the *planned output* which is of the scheduled *Business Process Diagram*.





**Figure 4:6: Flowchart of the UCBTA Algorithm**

#### ***4.4 Advantages of the Use Case to BORM Transformation Algorithm regarding business process requirement analysis***

The UCBTA approach to business process requirement analysis is introduced and constructed as a new methodology, so that the forthcoming steps of information system integration could be successfully implemented. The discussed topic regarding the usefulness of the analyzed algorithm is the whether specific scientific advantages and upgrading results will be inferred throughout the transformation from Use Case requirement analysis to BORM requirement analysis.

The advantages of the concrete approach by which the effective and efficient business process requirement analysis is implied, are defined and analyzed by the author of the current thesis throughout the current section of the thesis.

##### **4.4.1 UCBTA: A pattern – oriented methodology**

The first strong point of the introduced methodology, from the author's standpoint is its *pattern based concept*. Pattern in architecture is the idea of capturing architectural design ideas as archetypal and reusable descriptions. The term "pattern" is usually attributed to Alexander C. [2].

One of the worst experiences of IT system integrators is that when a project is demanded by an organization or an enterprise, regarding existing system's upgrade or the design and the development of a new system from scratch, is the time planned as far as the delivered product is concerned.

Indeed one of the main characteristics of the IT documents is the special section that concerns time estimation of the delivered product and the completion of each phase of the entire system integration procedure.

Of course time is a pressing factor in the cases when the phases of the project are not accurately defined from the beginning, and as the project is in progress, each phase looks chaotic without defined steps that should be initially set. As a result, the progress of the project delays and the end users of the integrated application are not satisfied by the IT project team.

Taking into account all the above mentioned time defects, it can be concluded that when the integration phases are carried out by utilizing pattern based paths with concrete and tested steps, there is no time loss for inspiring new ways of implementing the concrete phases.

Under this concept the UCBTA algorithm was inspired by the author of the current thesis. The idea is to follow a pattern based path in order to perform detailed and accurately defined business process requirement analysis by following the algorithm's steps, in order to prevent useless time spending on tasks that could actually have been avoided. The UCBTA steps, if followed as they are defined they enable system analysts IT experts and domain experts to perform efficient process requirement analysis and complete the concrete part of information system development on time.

#### **4.4.2 Mathematically defined approach**

Every algorithm that is scientifically introduced as an upgrade to the already defined approaches, must be based on terminologies and methodologies that cannot be ignored by the scientific community, and can be easily be absorbed by experts who can judge its pure and justified scientific foundation. As a consequence, the author of the present work utilizes mathematical approaches in order to analyze and justify the way the way the UCBTA algorithm is constructed.

The aforementioned mathematical approach is the so called *Non – Deterministic Finite Automaton*. This approach was initially introduced by the scientists in order to define and explain mathematically and in algorithmic manner computer process operations.

The author utilizes the concrete mathematical concept in order to analyze the UCBTA steps in the same way as the computer processes are defined, but concentrates on the business process layer in a similar way as the technical process layer is defined.

#### **4.4.3 Process feasibility analysis from project commencement**

Another characteristic of the UCBTA approach to the business process requirement analysis, which is mentioned and utilized by the author as an important benefit of the specific methodology, is that before the requirement analysis of the business process is initialized, the first task of the analyst is to consider the so called *process feasibility*.

Feasibility study approaches, methods and steps are analyzed in previous chapter of the current dissertation thesis. Nevertheless, even if it is not included in the algorithmic steps, a short reference to whether the business process existence makes sense is always demanded since information system business aspect should in all circumstances enforce the process automation and easy operation on a daily basis.

#### **4.4.4 Based on tested methodologies**

Another advantage of the UCBTA algorithmic methodology is the fact that the analyzed approach is based on the connection and the involvement of two very important and tested requirement analysis methods. As it was analyzed in previous

sections the algorithm's concept is inspired by the transformation of the Use Case methodology to the BORM method of business process requirement analysis.

As it was mentioned, the Use Case method is a UML approach to requirement analysis, which is the most famous tool, utilized in order to complete the specific part of the information system development. This means that the Use Case methodology is a tested method and has been already been used for many successfully integrated IT projects.

On the other hand, the BORM approach to business process requirement analysis is also a method which is tested by IT analysts in order to control if the defined process will be executed in the way the end users demand. Indeed, as it was previously stated this method which reveals the process flow in a dynamic way, enables the domain experts to absorb the exact operation of the business process, even those who have no computer knowledge and orientation.

Consequently, by utilizing two practically tested and accepted by the IT community methodologies, and more precisely by transforming the Use Case approach to the BORM method, it can be stated that the UCBTA approach can be undoubtedly utilized to analyze and define requirements of basic and vital processes. Moreover the transition from one method to the other will enable the analysts in order to fill the gaps that both processes might entail.

#### **4.4.5 UCBTA Transition Rules eliminate possible loss of data**

Each time a model is transformed to another model, as in the case of the UCBTA algorithm where the Use Case model is transformed to BORM, the most important element which guarantees the successful implementation of the specific transition, is the wiping out of the possibility of data loss emergence throughout the transition process.

In the case of the Use Case To BORM approach to requirement analysis, the author's general idea is not only the construction of a pattern based solution in order to perform requirement analysis; one of the most important features of the algorithm is that loss of data throughout the transformation procedure is eliminated due to the so called *UCBTA Semantics*.

According to the aforementioned semantics, the model transition is based on precisely defined rules. As it was mentioned throughout the previous section of the current dissertation thesis, an important rule for example is that the Use Case Actors are entitled as Participants as far as the BORM methodology is concerned after the entire transition is completed.

The UCBTA Semantics are utilized in order to implement exact transformation of the Use Case Main Success Scenario and all the subsets involved to BORM activities states and data flows without any possibility of facing the problem of data loss. The types of the semantics utilized in terms of UCBTA algorithmic process are analyzed in a following part of the present document.

#### ***4.5 Formal definition of the UCBTA approach to business process requirement analysis***

The UCBTA transformation algorithm is a modern approach for implementing effective and precise business process requirement analysis. The key factors according to which the official definition of the algorithm is considered to be complete are the *mapping* of concrete Use Case elements to the corresponding BORM elements, and also its formal *mathematical expression*. Throughout the following sections of the current thesis, the aforementioned parts of the algorithm's description are analyzed and justified in detail.

### 4.5.1 Mapping the Use Case approach to the BORM methodology

The mapping determination of the UCBTA algorithm stems from the fact that concrete elements of which the Use Case model is comprised, correspond directly to certain parts of the BORM methodology.

The initial part of the Use Case Analysis throughout the UCBTA approach to requirement analysis is the definition of the Use Case. The first changeover from the Use Case model to the BORM model, occurs when it is controlled whether the defined Use Case is part of the BORM general function; thus the first mapping of the procedure involves the Use Case and the BORM function.

Another crucial part of the UCBTA method, is the relationship between the *actors* of the Use Case model which are entitled or renamed as *participants* in the BORM model. Consequently, the presence of a new mapping is realized. Considering the *Use Case Actors* who are defined as persons, organizations, or external systems that play a role in one or more interactions with the information system that is to be constructed, correspond directly to the BORM participants of the target system (**Fig. 4:7**). An exception to the actors mapping to the BORM participants that should be taken into consideration is the declaration of the *database system as a BORM Participant*. Throughout the Use Case approach the *Database System* cannot be considered as an actor since it is an internal entity of the entire system. Consequently Database System can be taken into account as a BORM participants even if it is absent from the Use Case business process requirement analysis method.

The most important part of the Use Case approach to requirement analysis which is mapped in various ways to the BORM business process requirement analysis method is the *Main Success Scenario*.

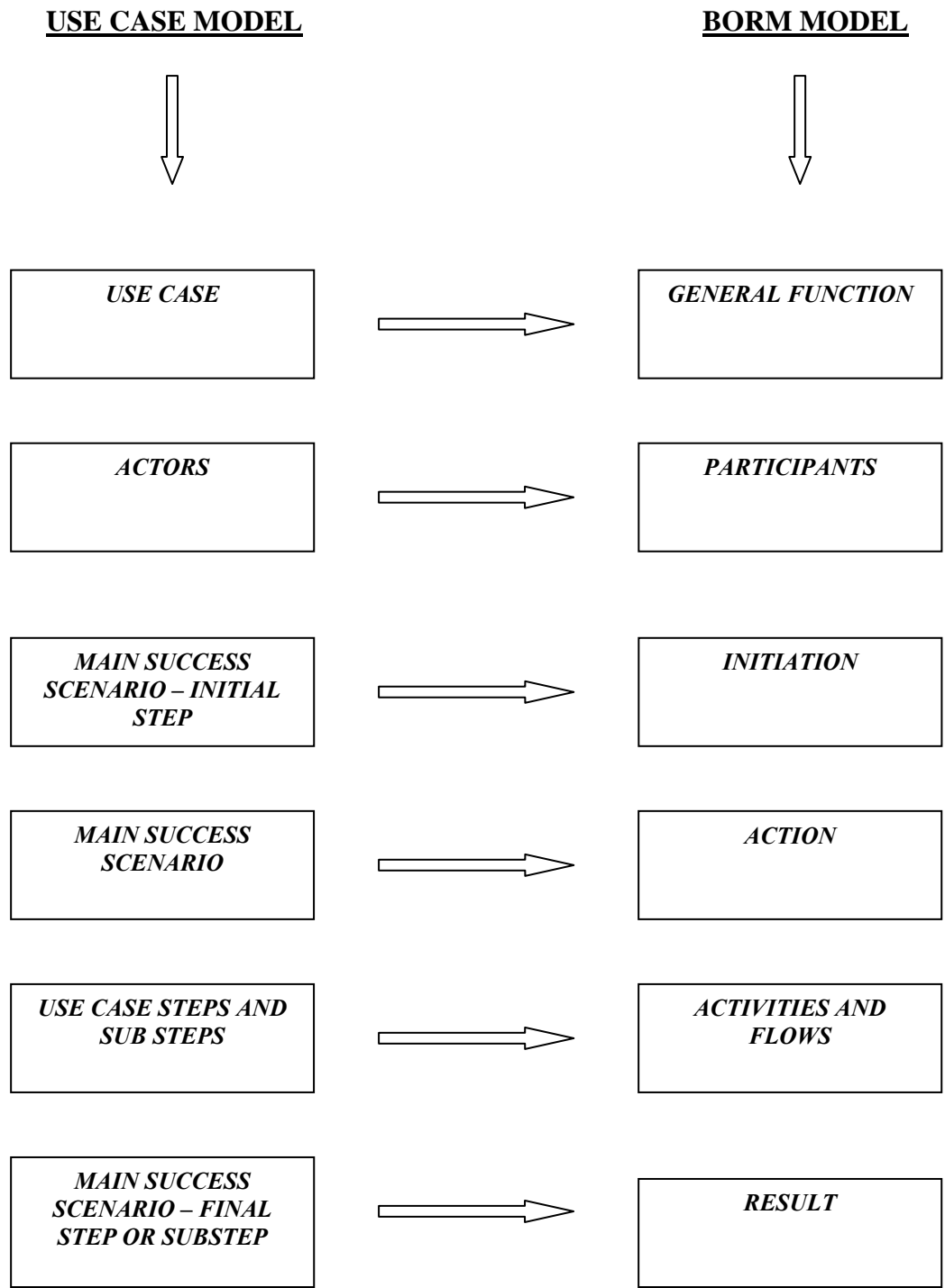
According to a previous section of the dissertation thesis, the first the so called *Initial Step* of the main success scenario is utilized in order to define the BORM *Initiation*. Moreover the steps between the first and the final step of the Use Case Analysis are

mapped to the BORM *Action*. The BORM action is expressed in relation with the Main Success Scenario steps and each step is part of the *action*. In the same way, the *final step* of the Main Success Scenario is utilized to express the BORM *Result*. As a consequence the main success scenario of the Use Case analysis is mapped to BORM approach in three different and very important ways.

The above mentioned main success scenario steps, are usually comprised of sub steps according to the concept of the Use Case Analysis and as a consequence to the concept of the UCBTA approach as well. Due to the danger of data loss throughout the transformation process of the Use Case method to BORM, the aforementioned Use Case Main Success Scenario steps and their sub steps should be in some way expressed in the BORM model as well. The BORM expression of the steps is implemented via *activities*, *states* and *data flows*. As a result, the final essential mapping for the complete delineation of the UCBTA algorithm is the one between Use Case steps and BORM activities, states and flows.

The main parts of the entire formal mapping of the UCBTA algorithm, are depicted at the following schema:





**Figure 4:7: Mapping of the Use Case Schema to BORM schema**

## 4.5.2 UCBTA Transition Rules [79]

For the precise comprehension of how the data loss is eliminated during the transformation of the Use Case Model to the BORM approach to business process requirement analysis, the author's concept, as far as the UCBTA algorithm's perfect functionality is concerned, is the creation of specific regulations that cover all the cases according to which the Use Case Main Success Scenario comprised of steps and sub steps is converted to BORM data flows, states and activities. Throughout the sections that follow the *UCBTA Transition Rules* ([79] – *Author's article accepted for publication*) are analyzed in detail. The tool utilized for the design, analysis and schematic depiction of the proposed rules is CraftCase [95].

### 4.5.2.1 Basic UCBTA transition rule

The basic type of the UCBTA transition rules comprises of the core transition from the Use Case Model to the BORM Business Process model. Throughout the core UCBTA transition, it is depicted how precisely a basic Use Case step of the main success scenario is diagrammatically adjusted to the BORM approach and depicted at the Object Relation diagram.

Supposing now that the above mentioned basic main success scenario Use Case step is divided into several sub steps; the Object Relation Diagram includes the aforementioned sub steps as well as they are described throughout the BORM method.

Let us assume a delineated Process and its corresponding Use Case A. The Use Case analysis also involves actors who take part in the process and are defined as Actor A and Actor B who are expressed as participants in BORM. Moreover, the Use Case step of the main success scenario is defined in the following way:

Actor A sends message to Actor B

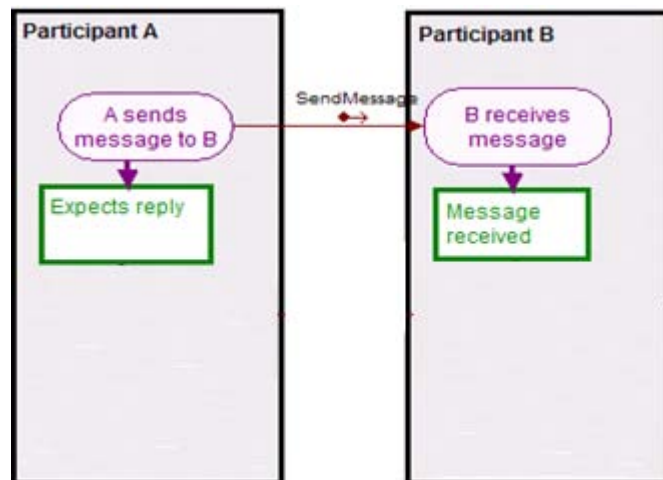
The aforementioned step is supposed to be comprised of the following sub steps as well:

Actor A expects reply

Actor B receives message

Message received by Actor B

The main point in which the author is interested is to transform the above written step and its subs steps to BORM activities flows and states, without any loss of data. Consequently, the Object Relation Diagram will be the following:



**Figure 4:8: BORM aspect of Process A after Primary UCBTA transition**

As it can be noticed by the reader, the main success scenario step is the corresponding BORM activity which is considered to be the starting point of the data flow. The activity that belongs to the participant who receives the message and the two states are considered to be the Use Case sub steps of the above mentioned step.

The currently defined rule, which is the most important of the 4 UCBTA Transition Rules, is the basis on which the following 3 rules are constructed.

#### **4.5.2.2 Primary Step UCBTA transition rule**

The second type of the analyzed rules of the Use Case transition to BORM is the *Primary UCBTA Transition*. Throughout the primary transition it is explained by the author how the *Initial* and the *second step* of the *main success scenario* are transformed to BORM activities, states and data flows.

The delineation of the primary transition is initiated with the assumption that UCBTA requirement analysis has to be performed for Process A. It is also assumed that the corresponding Use Case which is related to the aforementioned process is Use Case A.

The Use Case analysis also involves actors who take part in the process and are defined as Actor A and Actor B who are expressed as participants in BORM. Moreover, the initial and the second step of the main success scenario are defined in the following way:

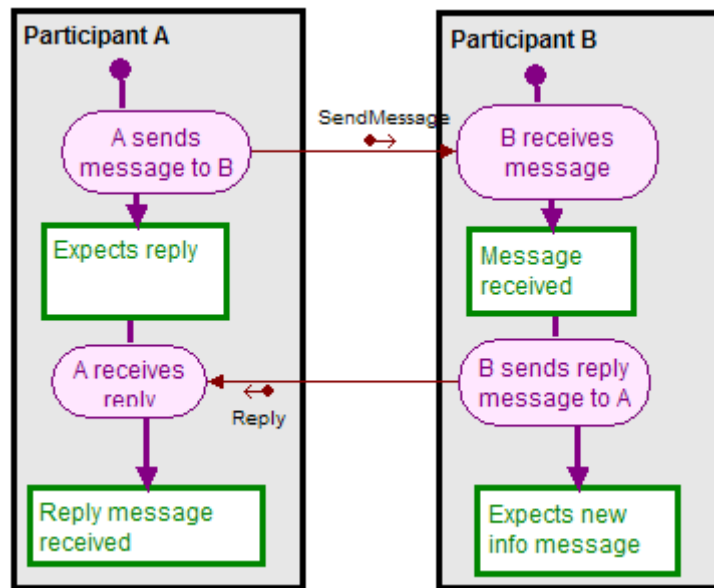
1. Actor A sends message to Actor B
2. Actor B sends reply message to Actor A

Considering the initial step of the main success scenario the sub steps involved are:

- 1a) Actor A expects reply
- 1b) Actor B receives message
- 1c) Message received by Actor B

In the same way the second step includes the following sub steps:

- 2a) Actor B expects new info message
- 2b) Actor A receives reply
- 2c) Reply message is received by Actor A



**Figure 4:9: BORM aspect of Process A after Primary UCBTA transition**

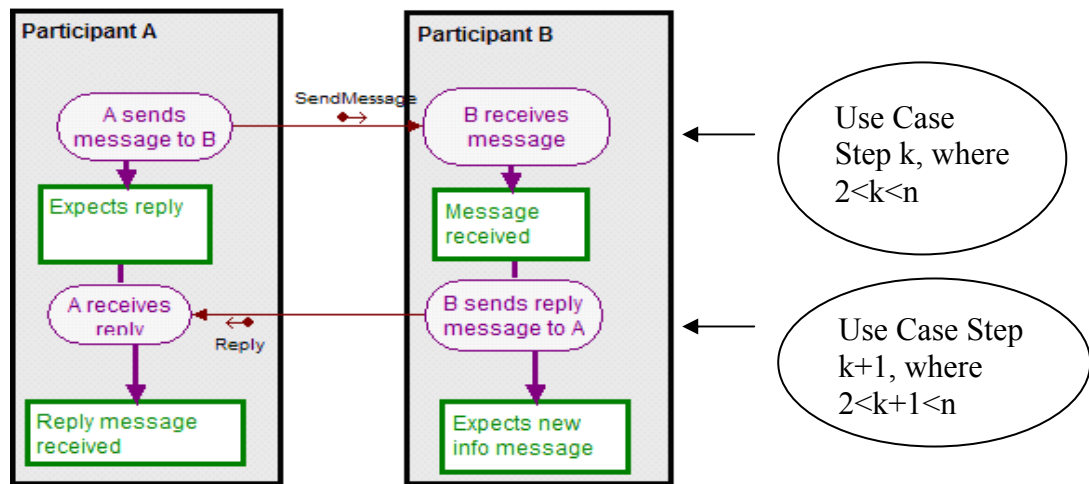
### 4.5.2.3 Middle Step UCBTA transition

The second type regarding the *UCBTA Transition rules* is the *Middle Step UCBTA transition*. The specific type follows exactly the same transformation path as the *Primary UCBTA transition type*; the main difference due to which the two types are distinguished is the fact that the *Middle transition type* refers to middle Use Case steps.

Supposing UCBTA requirement analysis should be implemented for a defined Process *B*. In a similar way as in the case of the first transition type its corresponding

Use Case B is defined as well. An additive presumption is that the Use Case Steps of which the analyzed Use Case main success scenario is comprised is  $n$ , where  $n \in N^*$ .

The *Middle UCBTA Transition rule* is applied for steps  $k$  and  $k+1$ , where  $2 < k < n$ ,  $k+1 \leq n$  and  $k, n \in N^*$ . The steps and sub steps of the main success scenario will be the same as in the primary UCBTA transition rule, and the BORM aspect is depicted in **Fig 4:10** Object Relation Diagram. It can be noticed that the difference with the first rule is that the middle step transition in BORM is without starting or ending points.



**Figure 4:10: BORM aspect of Process B after Middle Step UCBTA transition**

#### 4.5.2.4 Conditional UCBTA Transition Rule

The final type of the analyzed rules of the Use Case transition to BORM is the *Conditional UCBTA Transition*. The specified UCBTA transition rule is based on the fact that one or more steps of the Use Case main success scenario could lead the process in many different states. The *Conditional UCBTA Transition Rule* can be applied as a *Primary Step Conditional Transition Rule* or as a *Middle Step Conditional Transition Rule*. Due to the fact that the *Middle Transition Case* is more

general, throughout the current document only this type is defined and presented. The *Primary Step Conditional Transition* is defined by the author as a sub case of the *Middle Transition Case*.

The delineation of the *Middle Step Conditional Transition Rule* is initiated with the assumption that UCBTA requirement analysis has to be performed for Process C. It is also assumed that the corresponding Use Case which is related to the aforementioned process is Use Case C.

The actors involved are defined as Actor A and Actor B who are expressed as Participants in BORM. It should be noticed that the author's concept with regard to the *Conditional Transformation Rule* includes an *IF Statement* in the case where the Use Case Step can lead to more than one results:

1. Actor A sends message to Actor B
2. Actor B replies to Actor A, if the message is recognized
3. Actor B rejects message, if message is not recognized, and procedure terminates

Considering the initial step of the main success scenario the sub steps involved are:

- 1a) Actor A expects reply
- 1b) Actor B receives message
- 1c) Message received by Actor B

In the same way the second step includes the following sub steps:

- 2a) Actor B expects new info message
- 2b) Actor A receives reply
- 2c) Reply message is received by Actor A

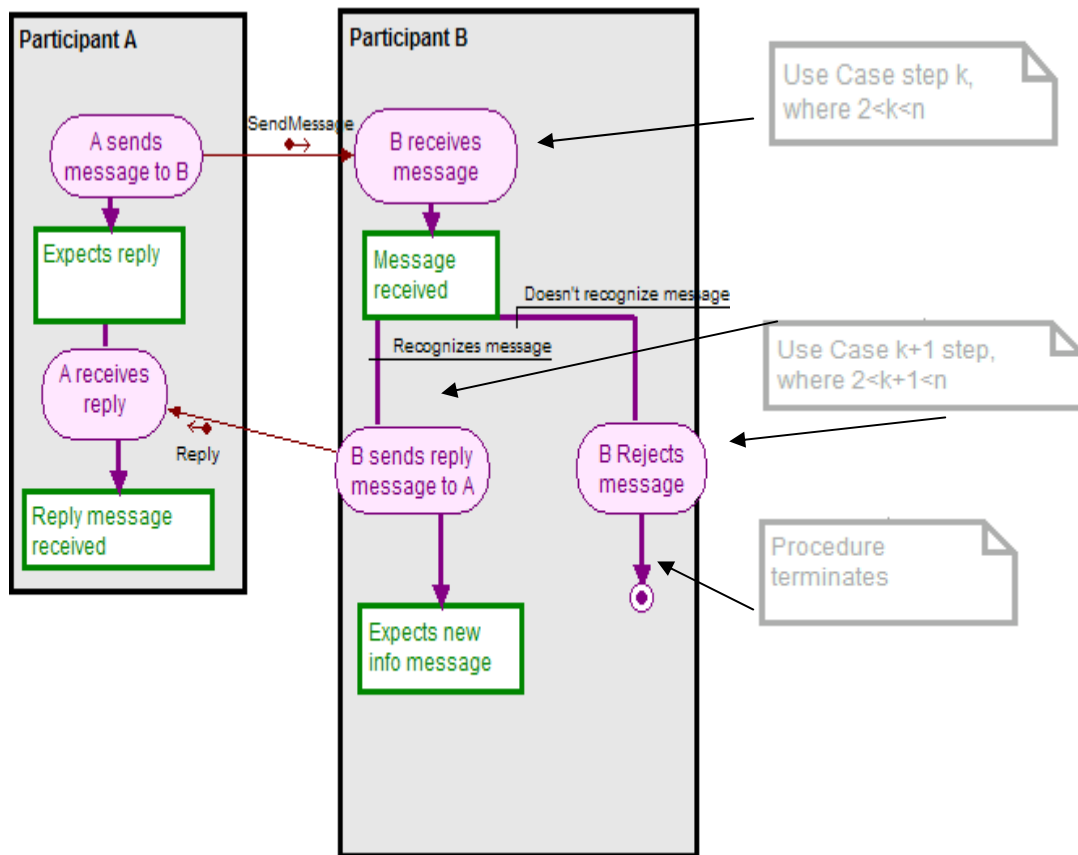


Figure 4:11: BORM aspect of Process C after Middle Step Conditional UCBTA transition

#### 4.6 Mathematical expression of the UCBTA Algorithm

As it was mentioned throughout the previous sections of the current thesis, where detailed analysis of the UCBTA Algorithm is performed the theory on which the concrete algorithm relies is that the *finite state automaton*. It was precisely stated that the analyzed algorithm is an automata or state machine based algorithm. As a consequence, by taking into consideration the fact that a variety of mathematical theorems are hidden behind several object – oriented business process approaches, it can be stated that the mathematical theory behind the UCBTA approach to requirement analysis, is the *non – deterministic finite automaton*.



It is widely known that several useful mathematical models have been derived with regard to the finite automaton or *finite state machine*. Before moving to the step of defining the UCBTA algorithm by utilizing the non – deterministic finite automaton formal mathematical definition and also the regulations which stem from the mentioned mathematical approach, it is considered critical by the author to mention and analyze the most critical parts of each of the most commonly mentioned finite automata types.

#### 4.6.1 General characteristics of the Finite State Machine

A classic form of a state diagram for a finite state machine is a directed graph with the following elements [15] [40]:

- *States*  $Q$ : a finite set of vertices normally represented by circles and labeled with unique designator symbols or words written inside them
- *Input symbols*  $\Sigma$ : a finite collection of input symbols or designators
- *Output symbols*  $Z$ : a finite collection of output symbols or designators

The output function  $\omega$  represents the mapping of input symbols into output symbols, denoted mathematically as:

$$\omega : \Sigma \times Q \rightarrow Z$$

- *Edges*  $\delta$ : represent the "transitions" between two states as caused by the input (identified by their symbols drawn on the "edges"). An edge is usually drawn as an arrow directed from the present-state toward the next-state. This mapping describes the state transitions that occur on input of a particular symbol. This is written mathematically as  $\delta : \Sigma \times Q \rightarrow Z$

- *Start state  $q_0$* : The start state  $q_0$  is usually represented by an arrow with no origin point to the state. In older texts [15], [54], the start state is not shown and must be inferred from the text. It is also stated that the start state is usually shown drawn with an arrow "pointing at it from nowhere" [66]
- *Accepting state(s)  $F$* : If used, for example for accepting automata, is the accepting state. It is usually drawn as a double circle. Sometimes the accept state(s) function as "Final" (halt, trapped) states [40]

#### 4.6.2 Formal Definition of the Non – Deterministic Finite Automaton

In the theory of computation ([22], [31], [37], [68]) a nondeterministic finite state machine or nondeterministic finite automaton (NFA) is a finite state machine where for each pair of state and input symbol there may be several possible next states. This distinguishes it from the deterministic finite automaton (DFA), where the next possible state is uniquely determined.

Apart from the above mentioned definition of the non-deterministic finite automaton, a formal mathematical expression of the analyzed state machine is required. Such a definition is considered to be indispensable due to the fact that

Two similar types of NFA's are commonly defined: the Non-Deterministic finite Automaton (NFA) and the Non-Deterministic finite Automaton (NFA) with  $\epsilon$ -moves ([66], [40]). In a similar way, many scientists define the NFA by utilizing slightly different approaches. The so called empty set of states (empty set) and moreover the empty string as an additive element of the input alphabet [71].

The ordinary NFA is defined as a 5-tuple,  $(Q, \Sigma, T, q_0, F)$ , consisting of:

- a finite set of states  $Q$
- a finite set of input symbols  $\Sigma$  (input alphabet)
- a transition function  $T : Q \times \Sigma \rightarrow P(Q)$

- an initial (or start) state  $q_0 \subseteq Q$
- a set of states  $F$  distinguished as accepting (or final) states  $F \subseteq Q$

Here,  $P(Q)$  denotes the power set of  $Q$ . The NFA with  $\varepsilon$ -moves (also sometimes called NFA-epsilon or NFA-lambda), replaces the transition function with one that allows the empty string  $\varepsilon$  as a possible input, so that one has instead

$$T : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow P(Q).$$

It can be shown that ordinary NFA and NFA with epsilon moves are equivalent, in that, given either one, one can construct the other, which recognizes the same language [71] every NFA can be expressed as DFA.

### 4.6.3 Formal Mathematical Definition of the UCBTA Non – Deterministic Finite Automaton

According to the formal definition of the non – deterministic finite automaton, the corresponding expression of the UCBTA NFA is defined as a 5-tuple,  $(Q, \Sigma, T, q_0, F)$ , consisting of:

- a finite set of states  $Q$
- a finite set of input symbols  $\Sigma$  (input alphabet)
- a transition function  $T : Q \times \Sigma \rightarrow P(Q)$ .
- an initial (or start) state  $q_0 \subseteq Q$
- a set of states  $F$  distinguished as accepting (or final) states  $F \subseteq Q$ .

The flowchart demonstrated in **Fig 4:6** includes all phases of the transformation process, throughout which the Use Case business process requirement analysis is performed and depicted by the BORM approach.

The aforementioned phases of the transformation are also depicted at the diagram of the non – deterministic finite automaton mathematical methodology of the algorithm's depiction. Considering the formal definition of the non-deterministic finite automaton, the vital part that has to be implemented, so that the completion of the mathematical delineation of the UCBTA algorithm will be completed, is the definition of the symbols of which the model is comprised.

At first the so called *states* of the finite state machine have to be defined. Considering the names provided at each step of the transformation process the definitions of the automaton steps will be as follows throughout the section that follows.

#### **4.6.3.1 UCBTA Finite state machine states**

- a) 1<sup>st</sup> State: The first part of the algorithm's finite state machine is characterized as the *input* or *start* state of the automaton. Each time that a non - deterministic finite automaton is depicted, before moving to the design of the *start* state, an arrow that comes from no input [71] is designed at first. Having utilized the concrete arrow, the starting process of the automaton is highlighted.

According to the state diagram of the described finite state machine, the start state of the concrete algorithm is the exact point for which the symbol  $q_0$  is provided. The immediate step is the provision of a certain description as far as

the concrete symbol is concerned. Taking into account the corresponding state of the flowchart the description which has to be provided to the automaton  $q_0$  symbol is the following:

$$q_0 = \textit{Logical Business Process Definition}$$

- b) 2<sup>nd</sup> State: Moving forwards in order to implement the overall construction of the UCBTA finite state machine, the second symbol defined as  $q_1$  should be characterized according to the same rules; the rule is to provide the symbol with a name or title with the help of the corresponding step throughout the UCBTA flowchart. Thus the defined symbol will be the following:

$$q_1 = \textit{Definition of the Use Case}$$

- c) 3<sup>rd</sup> State: Proceeding to the creation of UCBTA non – deterministic finite automaton, the provision of a description of the state for which the symbol  $q_2$  is defined, is the step that must be implemented by the author of the present paper. Due to the fact that the corresponding flowchart step is the creation of the *BORM General function* it is presumed that provided title to the symbol must be the following:

$$q_2 = \textit{Definition of the BORM General Function}$$

- d) 4<sup>th</sup> State: The concrete state, considering the state diagram of the finite state automaton, is provided with the symbol  $q_3$ . The corresponding depicted step at the UCBTA flowchart is characterized by the process of the *Use Case Actors* introduction. Thus the notation provided to the symbol is:

$q_3 = \textit{Introduction of Use Case Actors}$

- e) 5<sup>th</sup> State: As the path of the UCBTA finite state machine is followed the introduction of the 5<sup>th</sup> state of the process should be fulfilled as well, in a similar way as the previous states. Consequently, since the phase of the process involves the *BORM Participants*, the state for which the symbol  $q_4$  is defined, will be provided with the following notation

$q_4 = \textit{Introduction of BORM Participants}$

- f) 6<sup>th</sup> State: The following step of the UCBTA algorithm includes the definition of the so called *Initial Step of the Use Case Main Success Scenario*. As a consequence, the corresponding state of the diagram depicted at **Fig. 4:12**, which is  $q_5$  is provided with the following description:

$q_5 = \textit{Definition of the Use Case Main Success Scenario – Initial Step}$

- g) 7<sup>th</sup> State: The previous finite state automaton step is a very important part of the overall UCBTA procedure. The Initial Step of the main success scenario, according to the UCBTA rules must be part of the BORM Function. If this statement is fulfilled then another transition from the Use Case model to the BORM model occurs by defining the so called BORM Initiation. The state is symbolized as  $q_6$  as it is declared by the author of the current paper,

$q_6 = \textit{Declaration of the BORM Initiation}$

h) 8<sup>th</sup> State: The 8<sup>th</sup> state UCBTA finite state machine is characterized by the author as critical; at the present state which is denoted as  $q_7$ , the *Use Case Steps* are defined in detail. The concrete step is critical, due to the fact that the corresponding *BORM Diagram* will be constructed according to the Use Case Steps. From the Use Case Steps the *BORM States* and *Data flows* will be correspondingly defined. As a result:

$q_7 = \text{Entry of Use Case Steps is completed}$

i) 9<sup>th</sup> State: The 9<sup>th</sup> state which is defined with the symbol  $q_8$  and is depicted at the UCBTA finite state machine is characterized as a point where a decision is made. The concrete step is related to the decision whether the *Main Success Scenario is Part of the general BORM Function* or not. Consequently, from the author's standpoint the concrete decision state is the following:

$q_8 = \text{Decision whether Use Case Main Success Scenario is Part Of the BORM}$   
*general function or not*

j) 10<sup>th</sup> State: The concrete state is comprised of a situation, named as *BORM Action*. The aforementioned action is a step similar to the *Use Case Main Success Scenario*. The slight difference is that the action is utilized in order to express a general process in which the Use Case steps are all included. The concrete state, throughout which the *BORM Action is performed*, has the  $q_9$  symbol:

$q_9 = \text{BORM Action implemented}$

- k) 11<sup>th</sup> State: As it was mentioned throughout previous sections the *BORM action* is a statement that includes all the Use Case Steps. In the case that not all those steps are included the action has to be modified. Throughout the current state, a second decision has to be made. In the case that the Use Case Steps are not parts of the defined action the process terminates ( $q_s$ ). The utilized symbol is  $q_{10}$ :

$$q_{10} = \textit{BORM Action} - \textit{All Use Case steps are included}$$

- l) 12<sup>th</sup> State: The concrete state is described as critical as well. It is the first time that the end user is able to have a picture of *who* takes part in the business process. The current point of the finite state machine is a very close state to the completion of the overall process which is described by the UCBTA finite state machine:

$$q_{11} = \textit{Use Case Diagram design}$$

- m) 13<sup>th</sup> State (Output State): The UCBTA finite state automaton includes the concrete state in which *BORM data flows* and *BORM States* defined according to the use case steps. The symbol of the process which enables its mathematical delineation is  $q_{12}$ :

$$q_{12} = \textit{BORM Data flows}$$

- n) 14<sup>th</sup> State: The symbol that denotes the current state of the automaton is  $q_{13}$ . With the successful completion of the process the IT consultant is able to depict the business process flow and discuss with the end user possible gaps.



$q_{13} = BORM \text{ Diagram Designed}$

- o) 15<sup>th</sup> State(Output): The expected output of the overall process described by the UCBTA approach to business process requirement analysis, is achieved. The utilized symbol is  $q_{14}$ :

$q_{14} = BORM \text{ Result}$

- p) Terminating state: Interesting parts of the non – deterministic finite state machine, by which the UCBTA algorithm is mathematically expressed, are the two decision states; the aforementioned states are  $q_6$  and  $q_9$ . The so called *terminating state* is symbolized by the symbol  $q_s$ :

$q_s = \text{Terminating State}$

- q) The empty set of states: The UCBTA finite automaton is a non – deterministic finite automaton. Consequently, the *empty set of states*, can be included when its mathematical expression is analyzed. As it can be absorbed, the empty set of states is utilized to describe the situation where the input alphabet leads to a non – defined state. It has to be clarified by the author of the present paper, that the empty set of states differs from the *termination state*  $q_s$ . For the depiction of the empty set of states the symbol  $\{\}$  is utilized:

$\{\} = \text{Empty Set of States}$

### 4.6.3.2 Input Alphabet of the UCBTA Finite State Machine [79]

The Input symbols or the so called *Input Alphabet* of the UCBTA Finite State Automaton utilized, is the following:

$$\Sigma = \{0, 1\}$$

The concrete declaration is based on the fact that the *empty input symbol* is not necessary for the mathematical representation of the algorithm. Furthermore, the algorithm is utilized for business level process depiction; thus, the problem defined is faced by a YES/NO solution.

### 4.6.3.3 Other characteristics of the UCBTA Finite State Machine

#### 4.6.3.3.1 Mapping or transition function $T : Q \times \Sigma \rightarrow P(Q)$ .

The concrete characteristic of the UCBTA finite state machine comprises of the mapping of  $Q \times \Sigma$  into the set  $P(Q)$  or the defined by many authors  $exp(\Sigma)$  [71] of all the subsets of possible states. The empty set is also included. Taking into consideration the above mentioned mapping, and for the completion of the UCBTA mathematical definition, the following mapping results are defined:

$$T(q_0, 0) = \{\}$$

$$T(q_0, 1) = \{q_1\}$$

$$T(q_1, 0) = \{\}$$

$$T(q_1, 1) = \{q_2\}$$

$$\begin{array}{ll}
T(q_2, 0) = \{\} & T(q_2, 1) = \{q_3\} \\
T(q_3, 0) = \{\} & T(q_3, 1) = \{q_4\} \\
T(q_4, 0) = \{\} & T(q_4, 1) = \{q_5\} \\
T(q_5, 0) = \{\} & T(q_5, 1) = \{q_6\} \\
T(q_6, 0) = \{\} & T(q_6, 1) = \{q_7\} \\
T(q_7, 0) = \{\} & T(q_7, 1) = \{q_8\} \\
T(q_8, 0) = \{q_8, q_s\} & T(q_8, 1) = \{q_9\} \\
T(q_9, 0) = \{\} & T(q_9, 1) = \{q_{10}\} \\
T(q_{10}, 0) = \{q_{10}, q_s\} & T(q_{10}, 1) = \{q_{11}\} \\
T(q_{11}, 0) = \{\} & T(q_{11}, 1) = \{q_{12}\} \\
T(q_{12}, 0) = \{\} & T(q_{12}, 1) = \{q_{13}\} \\
T(q_{13}, 0) = \{\} & T(q_{13}, 1) = \{q_{14}\}
\end{array}$$

#### 4.6.3.3.2 *Initial State, final set of states and Mathematical expression of the UCBTA Finite Automaton*

The non-empty subset of the possible initial states is symbolized with  $q_0$ . At that point it has to be underlined by the author of the present paper, that the initial state is not uniquely defined; contrary to the case in which a deterministic finite state automaton is utilized, the term *initial set of states* declared. As a consequence, in the case of the UCBTA finite state machine, it can be written that:

$$q_0 \subseteq K \text{ and } K = \{q_0\}$$

Finally, regarding the set of all final states  $F$  it can be written:

$$F \subseteq K \text{ and } F = \{q_{12}, q_s\}.$$

As a result, the whole mathematical expression of the UCBTA Algorithm is the following:

$$M = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}, q_{13}, q_{14}, q_s\}, \{0, 1\}, T, \{q_0\}, \{q_{12}, q_s\}$$

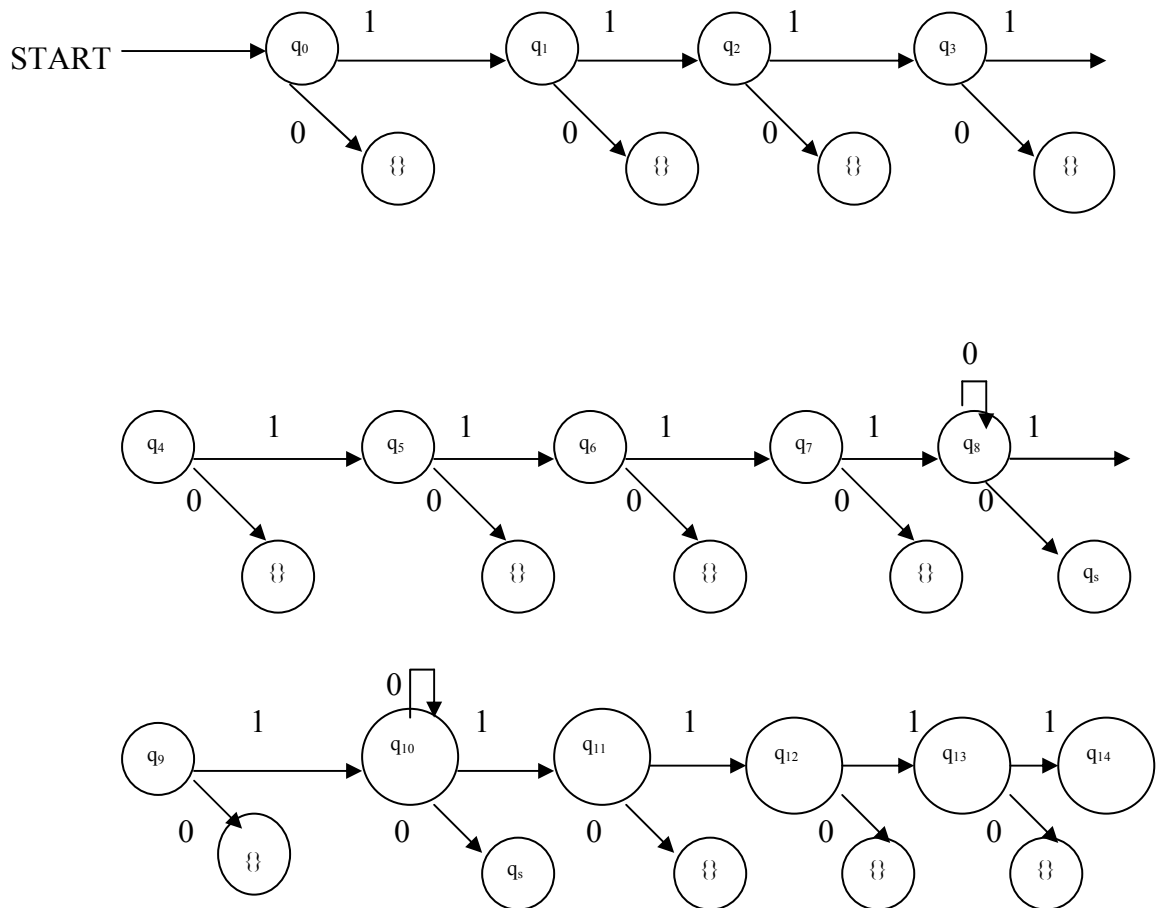


Figure 4:12 Non – deterministic finite automaton schema of the UCBTA Algorithm [79]

## ***4.7 Business Process Requirement Analysis with the UCBTA\_PROJECTS application [79]***

### **4.7.1 Primary Goal**

The transformation of the Use Case requirement analysis methodology into BORM, is analyzed throughout the previous section of the present document from a theoretical aspect. The theoretical definition of the UCBTA algorithm is composed of the following features:

- UCBTA Parts
- UCBTA Flowchart
- UCBTA Mathematical definition
- UCBTA Transition rules

With respect to the completeness of the UCBTA model an innovative software application has been created by the author, so that the aforementioned features will be fully and practically supported.

The current section is comprised of a thorough delineation of the UCBTA\_PROJECTS application, which is designed and created by the author [79] in order to secure a plain, rapid and efficient transition from one model to the other. Secure transition from the Use Case method to BORM, is implied by the elimination of data loss throughout the transformation procedure.

UCBTA\_PROJECTS is utterly based on the UCBTA transition rules according to which the Use Case steps and sub – steps are directly mapped to the activities, states

and data flows of the Process Participant interaction model – defined as Object Relation Diagram as well – in BORM. As a consequence the target output of the application is the BORM model automatically derived from the basic process definition; the named process is automatically translated into the corresponding Use Case which is further utilized for the BORM output derivation.

All Use Case features such as, Actors, Main Success Scenario Steps and Main Success Scenario Sub – Steps are determined in a specific part of the application and the BORM model is automatically generated with one simple button click. The programming language utilized for the UCBTA\_PROJECTS development is Microsoft Visual Basic 6.0 [75] and the environment in which the application is developed and designed is the Microsoft Visual Studio 6.0. Furthermore, the data which is created through the UCBTA\_PROJECTS windows is saved in a Microsoft Access 2003 database file.

From all the aforementioned details with regard to the constructed application it is concluded that:

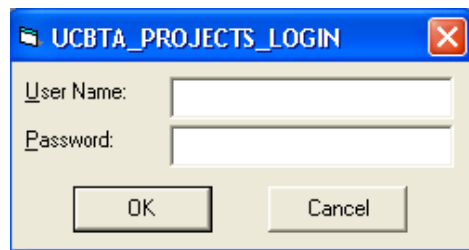
*The main goal of the creation of the defined application, which is entitled as UCBTA\_PROJECTS, is to provide the system analysts with an efficient and easy to user interface in order to implement effective business process requirement analysis by utilizing the UCBTA algorithmic methodology.*

#### **4.7.2 Delineation of the UCBTA\_PROJECTS interface**

Throughout the current part of the section, a detailed description of the UCBTA\_PROJECTS application is performed. Microsoft Visual Studio 6.0 was the development environment utilized in order to create the forms from which the windows of the application stem.

#### 4.7.2.1 UCBTA\_PROJECTS LOGIN Window

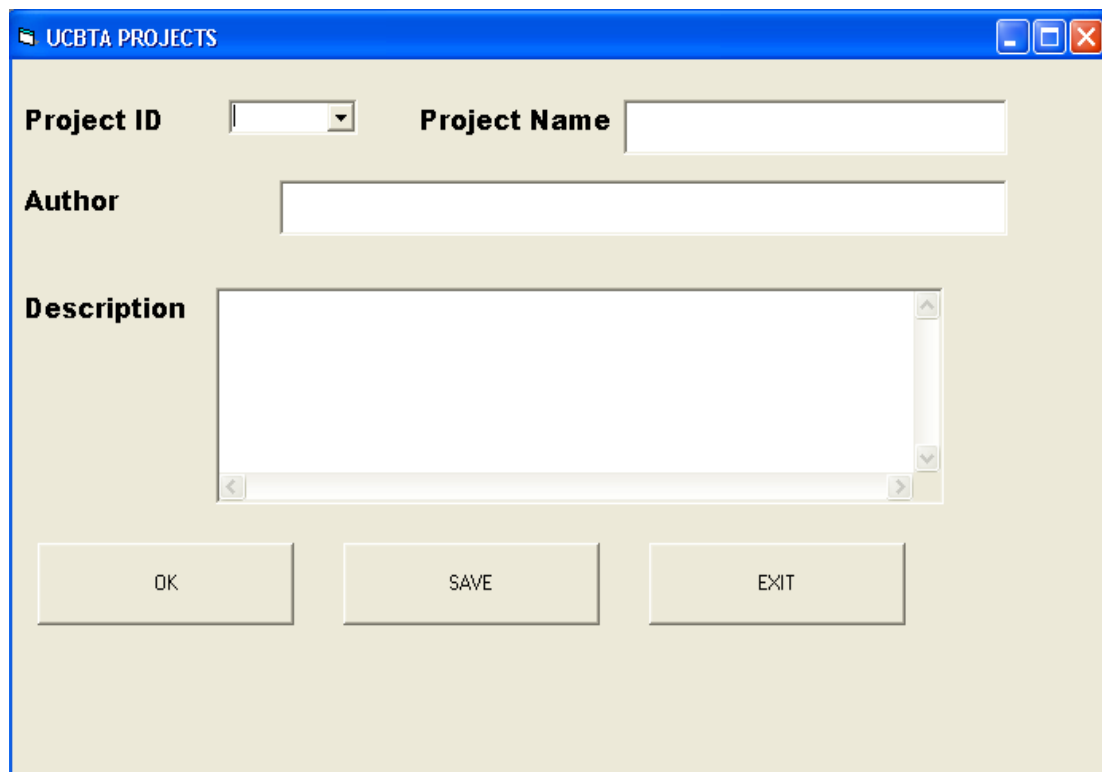
When the UCBTA\_PROJECTS application is initiated a Login window (*Fig. 4:13*) emerges and the task for which the user is prompted is to provide the system with a username and a password. In the case that the appropriate data is provided, a Microsoft Access 2003 database file opens and the system is connected with the specific .mdb file. The database file is utilized for storing the Project data.



**Figure 4:13: The UCBTA\_PROJECTS\_LOGIN window**

#### 4.7.2.2 UCBTA\_PROJECTS Window

After entering the valid username and password for utilizing the application, a new window is available to the user for entering new data. The window's caption is the UCBTA PROJECTS (*Fig. 4:14*). The fields that are included in the current window are the Project ID, Project Name, Author and Description field where some short paragraph with regard to the specific project is placed.



**Figure 4:14: The UCBTA\_PROJECTS window**

It is obvious that the data which is stored with respect to each project is based on the logic that each project has a unique project ID (*Fig 4:15*) and that the one specified business process and the one corresponding Use Case is related to the concrete project. The project author of course can be the same in different projects.

When inserting the data which is related to a project, the defined information is saved by clicking the SAVE button. In the case that the user wishes to finish the performed task the EXIT button will be utilized; finally to proceed with the business process requirement analysis and in order to reach the desired window of the application where the BORM data is available for constructing the corresponding Business Object Relationship Diagram, the user should press the OK button.



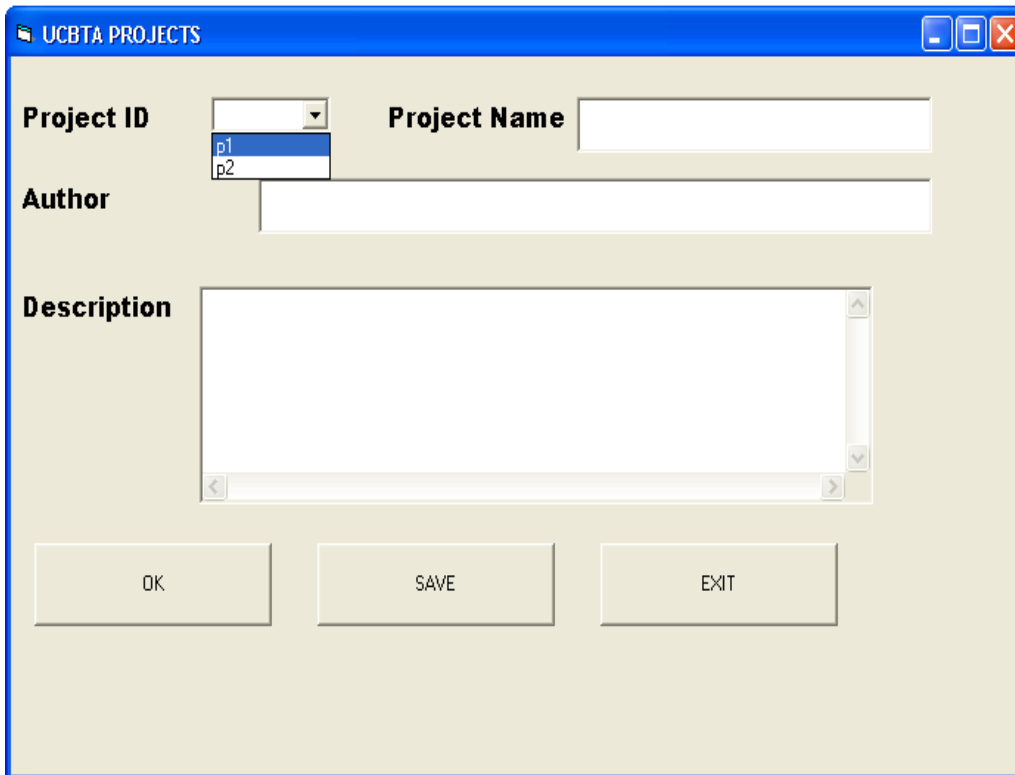


Figure 4:15: The UCBTA\_PROJECTS window with the Project\_ID combo box

#### 4.7.2.3 Use Case Data Model Window

By pressing the OK button of the UCBTA PROJECTS window the third window is available to the author of the project. The caption of the new form is entitled as Use Case Data Model. The concrete form is designed in order to enter the Use Case data. The window is composed of two important frames (*Fig 4:16*). The first frame is the Project Properties frame and the second is the Use Case – Main Success Scenario frame. The Use Case main success scenario is ideally delineated via nine Use Case steps. The defined steps are presented in the form through 9 short sub frames.

The screenshot displays the 'Use Case Data Model' application window. It is divided into two main sections: 'Project Properties' and 'Use Case - Main Success Scenario'.

**Project Properties:** This section contains five input fields: 'Project Name', 'Process', 'Use Case', 'BORM General Function', and 'BORM Action'. To the right of these fields are three buttons: 'Show BORM', 'Save Process Data', and 'Exit'.

**Use Case - Main Success Scenario:** This section is used to define the steps of a use case. It starts with an 'Initial Step' and is followed by steps 2 through 11, and a 'Final Step'. Each step is represented by a table with three columns: 'Actor A', 'Action', and 'Actor B'. Below each table is a 'Properties...' button. To the left of the steps, there is a section for 'Insert Actors' with a list box and buttons for 'Add Actor', 'Clear Actor Lists', 'Save Actor List', and 'Delete Actor'.

**Figure 4:16: The Use Case Data Model window with all Main Success Scenario steps included**

The Project Properties frame is comprised of five fields. The Project Name field is automatically stored when the OK button of the previous window is pressed by the user. The name textbox is filled with the same text as the corresponding field of the previous window form.

When the Process field is filled it shall be noticed the Use Case field is simultaneously and automatically completed since according to the UCBTA algorithm the Use Case is defined by the same notation as the Process name.

Other available fields to the user are the BORM General Function and the BORM action fields. The data placed in those fields is found in the final window of the application (**Fig.4:18**)

Other important features of the Use Case model are the so called Use Case Actors. The user is able to insert actors in the corresponding textbox of the form and edit Actor data by adding and deleting actors with the appropriate buttons. When the SAVE ACTOR LIST button is pressed the actor list of the project is automatically placed in all combo boxes of each sub frame.

On the other hand when the CLEAR ACTOR LIST button is pressed the user is able to remove the inserted actor data from all combo boxes. Another important feature of the present window is the Action textbox which is included in all sub frames (Use Case Steps). The concrete textbox is filled with a verbal phrase i.e. *sends message to*. As a consequence when Actor A is selected from the combo list, an Action phrase is placed in the Action textbox and additionally when Actor B is chosen the data of the Use Case Step is currently placed.

In order to complete the Use Case step data insertion the Properties button should be pressed and the Use Case Step Details window emerges.(**Fig 4:17**). The new window form will be further analyzed in the following paragraph.

Furthermore, when the SHOW BORM button is pressed the final window form (**Fig 4:18**) in which the BORM model is included emerges. The form shall be presented in following paragraph as well.

#### **4.7.2.4 Use Case Step Details window**

The Use Case Step Details window is activated when the Properties button of a Main Success Scenario step sub frame is pressed by the user. The Sub – Steps are parts of the Use Case step.

Actor A	Action	Actor B	Activity	State
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>

**Figure 4:17 Use Case Step Details window with sub steps of each Use Case Main Success Scenario step**

The combo boxes that are included in the parent form are filled when the SAVE ACTOR LIST button is utilized from the previous form. Action textboxes are filled in by the user. In the *Step* field the user is informed for which step the Use Case sub – steps are defined. The final step for the completion of the sub – step data storage process is performed when the selection of the sub – step type is done. A sub – step can be either an *activity* or a state. By selecting the sub – step type and after pressing the SAVE BORM DATA button the information is passed to the final BORM window with the corresponding capital letter (A for Activity and S for a State type) inside the each textbox which is related to the appropriate sub step textbox of the window.

#### **4.7.2.5 BORM (Process Participant Interaction Model) window**

The present window comprises of the basic tool by which the Object Relation Diagram will be constructed. The main part of the BORM window is comprised of a special frame which is entitled as *Business Process Workflow*.

The screenshot displays the BORM (Process Participant Interaction Model) window. At the top, there is a blue header with the text "BORM (Process Participant Interaction Model)". Below the header, there are several input fields: "Initiation", "Action", "Result", and "BORM General Function". To the right of these fields is a "Participants" section with a large empty box. The main area of the window is titled "Business Process Workflow" and contains a grid of 12 steps, arranged in three rows of four. The steps are labeled as follows:
 

- Row 1: Initial Step, Step 4, Step 7, Step 10
- Row 2: Step 2, Step 5, Step 8, Step 11
- Row 3: Step 3, Step 6, Step 9, Final Step

 Each step is represented by a large rectangular box for the main step description and a smaller box for sub-steps. At the bottom right of the window, there is a button labeled "VALIDATE BORM MODEL".

**Figure 4:18: The BORM (Process Participant Interaction Model) window – Business Process Workflow is included**

The specific frame includes all data about the Use Case steps which are now translated by the system as BORM activities, states and data flows. The corresponding Use Case Steps are depicted with a larger textbox so that they can be distinguished by the sub – steps. The sub – steps are drawn with a shorter text box (**Fig 4:18**). The shortest textbox near the sub – steps is utilized in order to inform the user of the sub – step status which can be either *activity* (A) or *state* (S).

The *Initiation* field on is automatically filled in by the Initial Step of the Use Case Main Success Scenario sub frame of the Use Case Data Model window by pressing the Properties button of the corresponding frame. In the same way the result field is filled in. The difference is that the BORM result is equal to the final step of the Use Case Model. It should be noticed by the analyst who is the user of the application and wishes to perform business process requirement analysis with the UCBTA\_PROJECTS tool that the text inserted into the Initiation field is the same as the text placed into Initial Step field. Moreover, the text placed into the Result field is the same as the text inside the last step of the model. The last step can be either the Final Step or some previous step if the number of steps is less than nine.

Finally the *Participants* list box is filled in by the Actors list box of the Use Case Data Model window and by pressing the SAVE ACTOR LISTS button. The *Validate BORM Model* button is utilized in order to control whether the BORM Function and the BORM action is valid. By pressing the concrete button the user is asked by the system if the Function and Action Data is valid. In the case that the data is not valid it has to be altered by the user so that the entire BORM model will be valid.

## **5 Applied UCBTA business process requirement analysis – Case Study of critical Greenhouse Integrated Pest Management (IPM) Processes**

The current section of the dissertation thesis comprises of a demonstrative methodology inspired by the author in order to apply the UCTBA algorithmic approach to business process requirement analysis. As it was mentioned in the beginning of the current work, and precisely throughout the introduction, there exist many interesting business areas where little effort has already been made by experts in order to introduce and utilize technological tools in order to implement business processes in an automated, rapid and pattern oriented manner. In other words, neither many software applications nor many entire Information Systems have been integrated so as to enable the automated performance of the business processes of which the aforementioned systems are comprised.

A scientific field of great interest as far as information technology techniques is concerned and for which, according to agricultural experts, growers, stakeholders and IT experts, little or zero work has been done so far in terms of business process engineering is the **Greenhouse Integrated Pest Management practices**. From the standpoint of many scientists, including the author of the current dissertation thesis who has both IT and Agricultural scientific background, Integrated Pest Management (IPM) is a promising area as far as business process requirement analysis is concerned, with a lot of possibilities to apply pattern based methodologies in order to design the model of these processes, so that they will be utterly absorbed by farmers and greenhouse growers.

The reason that some prepared applications are demonstrated in the beginning of the present work, is the fact that what is stated by the scientists and university teams who created the concrete applications, is that the most difficult part of the system integration is the inability of IT experts to communicate with business domain experts (growers and agronomists), since the former utilize computer oriented techniques for demonstrating business processes that cannot be absorbed by the latter. In that case,

when the system or the integrated software is delivered to the end users a huge amount of bugs is discovered and in many cases a lot of bug fixing working man days have to be spent by IT experts in order to deliver the desired application to the hand of the system users.

The UCBTA algorithm, as it was introduced as a *new and pattern based* approach to implementing business process requirement analysis, is a method proposed so that time loss caused by irritating bug fixing will be prevented. It is proposed as a new method for requirement analysis in the case of Greenhouse Integrated Pest Management practices, and for the efficient modeling of the business processes that are the skeleton these practices.

The detailed Case Study as far as the derivation of IPM business process requirement analysis is performed with the UCBTA utilization, will be delineated throughout the sub sections that follow.

### ***5.1 What is Integrated Pest Management?***

Integrated pest management (IPM) is a systems approach that provides an ecologically-based solution to pest control problems. IPM is defined by many experts *as a sustainable approach to managing pests that combines biological, cultural, physical, and chemical tools in a way that minimizes economic, health, and environmental risks*. It is a proven approach that balances economic, environmental, and health objectives [26]. Alternative definitions regarding the Integrated Pest Management issue are provided by many experts. IPM is also defined as a pest management strategy that focuses on long – term prevention or suppression of pest problems with minimum impact on human health, the environment and non target organisms [33] A simpler delineation of the IPM issue is provided by Greer L. and Diver S., who are NCAT Agricultural Specialists [8].According to their statements *IPM is an important tool for the management of pests. Its primary goal is the pest control optimization in an economically and ecologically sound way.*



Many authors in the past had an aspect of IPM that focused mainly on “integrated control,” a strategy involving primarily chemical and biological control ([19] [21], [67], [70], [35]). While the latter two groups of authors broadened their scope of integrated control to include cultural and other means of physical pest control, their discussions still centered on classical chemical and biological controls ([70], [35]).

IPM has expanded its scope over the past 40 years to encompass a variety of applications in rural and urban settings. This expansion has resulted in a scientific exploration to discover new tools for maintaining pest populations at acceptable levels while sustaining an ecological balance. In addition to this expansion, IPM has become a target for change.

IPM practitioners first realized the need for this change as public concern over pesticide issues came to the foreground. This concern has blossomed with the advent of additional pest control and regulatory issues. Resistance management, worker protection standards, water quality concerns, and food quality protection represent only a portion of the issues confronting IPM implementation nowadays.

## ***5.2 What does effective Integrated Pest Management entail?***

In order to implement an effective IPM program today, many changes in current decision-making processes may be required. Such programs must merge ecology, economics, and environmental concerns with practical management concepts.

Growers must recognize that their decisions have consequences that reach far beyond the immediate time and location of their operation. They need to incorporate information gained from the use of key tools such as crop monitoring, and good record keeping, making sound management decisions [26].

Further to the above mentioned statements with regard to the proper definition of IPM, the effectiveness of such a program comes to provide the science with a

complete definition taking into consideration management determination practices as well; more precisely it is stated that IPM involves the integration of cultural, physical biological and chemical practices to grow crops with minimal use of pesticides. Monitoring or scouting, sampling and record keeping are used to determine when control options are essential in order to preserve pests below an economically damaging threshold.

From the aforementioned statements which are provided and supported by many agricultural experts, successful, effective and efficient Integrated Pest Management strategy is always followed by practices that reassure the economical and ecological utilization of resources; resources can be divided to human resources and material resources.

The ideal human resources exploitation depends on the so called labour hour's measurement. Moreover, it has to be mentioned that throughout some sub processes that comprise of the entire IPM procedure, human presence and action play an actual and crucial part in its success. For instance, record keeping and scouting process include a step where the name and the working hours of the performing person is recorded; consequently the worker's or farmer's or expert's work, operation and knowledge can be judged and improved in the case when results are not satisfactory.

Having taken into consideration all the above mentioned facts with regard to the entire Greenhouse IPM procedure, it is concluded that successful and effective IPM depends mainly on the action performed at the time when a *threshold value* of the pest presence is detected and with the utilization of a certain *pesticide quantity* in order to implement ecological and economical IPM.

From the author's standpoint, the construction of a Greenhouse Information System in which Integrated Pest Management processes would be based on efficiently integrated computer applications, is the critical success factor to the successfulness of IPM practices. As it was stated, little or no IT scientific work has been done with respect to IPM.

Detailed pattern based business oriented requirement analysis for computerised IPM, that will entail all the necessary business processes defined and performed in an automated manner, is the research subject of the current dissertation thesis.

The target result of the current document will be the delineation of selected Greenhouse IPM business processes by utilizing the Use Case approach in conjunction with BORM method and the Object Relation Diagrams. The Object Relation Diagrams will comprise of a visual representation of the selected IPM processes on a business level, and the transformation path from the Use Case analysis of the processes to the BORM methodology and the schematic depiction in ORD will be indicated by the UCBTA algorithm which was analysed in detail throughout the previous chapter of the thesis.

The important steps that have to be followed for the detailed business process requirement analysis of the concrete processes are:

- Feasibility study
- UCBTA construction with unambiguous Use Case definition
- ORD Diagram design

Moreover, the processes for which business process to requirement analysis will be performed are the following:

- *Daily Scouting (Monitoring) record keeping*
- *Weekly Scouting (Monitoring) record keeping*
- *Evaluation of pesticide's effectiveness*

### ***5.3 UCBTA - Case Study of the Greenhouse Integrated Pest Management practices***

Throughout the previous chapter of the current thesis, a short practical example which aimed at the understanding of the UCBTA algorithm functionality with regard to

business requirement analysis was performed, and for the entire analytical demonstration all parts of the algorithm have been recorded one by one so that the so called step – by – step transformation from the Use Case methodology to the BORM approach would be utterly achieved.

The concrete example was the greenhouse humidity control process. It has to be stated though that the entire humidity control procedure includes parts that were excluded, since the goal was a simple and short demonstration of the UCBTA functionality.

Throughout the final part of the current thesis, a UCBTA analytical and thoroughly described Case Study will be implemented. The selected processes for which detailed business process requirement analysis will be performed, are considered to be crucial and have a precise and immediate impact on the Greenhouse production and environment.

### **5.3.1 Scouting (Monitoring) Record Keeping**

The primary goals of monitoring, else called scouting, are to locate and identify insect, mite and disease problems, and to observe changes in the severity of infestation. [69]. A scouting procedure must be as routine as possible; according to many agricultural experts, whenever a scouting method is utilized it should be intensive enough so as to reassure the program's success as far as pests' elimination is concerned.

Scouting usually starts from doorway where the most dangerous location for a pest infestation is pointed. Moreover special attention should be paid to plants around any openings in the greenhouse, especially those plants on the outside rows of benches.

Scouting (monitoring) benefits mainly stem from the fact that the symptoms of pest damage and the numbers of the pests themselves can increase very rapidly. If problems are not detected early, crops may be severely damaged and damaging

options will be very limited. Regular scouting enables the grower gain the following profits [30].

1. Prevent problems or reduce the amount of damage and the cost of control by providing early warnings that pest problems are developing
2. Determine the specific cause and severity of the problem
3. Identify the locations that require immediate and absolute treatment, so as to avoid unnecessary control actions
4. Determine the most effective and economical timing and method of treatment
5. Use slower-acting methods that are more environmentally friendly and much safer for workers
6. Evaluate control efficacy

An effective and efficient scouting or monitoring procedure is characterised by the proper utilization of technology. From the agronomists' viewpoint, and according to many scientific references, effective and efficient Greenhouse IPM scouting procedure is in correlation with detailed and carefully planned *monitoring record keeping*.

The question pointed is which is the exact technological involvement in the procedure of keeping records of monitoring observation? The answer is provided by many IPM practitioners, who are either farmers or agronomists but who both support that hand based record keeping is more likely to fall into traps and finally fail rather than automated and computer oriented record keeping of scouting observations.

Careful attention must be paid to the claim that without the proper records that mainly should be kept in computer [8] for better and more precise diagnostic results scouting will be ineffective. Managers who are attempting to perform pest diagnosis without the utilization of proper records are at disadvantage and are will overlook potential causes of the problem. [69]

Effective and adequate monitoring is comprised of the following methodologies:

- Incoming plant material monitoring
- Yellow sticky traps scouting
- Indicator plant monitoring
- Random or individual plant scouting

Whichever of the aforementioned scouting methodologies is chosen, detailed record keeping on a daily basis and weekly data observation must be performed in terms of pest population. Pest population data is recorded daily, and weekly computer based summaries, enables the growers and the agricultural experts to perform the corresponding and indispensable action against the pest observed. Pest population data records are comprised of the following elements

- Number of pests revealed after the scouting (monitoring) methodology is applied
- The life stage of the revealed pest
- The type of the pest found during the scouting process
- The part of the plant inspected ( in the case that the monitoring method utilized is based on individual plant inspection or the indicator plant scouting)

The following section of the current dissertation thesis include the utilization of the UCBTA algorithm as a proposal to perform pattern-based and extensive business process requirement analysis of an automated and computer – derived approach to accomplishing proper record keeping as far as any of the above stated scouting methods is concerned.

### 5.3.1.1 Daily scouting (monitoring) record keeping

Regular observation is the cornerstone of IPM. Observation is broken into two steps; first inspection and second identification. [11]. Visual inspection, insect and spore traps, and other measurement methods and monitoring tools are used to monitor pest levels. Accurate pest identification is critical to a successful IPM program. Record-keeping is essential, as is a thorough knowledge of the behaviour and reproductive cycles of target pests.

Without proper records, scouting will be ineffective. Since insects are cold-blooded, their physical development is dependent on the temperature of their environment. Many insects have had their development cycles modelled in terms of degree days. Monitor the degree days of an environment to determine when is the optimal time for a specific insect's outbreak.

From all the above mentioned scouting knowledge, it should be noted that computerised monitoring is inevitable element of an efficient and utterly computer – oriented greenhouse IPM. Thus, the *process feasibility* is underlined by the author since automated and IT process based scouting is regarded by the author as essential and it is related to detailed and analytical record keeping.

Having analyzed the feasibility of the current business process, the forthcoming author's aim is to define the process, the Use Case name provision as far as the process is concerned and finally construct an analytical business process model throughout the *Use Case To BORM Transformation Algorithm* procedure; the prerequisite for deriving such a model, as it was underlined throughout the current dissertation thesis is the detailed business process requirement analysis which shall be carried out with the utilization of an unambiguous Use Case definition and its step-by-step transformation to BORM according to the rules stated by the author, with regard to the defined algorithmic business process requirement analysis approach.

Considering the UCBTA steps of the transformation of the Use Case Model to the BORM business process requirement analysis model, and the final process representation with the ORD (Object Relation Diagram), the complete UCBTA based requirement analysis is delineated with a thorough UCBTA step description which is comprised of the following:

**Input Part:** The name of the delineated process is provided throughout this initial part of the algorithm. The concrete analyzed procedure is characterized as:

*“Daily Record Keeping for Scouting (Monitoring) purposes”*

**Use Case definition:** The demanded Use Case is related to and defined according to the analyzed process. The parent Use case is entitled as:

*“Performing Daily Scouting Record Keeping”*

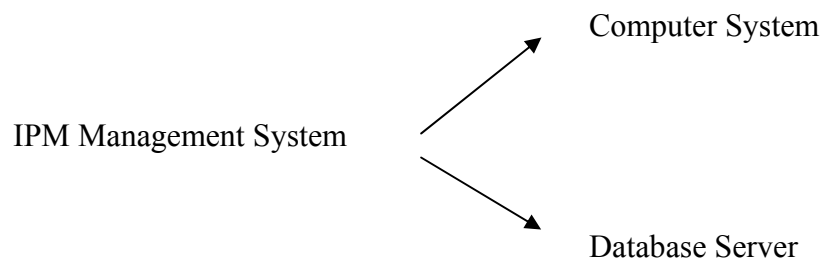
**BORM General Function:** The demanded BORM general function, which is the starting point of the changeover between the two models, is provided with the characterization *“Economic IPM Administration”*. What should be noticed at this point is that regarding the mathematical model of the algorithm, the Use Case according to the author of the present work, must be *a part of* the BORM general function; indeed *Daily record keeping* is part of the overall *Economic IPM administration*.

**Use Case Actors’ Definition:** The concrete process requires the existence of two main actors:

- IPM Management System
- Grower

The above mentioned IPM Management System is comprised of the following parts:





**BORM Participants' Definition:** The notation utilized for the BORM Participants, with regard to the UCBTA algorithm, is exactly the same as the notation used for the Actors' definition; thus, in the case of the current described process, the participants are:

- Computer System
- Database Server
- Grower

What should be clarified at that point is the role and also a general part or even a short delineation of the Use Case Actors, and consequently the so called BORM participants which are identical according to the UCBTA rules.

Computer system: The system in which a Greenhouse IPM is installed and is utilized by the grower.

Database Server: A mainframe in which huge amount of data is stored, and in which an application server is installed for communicating with the IPM interface of the computer system.

Grower: An agronomist, a greenhouse owner who is Greenhouse IPM domain expert and is able to absorb IPM business processes.

**Use Case Main Success Scenario – Initial Step:** The current process is initialized, when Grower considers daily scouting for the control of pest population essential, and must start keeping the necessary records. Thus, the *initial step* of the specific main success scenario will be the following:

*Grower selects cultivation type in order to perform daily scouting, control pest population, and keep the necessary records.*

**BORM Initiation:** As it was mentioned in the previous chapter of the dissertation thesis, this part of the procedure stems from the *Initial Step* of the Use Case *main success scenario*. As a consequence, the same notation utilized in terms of the main success scenario definition can be utilized in this transformation step as well.

In the case of the parent business process for which analytical requirement analysis is implemented, the BORM Initiation is entitled as:

*Grower selects cultivation type in order to perform daily scouting, control pest population, and to keep the necessary records.*

**Use Case Steps Definition:** The Use case steps and their sub steps which concern the current Greenhouse IPM business process and that comprise the *Main Success Scenario* are also recorded for the needs of the UCBTA algorithmic approach; the specified steps are the following:

A) Main Success Scenario

- 1) Grower selects cultivation type
- 2) Computer System demands task
- 3) Grower selects daily scouting (monitoring)
- 4) Computer system demands time period
- 5) Grower stores time period data to the system
- 6) Computer System requests area sector definition
- 7) Grower selects areas where scouting results revealed pest population on marked plants
- 8) Computer System demands pest scouting method and pest population data
- 9) Grower selects scouting method and stores pest population data

- 10) Computer System sends message to the server (Database Server)
- 11) Database Server produces daily monitoring report and action threshold message
- 12) Computer System Displays message to the grower for the pest status and the action threshold

B) Sub steps

- 1a) Grower awaits response
- 1b) Selection is obtained
- 1c) Computer System receives cultivation type selection command.
- 2a) Computer System is expecting new selection
- 2b) Demand task is transmitted
- 2c) Grower receives task demand
- 3a) Grower awaits daily scouting screen
- 3b) Computer System obtains daily scouting selection
- 3c) Selection received
- 4a) Computer System obtains daily scouting selection
- 4b) Computer System awaits data
- 4c) Demand sent
- 5a) Grower expects new request
- 5b) Computer System receives time period data
- 5c) Time period data is stored
- 6a) Grower obtains area sector definition demand
- 6b) Area sector demand is obtained
- 6c) Area sector definition is expected
- 7a) Grower expects new demand
- 7b) Computer System receives area sector data
- 7c) Area sector data is received
- 8a) Computer System expects scouting method and pest data results
- 8b) Grower receives pest monitoring method and pest population demand
- 8c) Demand is received
- 9a) Computer System obtains expected results
- 10a) Computer System expects server report

- 10b) Database Server receives daily scouting data by the Computer System
- 10c) Daily monitoring data is obtained
- 11a) Computer System receives daily scouting report and
- 12a) Grower receives daily scouting report and system's message for action threshold and considers Integrated Pest Management (IPM) strategy

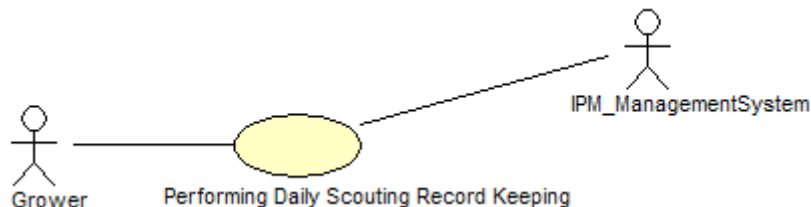
**Main success scenario, as a subset of the BORM General Function:** The concrete algorithmic part presupposes the existence of the main success scenario in the inner part of the BORM general Function; the aforementioned scenario comprises of a standard subset of the BORM General Function; consequently the model transformation algorithmic procedure can be normally carried out.

**BORM Action definition:** The action defined with regard to the delineated process is the following:

*Grower performs scouting and data record keeping of the necessary values.*

It should be also noticed that the Use Case steps are included in the defined BORM action and as a result the action should not be modified.

**Use Case Diagram:** Having completed a significant part of the described business process, the transition to the design of the Use Case diagram is considered to be critical process depiction tool, before the derivation of the output and the business process diagram. The Use Case Diagram which is related to the business process requirement analysis of the *Daily Record Keeping for Monitoring Purposes* process, will be designed according to **Fig. 5:1**



**Figure 5:1: Use Case Diagram for the “Daily Record Keeping for Scouting Purposes” Process**

**Defining the BORM Data flows:** According to the UCBTA algorithmic transformation steps and rules, the BORM Data flows are related to the analyzed Use Case main success scenario. Data flows express the communication between participants in BORM. Communication is achieved via connection of activities.

**Design Object Relation Diagram :** After completing the step of the Data Flow Definition, , provided that data flows are carefully stated with the co-operation of IT experts, stakeholders and end-users and under the condition that with regard to the initial analysis level and without taking into account any software orientation, the user requirements are met. In *Fig. 5:2* the currently delineated and oriented to the Greenhouse IPM process defined as “*Daily Record Keeping for Scouting (Monitoring) purposes*” is depicted.

**UCBTA Output: BORM Result :** the algorithmic output is the actual BORM result; it is derived from the transformation of the Use Case final step to the BORM final activity according to which the entire process terminates. Consequently the output of the currently analyzed business process is provided with the following title:

*Grower receives daily scouting report and system’s message for action threshold and considers Integrated Pest Management (IPM) strategy*

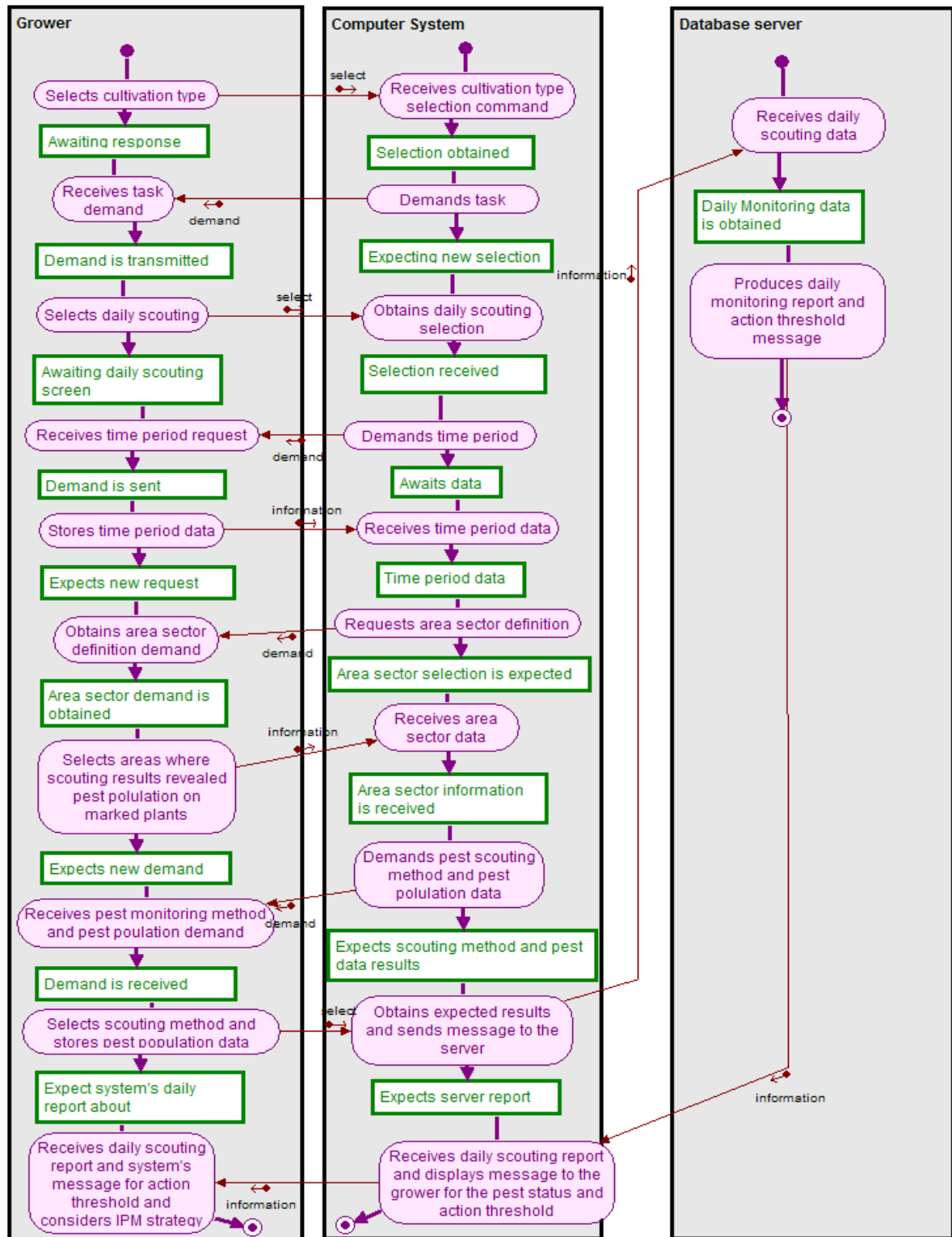


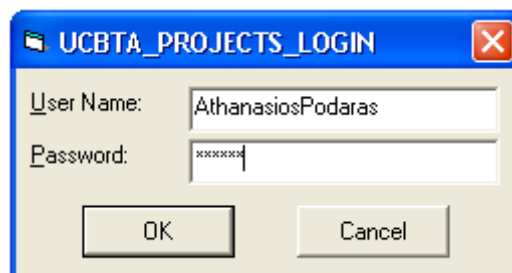
Figure 5:2 : Object Relation diagram based on the Daily Record Keeping for Monitoring Process

## **Deriving business process requirement analysis results with the UCBTA PROJECTS Application:**

The UCBTA\_PROJECTS application is designed and created by the author for implementing automatic derivation of the BORM model after the recording of all the Use Case Main Success Scenario Steps.

Its target goal is to provide the system analysts with the possibility to perform business process requirement analysis via a friendly environment and enable the end users to absorb and easily correct the steps according to which the delineated process will be performed.

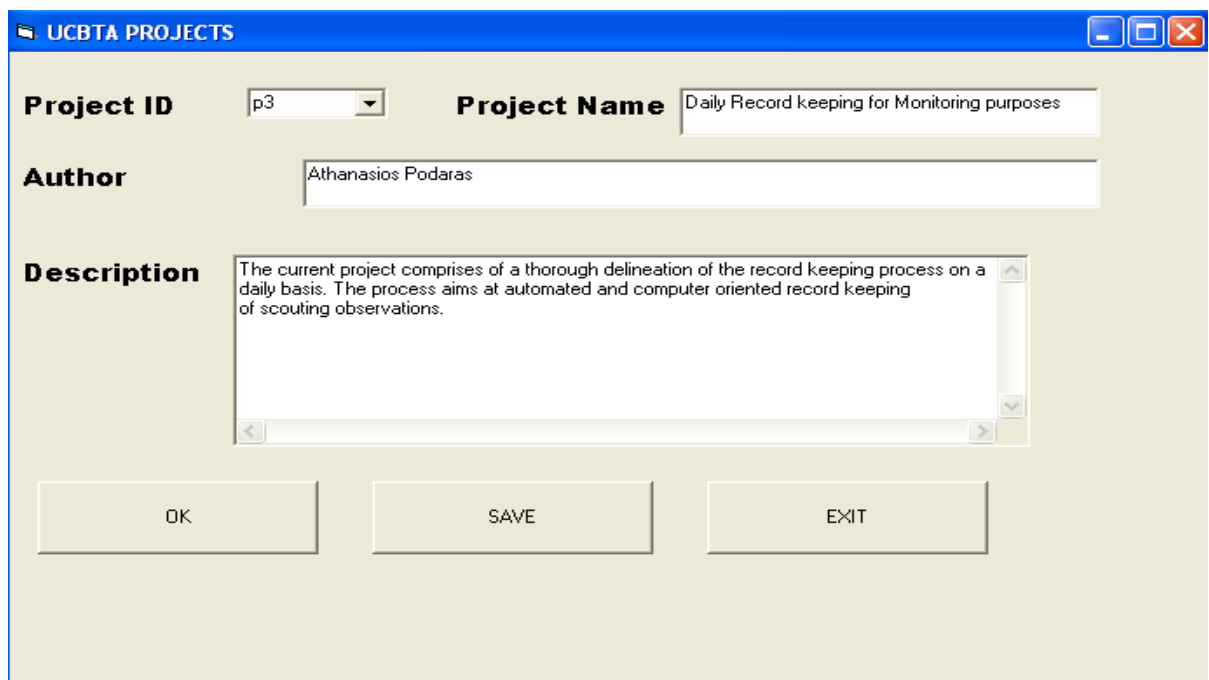
Further to the above statement and according to the description of the application throughout the previous chapter of the current thesis, the procedure of utilizing the UCBTA\_PROOJECTS environment is initiated by entering the appropriate username and password via the Login Window (*Fig. 5:3*) of the application.



**Figure 5:3: Entering username and password to the Login Window**

The step that follows the username and password data entry is the process data entry. The initial data is entered via the UCBTA PROJECTS window. The initial data of the process is comprised of the Project ID, Project Name, Author and Description data fields.

The data which is related to the *Daily Record Keeping for Monitoring Purposes* Greenhouse IPM business process is entered to the corresponding fields of the UCBTA PROJECTS window (**Fig. 5:4**)



The screenshot shows a software window titled "UCBTA PROJECTS". It contains the following fields and controls:

- Project ID:** A dropdown menu with "p3" selected.
- Project Name:** A text box containing "Daily Record keeping for Monitoring purposes".
- Author:** A text box containing "Athanasios Podaras".
- Description:** A text area containing the text: "The current project comprises of a thorough delineation of the record keeping process on a daily basis. The process aims at automated and computer oriented record keeping of scouting observations."
- Buttons:** Three buttons labeled "OK", "SAVE", and "EXIT" are positioned at the bottom of the window.

**Figure 5:4: Entering Initial Data to the UCBTA PROJECTS Window – Daily scouting and record keeping process**

Further to the *Daily scouting record keeping process* data entry the Use Case Main Success Scenario Steps are recorded one by one to the sub – frames of the Use Case Main Success Scenario frame of the Use Case Data Model window. (**Fig. 5:5**)



The screenshot displays the 'Use Case Data Model' application window. At the top, the title bar reads 'Use Case Data Model'. Below it, the 'Project Properties' section contains several input fields: 'Project Name' (Daily Record keeping for Monitoring purposes), 'Process' (Performing Daily Scouting Record Keeping), 'Use Case' (Performing Daily Scouting Record Keeping), 'BORM General Function' (Economic IPM Administration), and 'BORM Action' (Grower performs scouting and data record keeping of th...). To the right of these fields are three buttons: 'Show BORM', 'Save Process Data', and 'Exit'.

The main section is titled 'Use Case - Main Success Scenario'. It features an 'Insert Actors' list on the left containing 'Grower', 'Computer System', and 'Database Server', with buttons for 'Add Actor', 'Clear Actor Lists', 'Save Actor List', and 'Delete Actor'. The scenario is composed of 11 steps, each with a table for Actor A, Action, and Actor B, and a 'Properties...' button below it:

- Initial Step:** Actor A: Grower, Action: selects cultivation type, Actor B: (empty)
- Step 2:** Actor A: Computer System, Action: demands task, Actor B: (empty)
- Step 3:** Actor A: Grower, Action: selects daily scouti, Actor B: (empty)
- Step 4:** Actor A: Computer System, Action: demands time period, Actor B: (empty)
- Step 5:** Actor A: Grower, Action: stores time period da, Actor B: Computer System
- Step 6:** Actor A: Computer System, Action: ea sector definition, Actor B: (empty)
- Step 7:** Actor A: Grower, Action: selects areas wher, Actor B: (empty)
- Step 8:** Actor A: Computer Sys, Action: demands pest scouti, Actor B: (empty)
- Step 9:** Actor A: Grower, Action: selects scouting m, Actor B: (empty)
- Step 10:** Actor A: Computer System, Action: sends message to t, Actor B: (empty)
- Step 11:** Actor A: Database Ser, Action: produces daily scouti, Actor B: (empty)
- Final Step:** Actor A: Computer System, Action: the action threshold to, Actor B: Grower

**Figure 5:5: Entering Main Success Scenario Data to the Use Case Data Model Window – Daily scouting and record keeping process**

After the Use Case Main Success Scenario data entry is completed by the analyst / end user, and after all the appropriate Project Properties Data (Process, Use Case, BORM General Function and BORM Action) is defined and taken into consideration, the BORM model depicted through the Object Relation Diagram (**Fig. 5:2**) is automatically derived by the UCBTA\_PROJECTS application. The *show BORM* button pressing leads to the straightforward derivation of the BORM model or the so called *Process - Participant Interaction Model*.

All the needed fields and all necessary BORM data is depicted at the following figure  
(Fig. 5:6)

**BORM (Process Participant Interaction Model)**

Initiation	Grower selects cultivation type	Participants	Grower Computer System Database Server
Action	Grower performs scouting and data record keeping of the necessary values.		
Result	Grower receives daily scouting report and system's message for action threshold and consider		
BORM General Function	Economic IPM Administration		

**Business Process Workflow**

Initial Step	Step 4	Step 7	Step 10
Grower selects cultivation type	Computer System demands time pr	Grower selects areas where scoutin	Computer System sends message to th
Grower awaits response	Grower receives time period requ	Grower expects new demand	Computer System expects server re
selection is obtained	Computer System awaits data	Computer System receives area se	Database Server daily scouting repo
Computer System receives cultivat	Demand sent	area sector data is received	daily monitoring data is obtained

Step 2	Step 5	Step 8	Step 11
Computer System demands task	Grower stores time period data to	Computer System demands pest sco	Database Server produces daily scou
Computer System is expecting n	Computer System receives time	Computer System expects scoutin	Computer System receives daily sco
demand task is transmitted	Grower expects new request	Grower receives pest monitoring m	
Grower receives task demand	Time period data is stored	demand is received	

Step 3	Step 6	Step 9	Final Step
Grower selects daily scouting	Computer System requests area se	Grower selects scouting method an	Computer System displays message fo
Grower awaits daily scouting scr	Grower obtains area sector defini	Computer System obtains expecte	Grower receives daily scouting repor
Computer System obtains daily s	area sector demand is obtained		
selection received	area sector definition is expecte		

VALIDATE BORM MODEL

**Figure 5:6: BORM (Process Participant Interaction Model) data automatically derived – Daily scouting and record keeping process**

### **5.3.1.2 Weekly scouting (monitoring) and record keeping**

#### ***5.3.1.2.1 Process description (Feasibility)***

The analyzed business process is a vital Greenhouse IPM operation, in terms of pest control effective and efficient strategy. The Agriculture expert (Grower) utilizes the IPM software to perform pest control, by analyzing pest population information, stored three times throughout a period of an entire week.

By selecting the requested time period for which pest population analysis is intended, the grower receives trend analysis report produced by the system (IPM application and application server). At the same time, the corresponding action threshold message available to the Grower, informing him of the action that should be taken against the pest population.

As far as the *process feasibility* is concerned, it has to be stated by the author that the defined process is also part of the Greenhouse IPM monitoring; thus, its concept is based on the idea that the daily performed monitoring, as it was thoroughly analyzed with the previous process requirement analysis, reveals no threat as far as the pest population is concerned. Moreover, if the procedure is repeated after the third time, and if proper statistical analysis is performed it can be possibly concluded that action threshold value is reached by the end of the weekly pest observation. In that case, proposed action by the Greenhouse IPM Information System will be proposed.

#### ***5.3.1.2.2 UCBTA Business Process Requirement Analysis***

Similarly to the previous paragraph, throughout which *daily record keeping for monitoring purposes* was analyzed in terms of the UCBTA steps of the transformation of the Use Case Model to the BORM business process requirement analysis model, and the final process representation with the ORD (Object Relation Diagram), the

complete UCBTA based requirement analysis will be delineated also for the analyzing monitoring results on a weekly basis. The UCBTA steps for deriving the business process requirement analysis are the following:

**Input Part:** The name of the delineated process which is provided throughout this initial part of the algorithm is defined as:

*“Performing monitoring (scouting) analysis based on weekly stored pest data”.*

**Use Case definition:** The asked Use Case is related to and defined according to the analyzed process. The parent Use case is entitled as:

*“Perform Scouting based on weekly Record Keeping”*

**BORM General Function:** BORM general function, which is the starting point of the changeover between the two models, is again entitled as:

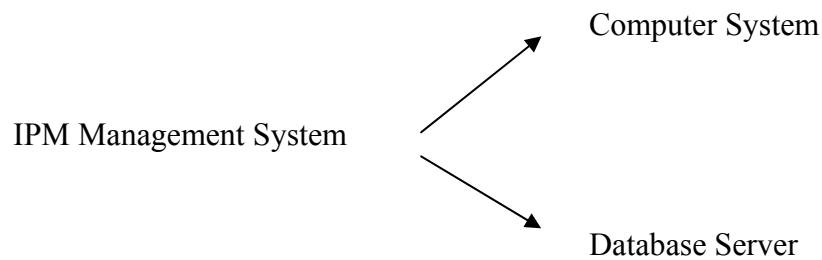
*“Economic IPM Administration”*

Following the rules of the mathematical model of the UCBTA algorithm, the Use Case according to the author of the present dissertation thesis, it has to be stated that since *scouting based on a weekly record keeping* is also part of a general *Economic IPM administration*, as well as the previous record keeping process based on daily pest observations.

**Use Case Actors’ Definition:** The concrete process, in comparison to the first business process, requires the existence of two main actors as well:

- IPM Management System
- Grower

Similarly, the mentioned IPM Management System is comprised of the following parts:



**BORM Participants' Definition:** The BORM Participants notation, with regard to the UCBTA algorithm, is exactly the same as the notation used for the Actors' definition; thus, in the case of the current described process, the participants are:

- Computer System
- Database Server
- Grower

The role and description of the Use Case Actors, and consequently the so called BORM participants which are identical according to the UCBTA rules, are defined as in the case of the daily scouting record keeping process:

Computer system: The system in which a Greenhouse IPM is installed and is utilized by the grower.

Database Server: A mainframe in which huge amount of data is stored, and in which an application server is installed for communicating with the IPM interface of the computer system.

Grower: An agronomist is a greenhouse owner or Greenhouse IPM domain expert and is able to absorb IPM business processes.

**Use Case Main Success Scenario – Initial Step:** The current process is initialized, when Grower considers weekly record keeping analysis essential for the control of pest population, and that must derive action threshold conclusions by the corresponding scouting records. Thus, the main success scenario will be the following:

*Grower selects cultivation type in order to estimate the weekly status of the pest population*

**BORM Initiation:** The notation utilized in terms of the main success scenario initial step is also used for the BORM Initiation definition

Thus, in the case of the currently delineated business process for which analytical requirement analysis is implemented, the BORM Initiation is entitled with the sentence that follows:

*Grower selects cultivation type in order to estimate the weekly status of the pest population*

**Use Case Steps Definition:** The Use case steps which refer to the present Greenhouse IPM business process are also recorded for the needs of the UCBTA algorithmic

approach. The steps that define the current business process, defined as *sub processes* of the entire process, are the following:

*A) Main Success Scenario*

- 1) Grower selects cultivation type
- 2) Computer System demands task
- 3) Grower selects weekly scouting (monitoring)
- 4) Computer system demands time period
- 5) Grower stores time period data to the system
- 6) Computer System sends the corresponding message to the Database Server
- 7) Database Server produces weekly monitoring report and action threshold message
- 8) Computer System displays message to the Grower for the pest status and action threshold

The corresponding sub steps of the above main success scenario are the following:

*B) Sub steps*

- 1a) Computer System receives cultivation type selection command
- 1b) Selection is obtained by the Computer System
- 1c) Grower awaits response
- 2a) Grower receives task demand
- 2b) Demand is transmitted to the Grower
- 2c) Computer System is expecting new selection
- 3a) Computer System obtains weekly scouting selection
- 3b) Selection is received by the Computer System
- 3c) Grower awaits daily scouting screen
- 4a) Grower receives time period request

- 4b) Demand is sent to the Grower
- 4c) Computer System awaits data (time period data)
- 5a) Computer System receives expected time period data
- 5b) Grower expects weekly pest report
- 6a) Database Server receives weekly scouting data
- 6b) Weekly monitoring data is obtained by the Server
- 6c) Computer System expects server report
- 7a) Computer System receives weekly scouting report
- 8a) Grower obtains weekly pest status report and action threshold message by the IPM system

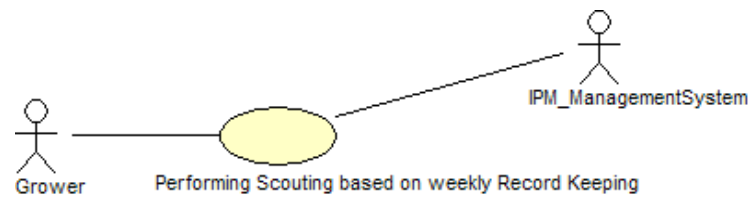
**Main success scenario, as a subset of the BORM General Function:** The main success scenario in the inner part of the BORM general Function is assumed; taking into account mathematical concepts, the main success scenario must be a subset of the BORM function; the aforementioned scenario comprises of a standard subset of the BORM General Function; consequently the model transformation algorithmic procedure can be normally derived.

**BORM Action definition:** Considering the fact that BORM action, defined with regard to the delineated process, must include all the aforementioned steps, the current action shall be entitled as it is stated below:

*Grower performs pest status analysis based on data values of the pest population measured throughout a week*

**Use Case Diagram:** The Use Case diagram which is an essential process depiction tool, before the derivation of the output object relation diagram, and which is related to the business process requirement analysis of the *Scouting (Monitoring) analysis based on weekly Record Keeping* process, will be designed according to **Fig.5:7**.





**Figure 5:7: Use Case Diagram for the “Scouting Analysis based on Weekly record keeping” Process**

**Defining the BORM Data flows:** Data flows are related to the analyzed Use Case main success scenario. Data flows express the communication between participants in BORM. Communication is achieved via connection of activities.

**Design Object Relation Diagram** : Diagram orientation depends on the fact that data flows are carefully stated with the co-operation of IT experts, stakeholders and end-users.

Furthermore, primary statement and condition which constitutes the business process requirement analysis success, is that with regard to the initial analysis level and without taking into account any software orientation, the user requirements are thoroughly defined.

In **Fig. 5:8** the currently delineated and oriented to the Greenhouse IPM business process defined as “*Performing monitoring (scouting) analysis based on weekly stored pest data*” is depicted.

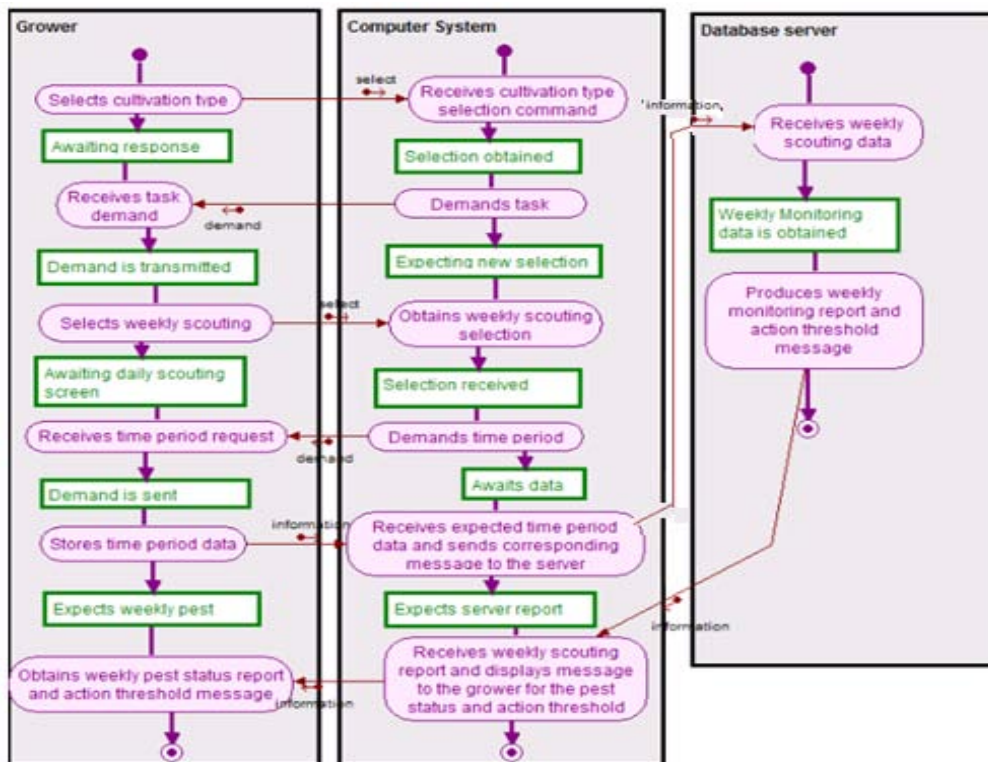


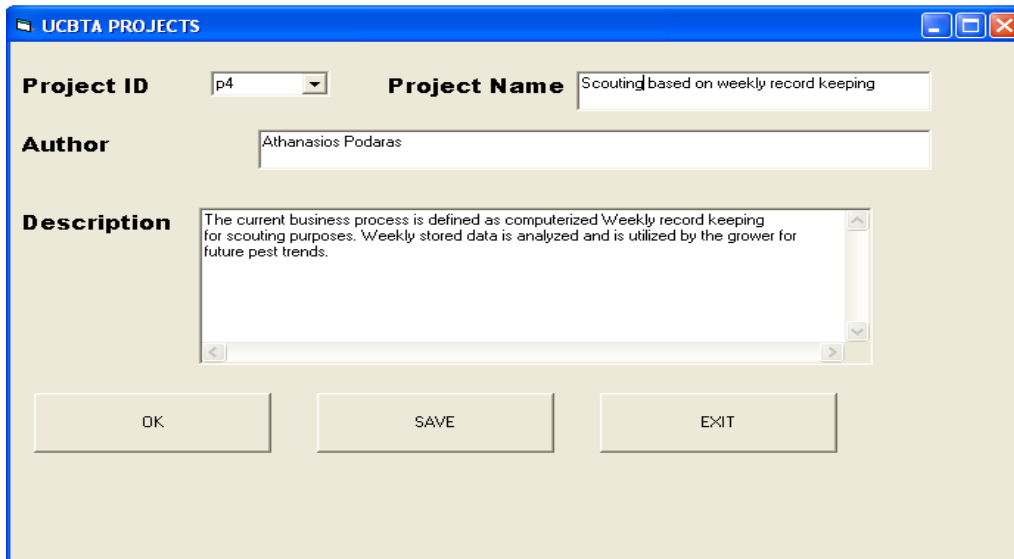
Figure 5:8: ORD Diagram for the “scouting based on weekly record keeping” process

**UCBTA Output – BORM Result :** The algorithmic output that depends on the transformation of the Use Case final step to the BORM final activity is defined by the following description:

*Grower obtains weekly pest status report and action threshold message by the IPM system*

**Deriving business process requirement analysis results with the UCBTA PROJECTS Application :**

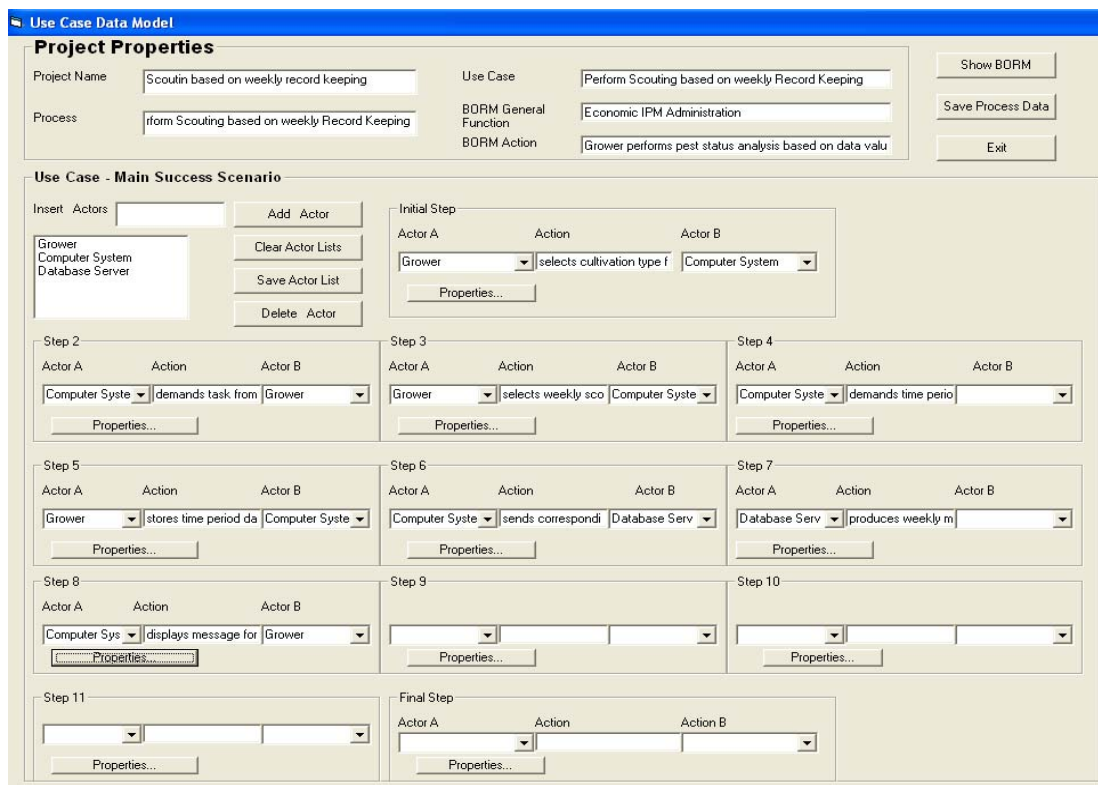
The data which is related to the *Scouting based on Weekly Record Keeping* Greenhouse IPM business process is entered to the corresponding fields of the UCBTA PROJECTS window **Fig. 5:9**.



**Figure 5:9: Entering Initial Data to the UCBTA PROJECTS Window – Weekly scouting and record keeping process**

Use Case Main Success Scenario Steps are recorded one by one to the sub – frames of the Use Case Main Success Scenario frame of the Use Case Data Model window.

**Fig.5:10.**



**Figure 5:10: Entering Main Success Scenario Data to the Use Case Data Model Window – Weekly scouting and record keeping process**

The *show BORM* button pressing leads to the straightforward derivation of the BORM model or the so called *Process - Participant Interaction Model*. All the needed fields and all necessary BORM data is depicted at the following figure **Fig. 5:11**

**BORM (Process Participant Interaction Model)**

Initiation: Grower selects cultivation type from Computer System

Action: Grower performs pest status analysis based on data values of the pest population measured th

Result: Grower obtains weekly pest status report and action threshold message by the IPM system Da

BORM General Function: Economic IPM Administration

Participants: Grower, Computer System, Database Server

**Business Process Workflow**

Initial Step: Grower selects cultivation type from Computer System

Step 4: Computer System demands time p; Grower receives time period requ; demand sent to Grower; Computer System awaits time per

Step 7: Database Server produces weekly; Computer System receives weekly

Step 10: [Empty]

Step 2: Computer System demands task fr; Grower receives task demand by; demand trasmitted to Grower; Computer System is expecting n

Step 5: Grower stores time period data to; Computer System receives expe; Grower expects weekly pest rep

Step 8: Computer System displays message; Grower obtains weekly pest status

Step 11: [Empty]

Step 3: Grower selects weekly scouting fr; Computer System obtains weekly; selection received by Computer; Grower awaits daily scouting scr

Step 6: Computer System sends correspon; Database Server receives weekl; weekly monitoring data obtained

Step 9: [Empty]

Final Step: [Empty]

VALIDATE BORM MODEL

**Figure 5:11: BORM (Process Participant Interaction Model) data automatically derived – Weekly scouting and record keeping process**

## **5.3.2 Evaluation of pesticide's effectiveness**

### **5.3.2.1 Process description (Feasibility)**

The current business process is related to the proper utilization of a pesticide; thus the BORM general function under which the process is defined is once more the Greenhouse *Economic IPM Administration*.

In the case that pest population increases and exceeds some threshold value, according to which the necessary action should be taken by the grower, then one of the Greenhouse IPM methodologies suggested is the utilization of the suitable pesticide.

Possible derived questions regarding the pesticide that should be used, is at first which is the proper quantity that must be utilized in the beginning of the action against the pest population and secondly how could the grower evaluate the effectiveness of the used pesticide after one month?

The defined business process is modelled as a proposal of a pattern based approach to evaluating pesticides' effectiveness throughout a testing period of 4 weeks. After producing the requested statistical results, the IPM information system informs the Grower of the pest effectiveness and stabilization or not of pest population, or in other words about the positive or negative implementation of the pesticide.

From all the above mentioned elements and concepts regarding the evaluation of the pesticide's effectiveness, it can be realized by the reader of the current work that it is *feasible enough* for agricultural experts and growers to co – operate with IT expert teams and model such a Greenhouse IPM business process in order to secure automated and computer based results about the positive or negative effect that a pesticide has on the pesticide revealed during the scouting procedure.

### 5.3.2.2 UCBTA Business Process Requirement Analysis

**Input Part:** The delineated process which is provided throughout this initial part of the algorithm is entitled as:

*“Evaluation of pesticide effectiveness through monthly record keeping”*

**Use Case definition:** The related Use Case, is related to and defined according to the analyzed process. The parent Use case is entitled as:

*“Evaluate pesticide effectiveness through monthly record keeping”.*

**BORM General Function:** BORM general function, which is the starting point of the changeover between the two models, is again entitled as:

*“Economic IPM Administration”*

According to the mathematical model of the UCBTA algorithm, the Use Case To BORM transition can be continued since *evaluation of pesticide effectiveness through monthly record keeping* is also part of a general *Economic IPM administration*, as well as the previous record keeping process based on daily pest observations.

**Use Case Actors’ Definition:** The concrete process, demands the existence of the same two basic Actors:

- IPM Management System
- Grower

Again it should be stated that the mentioned IPM Management System is comprised of the following parts:

- Computer System
- Database Server

**BORM Participants' Definition:** The role and description of the Use Case Actors, and consequently the so called BORM participants which are identical according to the UCBTA rules, are defined as in the case of the pesticide's effectiveness evaluation from monthly record keeping results are the following:

- Computer System
- Database Server
- Grower

**Use Case Main Success Scenario – Initial Step:** The current process is initialized, when Grower realizes the need to control the usefulness of a pesticide utilized against the revealed pest population. The Grower, having kept the necessary records regarding data about the specific pesticide, and especially the quantity of the pesticide implemented during the month, he should control if the defined quantity is less, the same or more comparing to the initial quantity implemented in the beginning of the pest emergence. Thus the BORM initiation in this case is defined as follows:

*Grower selects pesticide monthly report task with regard to the amount of pesticide consumed*

**BORM Initiation:** In the case of the currently delineated business process for which requirement analysis is performed, the BORM Initiation is entitled with the sentence that follows:

*Grower selects pesticide monthly report task with regard to the amount of pesticide consumed*

**Use Case Steps Definition:** The Use Case Main Success Scenario steps which refer to the present Greenhouse IPM process are also recorded for the needs of the UCBTA methodology.

*A) Main Success Scenario*

- 1) Grower selects pesticide monthly report task
- 2) Computer System demands time period
- 3) Grower stores time period data
- 4) Computer System demands registration number of the pesticide
- 5) Grower stores pesticide data
- 6) Computer System sends pesticide information to the Database Server
- 7) Database Server produces monthly pesticide data report and message about its effectiveness
- 8) Computer System displays message to the grower for the pesticide effectiveness

*B) Sub steps*

- 1a) Computer System receives pesticide monthly report command
- 1b) Selection is obtained by the Computer System
- 1c) Grower awaits response
- 2a) Grower receives time period demand by the system
- 2b) Computer System awaits data
- 2c) Demand (for time period data) is received by the user
- 3a) Grower expects registration number request
- 3b) Computer System receives expected time period data
- 3c) Time period data is obtained by the Computer System
- 4a) Grower receives pesticide registration number request



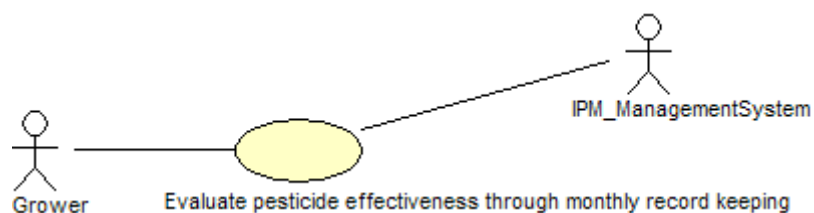
- 4b) Computer System expects registration number
- 4c) Pesticide registration number is obtained by the Grower
- 5a) Grower expects system's pesticide report
- 5b) Computer System receives registration number
- 6a) Database Server receives monthly data of the used pesticide
- 6b) Computer System expects server report
- 6c) 4 week pesticide information is obtained by the DB Server
- 7a) Computer System receives monthly pesticide report
- 8a) Grower obtains monthly pesticide effectiveness report and corresponding action message

**Main success scenario, as a subset of the BORM General Function:** It should be mentioned once more that the main success scenario must be a subset of the BORM function; the aforementioned Use Case scenario comprises of a standard subset of the BORM General Function; consequently the model transformation algorithmic procedure can be normally continued.

**BORM Action definition:** The action defined with regard to the delineated process is the following and includes all the aforementioned Use Case steps:

*Grower performs analysis based on data values and record keeping of the pesticide utilized throughout a period of 4 weeks*

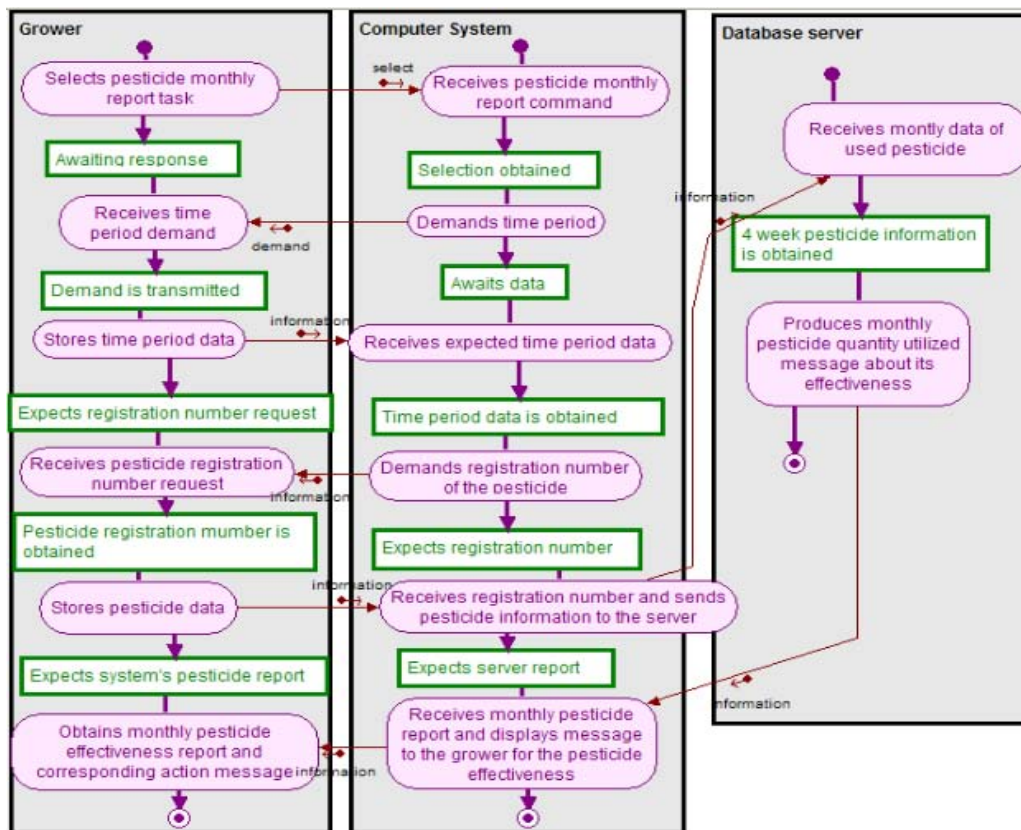
**Use Case Diagram:** The Use Case Diagram which is related to the business process requirement analysis of the *Pesticide effectiveness evaluation through monthly record keeping* process, will be designed according to **Fig. 5:12**.



**Figure 5:12: Use Case Diagram for the “Evaluating pesticide effectiveness from monthly record keeping” Process**

**Defining the BORM Data flows:** The BORM Data flows are related to the analyzed throughout the previous section concepts of the Business Process Diagram. Communication between participants, *states*, and *transitions* are defined in terms of ORD (Object Relation Diagram or Business Process Diagram) construction.

**Design Business Process Diagram :** In *Fig. 5:13* the currently delineated and oriented to the Greenhouse IPM process defined as “*Evaluation of pesticide effectiveness through monthly record keeping*” is depicted.



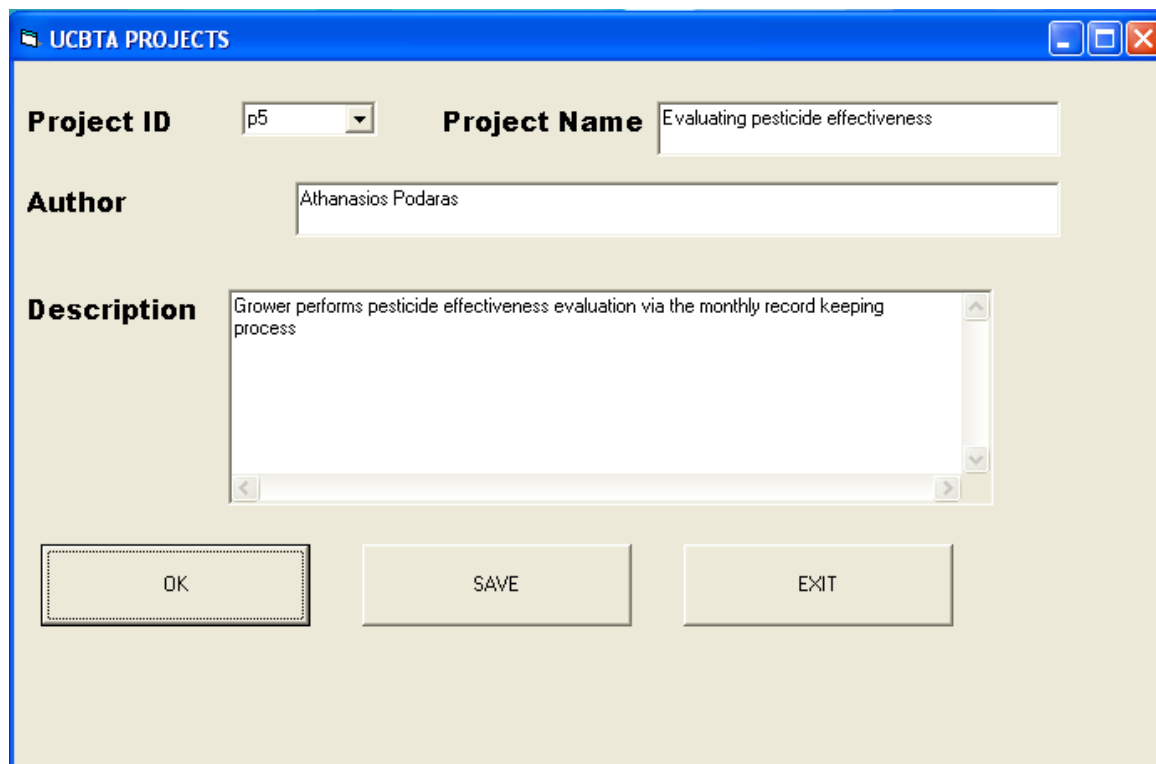
**Figure 5:13: ORD Diagram for the “Evaluating pesticide effectiveness through monthly record keeping” process**

**UCBTA Output – BORM Result** : The algorithmic output that depends on the transformation of the Use Case final step to the BORM final activity is defined by the following description:

*Grower obtains monthly pesticide effectiveness report and corresponding action message.*

**Deriving business process requirement analysis results with the UCBTA PROJECTS Application:**

The data which is related to the *Evaluation of Pesticide Effectiveness* Greenhouse IPM business process is entered to the corresponding fields of the UCBTA PROJECTS window **Fig. 5:14**



The screenshot shows a software window titled "UCBTA PROJECTS". It contains several input fields and buttons. The "Project ID" field is a dropdown menu with "p5" selected. The "Project Name" field is a text box containing "Evaluating pesticide effectiveness". The "Author" field is a text box containing "Athanasios Podaras". The "Description" field is a multi-line text area containing "Grower performs pesticide effectiveness evaluation via the monthly record keeping process". At the bottom of the window, there are three buttons: "OK", "SAVE", and "EXIT".

**Figure 5:14: Entering Initial Data to the UCBTA PROJECTS Window – Evaluation of pesticide effectiveness process**

Use Case Main Success Scenario Steps are recorded one by one to the sub – frames of the Use Case Main Success Scenario frame of the Use Case Data Model window. **Fig 5:15.**

The screenshot displays the 'Use Case Data Model' window. At the top, the 'Project Properties' section includes fields for 'Project Name' (Evaluating pesticide effectiveness), 'Process' (Evaluate pesticide effectiveness through monthly), 'Use Case' (Evaluate pesticide effectiveness through monthly record), 'BDRM General Function' (Economic IPM Administration), and 'BDRM Action' (Grower performs analysis based on data values and reco). Buttons for 'Show BDRM', 'Save Process Data', and 'Exit' are also present.

The main section is titled 'Use Case - Main Success Scenario'. It features an 'Insert Actors' list with 'Grower', 'Computer System', and 'Database Server', and buttons for 'Add Actor', 'Clear Actor Lists', 'Save Actor List', and 'Delete Actor'. Below this is the 'Initial Step' and 'Final Step' sections, each with 'Actor A', 'Action', and 'Actor B' dropdowns and a 'Properties...' button.

The scenario consists of 11 steps, each with its own 'Actor A', 'Action', and 'Actor B' dropdowns and a 'Properties...' button:

- Step 2:** Actor A: Computer System, Action: demands time period, Actor B: [blank]
- Step 3:** Actor A: Grower, Action: stores time period d, Actor B: Computer System
- Step 4:** Actor A: Computer System, Action: demands registratio, Actor B: Grower
- Step 5:** Actor A: Grower, Action: stores pesticide data, Actor B: Computer System
- Step 6:** Actor A: Computer System, Action: sends pesticide inf, Actor B: Database Serv
- Step 7:** Actor A: Database Serv, Action: bout effectiveness, Actor B: [blank]
- Step 8:** Actor A: Computer Sys, Action: displays message to, Actor B: [blank]
- Step 9:** Actor A: [blank], Action: [blank], Actor B: [blank]
- Step 10:** Actor A: [blank], Action: [blank], Actor B: [blank]
- Step 11:** Actor A: [blank], Action: [blank], Actor B: [blank]

**Figure 5:15: Entering Main Success Scenario Data to the Use Case Data Model Window – Evaluation of pesticide effectiveness process**

The *show BORM* button pressing leads to the straightforward derivation of the BORM model or the so called *Process - Participant Interaction Model*. All the needed fields and all necessary BORM data is depicted at the following figure **Fig. 5:16.**

BORM (Process Participant Interaction Model)

Initiation	Grower selects pesticide monthly report task	Participants	Grower Computer System Database Server
Action	Grower performs analysis based on data values and record keeping of the pesticide utilized thr		
Result	Grower obtains monthly pesticide effectiveness report and corresponding action message		
BORM General Function	Economic IPM Administration		

**Business Process Workflow**

Initial Step	Step 4	Step 7	Step 10
Grower selects pesticide monthly re	Computer System demands registr	Database Server produces monthly	
Computer System receives pestici	Grower ereceives pesticide regis	Computer System receives monthl	
selection is obtained by the Comp	Computer System expects registra		
Grower awaits response	pesticide registration number is o		

Step 2	Step 5	Step 8	Step 11
Computer System demands time p	Grower stores pesticide data to Co	Computer System displays message	
Grower receives time period dem	Grower expects system's pesticid	Grower obtains monthly pesticide	
Computer System awaits data Co	Computer System receives regist		
demand (for time period data) tra			

Step 3	Step 6	Step9	Final Step
Grower stores time period data to	Computer System sends pesticide i		
Grower expects registration num	Database Server receives month		
Computer System receives expe	Computer System expects server		
time period is obtained by Comp	4 week pesticide information is		

**Figure 5:16: BORM (Process Participant Interaction Model) data automatically derived – Evaluation of pesticide effectiveness process**

## 6 Conclusion

The rapid evolution of Information Technology and its emergence to all scientific fields and to all types of business is a fact that cannot be ignored by Information System developers. When a new system is integrated, a software or application is developed, or when any of the above mentioned products is upgraded for better functionality, each of the phases which will lead to the final product successful delivery to the hands of the end users must be taken into account; according to all IT experts, extensive analysis of all the aforementioned phases is regarded as indispensable when application's efficiency and effectiveness are demanded.

The most critical phase of the application or system development is the requirement analysis phase. Throughout the concrete phase the business needs of the end users are defined and analyzed by the IT experts. The current document dealt with requirement analysis at a business level; in other words, business process oriented requirement analysis is the analyzed topic of the present research work.

For the detailed business process requirement analysis, many tools have been suggested by IT experts so far. The Object – Oriented UML Use Case Analysis is the primary form of gathering requirements for a new software program or task that must be completed. It is a concept associated to both business and software requirements.

On the other hand, it has been stated by many IT experts, who strongly recommend the UML tools such as Use Case diagrams followed by the Sequence, Collaboration and State Transition Diagrams for the integration of efficient and effective requirement analysis that the aforementioned tools are too oriented at the programming concepts and quite weak in terms of business logic and business process modelling. Consequently, in the case that end users are not familiar with programming notations and are not computer oriented, the Use Case Analysis must be followed by a methodology for which such an orientation is not required.

The proposed Object – Oriented methodology to business process requirement analysis which was analyzed by throughout the present dissertation thesis is the so

called Business Object Relation Modelling (BORM). The specific method, like the Use Case Analysis approach, has been successfully utilized for the integration of several information systems worldwide, especially in Czech Republic.

The argument which is stated by the author of the current work is that what must be reassured during the process of information system integration is the smooth and precise transition from the Use Case Analysis to the BORM Methodology. The aforementioned transition is implemented by the Use Case To BORM Transformation Algorithm (UCBTA). The concrete algorithm is based on the theory of the finite state automaton, is analyzed in detail throughout the current document and is introduced by the author of the thesis as a new, modern, pattern based and not oriented in programming or strict IT concepts, which are not absorbable by the end users, business process requirement analysis method. The aforementioned algorithm which is comprised of several algorithmic steps and is pattern oriented so that IT analysts who are challenged to utilize it, will have the opportunity to avoid useless and time consuming integration steps which are related to the analysis phase.

The transition from the Use Case Model to BORM is based on four essentially proposed by the author rules, the so called UCBTA Transition rules. The concrete rules are utilized as semantics for the UCBTA procedure. Moreover, a proposed software application which is integrated by the author in order to fully support the automated transition from the Use Case Model to BORM is the UCBTA\_PROJECTS application.

What is also expected to be solved by the construction of the above mentioned algorithm is the problem of the automation of concrete agricultural business processes, and precisely Greenhouse Integrated Pest Management processes. In the beginning of the current paper, two very important applications, integrated to serve IPM purposes are analyzed, but in both cases what is underlined by the scientists and by the experts who were responsible about these applications is the gap in the communication between the IT experts and the end users.

The author's ambition with the construction of the UCBTA algorithm is the gap covering of the above stated communication with a detailed business process requirement analysis in terms of Greenhouse Integrated Pest Management.

Consequently, an IPM Case Study is presented at the end of the current thesis; the Case Study concerns the Use Case Analysis part of two important IPM business processes.

From the author's standpoint an object – oriented approach to business process requirement analysis is the most convenient for defining greenhouse computer based processes, since the agricultural scientific field involves taxonomies of plants, insects, diseases and pests that are identical to object – oriented notations such as classes, objects, subclasses and data sets. As a result, the UCBTA algorithm, as an object – oriented method to business process requirement analysis is considered to be ideal for the Greenhouse computer based IPM business process requirement analysis.

As an overall statement, the current document deals with the introduction and the detailed delineation of a new algorithm which will enable the effective and efficient business process requirement analysis entitled as Use Case To BORM Transformation Algorithm, and the implementation of UCBTA business process requirement analysis of three critical IPM business processes.

The author's future work, which will be carried out as the following scientific upgrade and achievement, will be comprised of the delineation of all processes related to Greenhouse Integrated Pest Management throughout the UCBTA theory. Moreover the Use Case to BORM business process requirement analysis of an entire greenhouse IPM will be implemented with a construction of an application based on the above mentioned UCBTA requirement analysis.

Author's another future goal is to perform an important update of the UCBTA\_PROJECTS application. The achievement will involve the automatic derivation of the BORM diagram (or Object Relation Diagram) through the currently integrated software.

Finally, new UCBTA transition rules will be proposed by the author for the derivation of more complicated ORD diagrams from more complex Use Cases.



## 7 Literature

### 7.1 *Scientific Books, Journals, Papers*

- [1] Adolph S., Bramble P., *Patterns for effective Use Cases*, 2002, ISBN 0-201-72184-8
- [2] Alexander C., *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press, 1977, ISBN 0-195-01919-9
- [3] Alexander I., Maiden N., *Scenarios, Stories, Use Cases*, Wiley, 2004, ISBN 0-470-86194-0.
- [4] Ambler S., *Building Object Applications that work*, SIGS Books, Cambridge University Press, 1997, ISBN 0-521-64826-2
- [5] Ambler S., *More Process Patterns: Delivering Large-Scale Systems Using Object Technology*, SIGS Books, Cambridge University Press, 1999, ISBN 0-521-65262-6
- [6] Ambler S., *Process Patterns: Building Large-Scale Systems Using Object Technology*, SIGS Books, Cambridge University Press, 1998, ISBN 0-521-64568-9
- [7] Armour F., Miller G., *Advanced Use Case Modeling: Software Systems*. Addison-Wesley Professional, 2000, ISBN 0 -201-61592-4
- [8] ATTRA (Appropriate Technology Transfers for Rural Areas), *Integrated Pest Management for Greenhouse Crops*, Pest Management Systems Guide, ISBN 800-346-9140).
- [9] Bahrami, *Object – Oriented System Development*, McGraw Hill, 1999, ISBN 0 – 071 – 16090 – 6
- [10] Beck K, *Smalltalk best practice Patterns*, Prentice Hall, 1997, ISBN 0-13-476904-X
- [11] Bennett et Al., *Truman's Scientific Guide to Pest Management Operations*, 6th edition, Purdue University/Questex Press, 2005, p. 10
- [12] Bittner K., Spence I., *Use Case Modeling*, Addison-Wesley Professional, 2002, ISBN 0-201-70913-9

- [13] Blaha M., Premerlani W., *Object – Oriented Modeling and Design for Database Applications*, Prentice Hall, 1998, ISBN 0 – 13 – 123829 – 9
- [14] Booch G., Rumbaugh J., Jacobson I., *The Unified Modeling Language User Guide*, Addison – Wesley, 1998, ISBN – 0 – 201 – 57168 – 4
- [15] Booth T. L., *Sequential Machines and Automata Theory*, John Wiley and Sons, 1967, New York.
- [16] Canton M., *Object Oriented Project Management with UML*, J. Willey and Sons, 1998, ISBN 0 – 471 – 25303 – 0
- [17] Carda G., Merunka V., Polák J., *Umění systémového návrhu*, Grada , 2002, ISBN 80-847-0424-2
- [18] Catell R.G.G., *The Object Database Standard – ODMG93*, Morgan Kauffman Publishers, 1994, ISBN 1 – 55860 – 302 – 6
- [19] Chant D. A., *Strategy and tactics of insect control*, Can. Entomol. 16, 1964
- [20] Cockburn A., *Writing Effective Use Cases*, Boston, MA, USA: Addison-Wesley Longman Publishing Co., 2001, ISBN 0-201-70225-8
- [21] Collyer E., *Biology of some predatory insects and mites associated with the fruit tree red spider mite (Metatetranychus ulmi (Koch) in south-eastern England II*, Some important predators of the mite. J. Hortic. Sci. 28, 1953, p. 85–112.
- [22] Cooper B. S., *Computability Theory*, Chapman and Hall/CRC, 2004, ISBN 1-58488-237-9.
- [23] Coplien J. O., *A generative development – Process pattern language. Pattern Languages of Program Design*, eds. Coplien J.O. and Schmidt D.C., Addison – Wesley Longman, Inc., 183-237
- [24] Coterrel M., Hughes B., *Software Project Management*, Thomson Computer Press, 1995, ISBN 1 – 850 – 32190 – 6
- [25] Cox B. J., *Object Oriented Programming – An evolutionary approach*, Addison Wesley, 1986, ISBN 0 – 201 – 10393 – 1
- [26] Cuperus G. W., Mulder P. G., and Royer T. A., *Insect Pest Management Techniques for Environmental Protection*, Chapter 6, CRC Press LLC, 2000, ISBN 1-56670-478-2,
- [27] Davis A., *Software Requirements – Objects, Functions, and States*, Prentice Hall, 1993, ISBN 0 – 13 – 562174 – 7
- [28] Denney R., *Succeeding with Use Cases: Working Smart to Deliver Quality*, Addison-Wesley Professional, 2005, ISBN 0-321-31643-6.

- [29] Derr K.W., *Applying OMT – A practical guide to using the Object Modelling Technique*, Sigs Books, 1995, ISBN 0 – 13 – 884842 – 10 – 0 , Prentice Hall, 1995, ISBN 0 -13 – 231390 – 1
- [30] Dreistadt S., *Integrated pest management for floriculture and nurseries*, University of California, University of California Integrated Pest Management Program, Division of Agriculture and Natural Resources, ANR Publications, 2001, ISBN 1879906465, 9781879906464
- [31] Epstein R., Carnielli W., *Computability: Computable Functions, Logic, and the Foundations of Mathematics, with Computability: A Timeline (2nd ed.)*, Wadsworth/Thomson Learning, 2000, ISBN 0-534-54644-7
- [32] Eriksson H. E., Penker M., *Business Modelling with UML*, J. Wiley and Sons, 2000, ISBN – 0 – 471 – 29551 – 5
- [33] Flint M.L., Daar S., Moinar R., *Establishing Integrated Pest Management Policies and Programs: A guide for public agencies*, University of California, Division of Agriculture and Natural Resources, 2003, ISBN 978-1-60107-267-2
- [34] Fowler M., Scott K., *UML Distilled (2<sup>nd</sup> Edition)*, Addison Wesley, 1999, ISBN 0 – 201 – 65783 – X
- [35] Franz J.M., *Biological control and pest insects in Europe*, Annu. Rev. of Entomol. 6, 1961
- [36] Gray Peter M.D, Kulkarni Krishnarao G., Paton Norman W., *Object Oriented Databases – A Semantic Data Model Approach*, Prentice Hall, 1992, ISBN 0 – 13 – 630203 – 3
- [37] Gurari E.,). *An Introduction to the Theory of Computation*, Computer Science Press, 1989, ISBN 0-7167-8182-4
- [38] Harrison N. B., *Organizational Patterns for teams. Pattern Languages of Program Design 2*, eds. Vlissides J.M., Coplien J.O. and Kerth N..L., Addison – Wesley Publishing Company, 345-352
- [39] Henderson – Sellers B., *A book of Object – Oriented Knowledge*, Prentice Hall, 1991, ISBN 0 – 13 – 059445 – 8
- [40] Hopcroft J. E., Ullman J. D., *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley Publishing Company, Reading Mass, ISBN 0-201-02988-X
- [41] Hopkins T., Horan B., *Smalltalk – an introduction to application development using VisualWorks*, Prentice Hall, 1995, ISBN 0-13-318387-4

- [42] Hunt, J.: *Smalltalk and Object Orientation*, Springer, 1997, ISBN 3540761152
- [43] Jacobson I., *Object – Oriented Software Engineering - A Use Case Driven Approach*, Addison Wesley, 1992, ISBN 0 – 201 – 54435 – 0
- [44] Knott R., Merunka V., Polak J., *Process Modeling for Object – Oriented Analysis using BORM Object Behavioral Analysis*, Proceedings of 4<sup>th</sup> International Conference on Requirements Engineering ICRE 2000, Chicago, Computer Society Press, 2000, ISBN 0 – 7695 – 0565 – 1
- [45] Knott R., Merunka V., Polak J., *The role of Object – Oriented Process Modeling in Requirements Engineering phase of Information Systems Development*, Conference Proceedings EFITA 2003, Debrecen, Hungary, 2003, p. 300 – 307
- [46] Kontonya G., Sommerville I., *Requirements Engineering: Processes and Techniques*, J. Wiley and Sons, 1999
- [47] Kozen D., *The design and analysis of algorithms*, Springer Verlag, New York, Berlin, 1992, ISBN 3-45-997082-6
- [48] Kulak D., Guiney E., *Use Cases – Requirements in context*, Second Edition, 2003, ISBN 0-321-15498-3
- [49] Larman G., *Applying UML and Patterns – An introduction to Object – Oriented Analysis and Design*, Prentice Hall, 1997, ISBN 0-13748-880-7
- [50] Liddle S., *Stakeholders for use case specifications*, 2009
- [51] Lorentzos N. A.; Yialouris, C.P. and Sideridis A. B., *Time-evolving rule-based knowledge bases*, Data & Knowledge Engineering 29, 1999, p.313-335.
- [52] Maliappis, M.T., Sideridis, A.B., Mahaman, B.D., *NEST: A New Expert System Tool and its application to pest management*, Proc. of the 3rd European Conference of the European Federation for Information Technology in Agriculture, Food and Environment (*J. Steffe, Ed.*), Vol. 2, 2001, pp. 421-426, organized by the Agro Montpellier Ecole Nationale Superieure Agronomique de Montpellier and ENITAB, Montpellier, France.
- [53] Martin J., Odell J., *Object-Oriented Methods*, Prentice Hall, 1997, ISBN 0-13905-597-5
- [54] McClusky E., *Introduction to the Theory of Switching Circuits*, McGraw-Hill, 1965
- [55] Merunka V., *BORM – Overview of the methodology and case study of the agrarian information system*, AGRIC. ECON. Magazine, Czech Republic, 2003, p. 397 – 406

- [56] Meyer B., *Object-Oriented Software Construction*, London, Prentice Hall, 1988
- [57] Ministry of Agriculture of Bulgaria, *List of the permitted products for plant protection and fertilizers in Bulgaria*, Bulgaria, Sofia, Videnov and Son, 2002.
- [58] Onkov K., Dimova D., “*Information technology for creation and use of PC “Phytopharmacy” database*”, Proc of HAICTA 2006 International Conference on “Information Systems in Sustainable Agriculture, Agroenvironment and Food
- [59] Pilone D, Pitman N., *UML 2.0 in a Nutshell – A desktop quick reference*, O’Reilly Media, Inc., 2005, ISBN – 0 – 596 – 00795 – 7
- [60] Rabin M. O., Scott D., *Finite Automata and their Decision Problems*, IBM Journal of Research and Development, 3:2 , 1959, p.115-125.
- [61] Royce, Walker, *Software Project Management – A Unified Framework*, Addison Wesley, 1998, ISBN 0 – 201 – 30958 – 0
- [62] Rumbaugh J., Jacobson I., Booch G., *The Unified Modeling Language Reference Manual*, Addison – Wesley, 1999, ISBN – 0 – 201 – 30998 – X
- [63] Satzinger John W., Orvik Tore U., *The Object – Oriented Approach – Concepts, Modeling and System Development*, Boyd & Fraser, 1996, ISBN 0 – 7895 – 0110 – 4
- [64] Shiver B., Wegner P., *Research Directions in OOP*, MIT Press, 1987, ISBN 0 – 262 – 19264 – 0
- [65] Simone A. J. H. and Graham I., *30 Things that go wrong in Object Modeling with UML 1.3*, chapter 17 in: Behavioral Specifications of Business and Systems eds. H Kilov, B. Rumpe, I. Simmonds, Kluwer Academic Publishers, 1999, p. 237 -257
- [66] Sipser, Michael., *Introduction to the Theory of Computation* (2nd ed.), Boston Mass: Thomson Course Technology, 2006, ISBN-10: 0-534-95097-3.
- [67] Stern V. M., Smith R. F., Van den Bosch R., Hagen K. S., *The integration of chemical and biological control of the spotted alfalfa aphid. I. The integrated control concept*, Hilgardia, 1959
- [68] Taylor G., *Models of Computation and Formal Languages*, New York, Oxford University Press, 1998, ISBN 978-0195109832 An unusually readable textbook, appropriate for upper-level undergraduates or beginning graduate students.
- [69] Topliff L., Schnelle M., Pinkston K., Cuperus G., Broembsen S., *Integrated Pest management – Scouting and Monitoring for Pests in Commercial Greenhouses*, Oklahoma State University, Division of Agriculture and Natural Resources
- [70] Van den Bosch R. and Stern V. M., *The integration of chemical and biological control of arthropod pests*, *Annu. Rev. of Entomol.* 7, 1962

- [71] Vaniček J., Pergl R., Papík M. and Vaniček, T., *Mathematical foundations of computer science*, Kernberg Publishing & Alfa Publishing, Praha, 2008, 272p., ISBN 978-80-87168-06-6.
- [72] Vrana et al., *Methods for Building a University Information System*, Brno, 2001, Edited by EUNIS, ISBN 80 – 214 – 1837 – 0
- [73] Weir C., *Patterns for designing in Teams, Pattern Languages of Program Design 3*, eds. Martin R.C. and Reihle D. and Buschmann F., Addison – Wesley Longman, Inc., 487-501
- [74] Weis M., *Data structures and algorithm analysis in C (2<sup>nd</sup> Edition)*, Addison Wesley, 1992, ISBN 0-80535-440-9
- [75] Winemiller E., Roff J. T., Heyman B., Groom R., *Visual Basic 6 Database – How To*, The Waite Group's, 1998, ISBN 0-57169-152-9
- [76] Yialouris, C. P., *Expert Systems: On the Structure of Expert System Shells-Applications in Agriculture*, Ph.D. Dissertation (In Greek with English abstract), Agricultural University of Athens, Science Department, Athens, Greece, 1993.
- [77] Yialouris, C. P. and Sideridis A. B., *A Kernel System for Developing Expert System Shells*, In: Proceedings Computer Models in Mediterranean Crop Production (M. R Dobao, and J. G. Andujar (Ed.)), 73-80. Investigacion Agraria, Madrid, Spain. 1994.
- [78] Yourdon E., *Mainstream Objects – An Analysis and Design Approach for Business*. Prentice Hall, 1995, ISBN 0 – 13 – 209156 – 9

## 7.2 Author's Literature

### Author's work: Publications in Scientific books and journals

[79] Podaras A.: *Efficient business process requirement analysis via transformation of the Use Case Model to the Business Object Relation Modeling (BORM) Method* (accepted for publication) In: Knowledge Management and Modern Information Technologies (Handbook of Research on Knowledge Management and Implementation), Alfa Nakladatelství, 2010, Praha, (being prepared for publishing)

### Author's work: Author's work: Publications in conference proceedings

[80] Podaras, A.: *Developing Information Systems with modern and advanced methodologies – Implementing the proposed techniques by constructing I.S. of the National Garden of Athens*, ČZU, Praha, 2006, ISBN 80-213-1474-5, Proceedings of THINK TOGETHER 2006

[81] Podaras, A.: *Object Oriented Reusable Analysis Patterns for implementing Integrated Pest Management Processes on Vegetable Bedding Plants – Using BORM Object approach to illustrate a dynamic model of the aforementioned processes*, ČZU, Praha, 2006, ISBN 80-213-1568-7, Proceedings of OBJEKTY 2006

[82] Merunka V., Podaras, A.: *Výuka objektového modelování pomocí Smalltalku*, ČZU, Ostrava 2006, ISBN 80-213-1568-7, Proceedings of 2006

[83] 4a) Podaras, A.: *Object Oriented Reusable Analysis Patterns for implementing Integrated Pest Management Processes on Vegetable Bedding Plants*, ČZU, Praha, 2007, ISBN 978-80-213-1623-2, sborník THINK TOGETHER 2007

[84] Podaras, A.: *Constructing Dynamic Object Greenhouse Process Models using BORM methodology*, ČZU, Praha, 2008, ISBN 978-80-213-1755-0, sborník THINK TOGETHER 2008

[85] Podaras, A.: *Greenhouse Pest Control practices based on Object Oriented Analysis and BORM business process models*, Proceedings of HAICTA 2008, Alexandroupolis, Greece, 2008, ISBN 80-213-1474-5

### 7.3 Internet

- [86] Ambler S., *UML 2 Use Case Diagrams*, Agile Modeling, <http://www.agilemodeling.com/artifacts/useCaseDiagram.htm>
- [87] Blueprint Technologies White Paper, “*Requirements Architecture*”: *A Use Case Driven Approach*”, <http://www.blueprinttechnologies.com/training/whitepapers/requir.html>
- [88] Bredemeyer D. Malan R, “*Functional Requirements and Use Cases*”, <http://www.digilife.be/quickreferences/pt/functional%20requirements%20and%20use%20cases.pdf>
- [89] Chitnis M., Tiwari P., Ananthamuthy L., *Creating Use Case Diagrams*, 2003, <http://www.developer.com/article.php/2109801>
- [90] Christerson, Magnus, *From Use Cases to Components*, Rose Architect, 5/99. <http://www.rosearchitect.com/cgi-bin/viewprint.pl>
- [91] Cockburn, Alistair, *Structuring Use Cases with Goals*, Journal of Object-Oriented Programming, Sep-Oct, 1997 and Nov-Dec, 1997. Also available on <http://alistair.cockburn.us/Structuring+use+cases+with+goals>
- [92] Cockburn, Alistair, “*Basic Use Case Template*”, Oct .1998. Available on <http://members.aol.com/acockburn/papers/uctempla.htm>
- [93] Coleman, Derek, *A Use Case Template: Draft for discussion*, Fusion Newsletter, April 1998. [http://www.hpl.hp.com/fusion/md\\_newsletters.html](http://www.hpl.hp.com/fusion/md_newsletters.html)
- [94] Constantine, Larry, *What Do Users Want? Engineering usability into software*”, <http://www.foruse.com>
- [95] Craft.Case, “*Business Process Modeling*”, <http://www.craftcase.com/> Engineering Notebook, C++ Report, “*UML Use Case Diagrams*”, 1998, <http://www.objectmentor.com/resources/articles/usecases.pdf>
- [96] Government of Alberta, *Greenhouse Bedding Plant Production and Marketing*, [http://www1.agric.gov.ab.ca/\\$department/deptdocs.nsf/all/agdex2864#key](http://www1.agric.gov.ab.ca/$department/deptdocs.nsf/all/agdex2864#key)
- [97] Merunka et al., *BORM – Business Object Relation Modeling*, 2000, [http://www.cse.msu.edu/ICRE2000/Merunka/borm\\_html/index.html](http://www.cse.msu.edu/ICRE2000/Merunka/borm_html/index.html)
- [98] Merunka et al., *BORM – Business Object Relation Modeling*, 2009, <http://aisel.aisnet.org/amcis2009/788/>



- [99] Parlez UML, “*Requirements Analysis using UML 2 Days*”, 2005,  
<http://parlezuml.com/training/umlrequirements.htm>
- [100] Pols, Andy, *Use Case Rules of Thumb: Guidelines and lessons learned*, Fusion Newsletter, Feb. 1997. available at  
[http://www.hpl.hp.com/fusion/md\\_newsletters.html](http://www.hpl.hp.com/fusion/md_newsletters.html)
- [101] “*UML Specification*”, <http://www.rational.com/uml/index.jttml> , paper.
- [102] University of Massachusetts Amherst, *Pest Management for Vegetable Bedding Plants*,  
[http://www.umass.edu/umext/floriculture/fact\\_sheets/pest\\_management/vegpest.html](http://www.umass.edu/umext/floriculture/fact_sheets/pest_management/vegpest.html)
- [103] University of Missouri, St Louis, *Object Oriented Analysis Methodology*,  
[http://www.umsl.edu/~sauterv/analysis/488\\_f01\\_papers/wang.htm](http://www.umsl.edu/~sauterv/analysis/488_f01_papers/wang.htm)
- [104] “*Use Case Fundamentals*”, 1998,  
<http://members.aol.com/acockburn/papers/AltIntro.htm>
- [105] “*Use Case Modelling: Capturing user requirements*”,  
[http://www.zoo.co.uk/~z0001039/PracGuides/pg\\_use\\_cases.htm](http://www.zoo.co.uk/~z0001039/PracGuides/pg_use_cases.htm)
- [106] Water Management Guidelines for Nursery/Floral Producers, *Integrated Pest Management for Greenhouse Crops*,  
<http://aggie-horticulture.tamu.edu/greenhouse/nursery/environ/wmipm.html>

## 8 Appendix

### VISUAL BASIC CODE

#### Login FORM :Starts UCBTA\_PROJECTS Application and connects it to MS Access Database

```
'check for correct password

If txtPassword = "than76" Then
'place code to here to pass the
'success to the calling sub
'setting a global var is the easiest
LoginSucceeded = True
frm1.Show
Else
MsgBox "Invalid Password, try again!", , "Login"
txtPassword.SetFocus
SendKeys "{Home}+{End}"
End If

'Dim db As Database
'Dim rs As Recordset

Set db = DBEngine.OpenDatabase("C:\Documents and Settings\Thanos\My
Documents\UCBTA_DBS.mdb", False, False)

Set rs = db.OpenRecordset("Projects")

Do Until rs.EOF = True
frm1.Combol.AddItem rs("Project_ID")
rs.MoveNext
Loop

rs.Close

End Sub

Private Sub cmdCancel_Click()
'set the global var to false
'to denote a failed login
LoginSucceeded = False
Me.Hide
End Sub
```

## **FORM 1: UCBTA PROJECTS Form**

### **OK BUTTON**

```
Private Sub frm1cmd1_Click()

Dim db As Database
Dim rs As Recordset

Set db = DBEngine.OpenDatabase("C:\Documents and Settings\Thanos\My
Documents\UCBTA_DBS.mdb", False, False)

If Combol.Text <> "" And frm1txt1.Text <> "" Then

frm2.Show
frm2.frm2frame1txt1.Text = frm1.frm1txt1.Text
Else
MsgBox "Project ID and Project Name fields cannot be blank! Please
fill in the blank fields and continue!", vbOKCancel, "Invalid project
data"
Combol.SetFocus

End If

End Sub
```

### **SAVE BUTTON**

```
Private Sub frm1cmd2_Click()
Dim db As Database
Dim rs As Recordset

Set db = DBEngine.OpenDatabase("C:\Documents and Settings\Thanos\My
Documents\UCBTA_DBS.mdb", False, False)

'code to add data to a fields of table Projects with the button

Set rs = db.OpenRecordset("Projects")
rs.AddNew
rs.Fields("Project_ID") = frm1.Combol.Text
rs.Fields("Project Name") = frm1.frm1txt1.Text
rs.Fields("Author") = frm1.auth.Text
rs.Fields("Description") = frm1.desc.Text
rs.Update

Do Until rs.EOF = True
frm1.Combol.AddItem rs("Project_ID")
rs.MoveNext
Loop

rs.Close

End Sub
```

### **EXIT BUTTON**

```
Private Sub frm1cmd3_Click()  
End  
End Sub
```

### **PROJECT\_ID COMBO1**

```
Private Sub Combo1_Click()  
  
' code to show elements of every record from table PROJECTS  
  
Dim db As Database  
Dim rs As Recordset  
  
Set db = DBEngine.OpenDatabase("C:\Documents and Settings\Thanos\My  
Documents\UCBTA_DBS.mdb", False, False)  
  
Set rs = db.OpenRecordset("select * from [Projects]where  
[Project_ID]='" & frm1.Combo1.Text & "'")  
  
If rs Is Nothing Then  
  
frm1.frm1txt1.SetFocus  
Else  
frm1txt1 = rs("Project Name")  
auth = rs("Author")  
desc = rs("Description")  
  
End If  
  
End Sub
```

### **FORM 2: USE CASE DATA MODEL FORM**

#### **PROCESS\_TEXTBOX**

```
Private Sub frm2frame1txt3_Change()  
  
Dim X As String  
  
X = frm2frame1txt3.Text  
  
frm2frame1txt2.Text = X  
  
End Sub
```

### ADD ACTOR BUTTON

```
Private Sub uccmd1_Click()  
  
frm2list1.AddItem frm2frame2txt0.Text      'Add the entered the  
characters to the list box  
  
frm2frame2txt0.Text = ""                  'Clearing the text box  
  
frm2frame2txt0.SetFocus                    'Get the focus back to the  
text box  
  
'lblDisplay.Caption = lstName.ListCount  'Display the number of  
items in the list box  
  
End Sub
```

### CLEAR ACTOR LIST BUTTON

```
Private Sub delactorscmd_Click()  
  
ucframecombo1.Clear  
ucframeCombo2.Clear  
ucframeCombo3.Clear  
ucframeCombo4.Clear  
ucframeCombo5.Clear  
ucframeCombo6.Clear  
ucframeCombo7.Clear  
ucframeCombo8.Clear  
ucframeCombo9.Clear  
ucframeCombo10.Clear  
ucframeCombo11.Clear  
ucframeCombo12.Clear  
ucframeCombo13.Clear  
ucframeCombo14.Clear  
ucframeCombo15.Clear  
ucframeCombo16.Clear  
ucframeCombo17.Clear  
ucframeCombo18.Clear  
  
frm3.frm3frame1combo1.Clear  
frm3.frm3frame1combo2.Clear  
frm3.frm3frame1combo3.Clear  
frm3.frm3frame1combo4.Clear  
frm3.frm3frame1combo5.Clear  
frm3.frm3frame1combo6.Clear  
frm3.frm3frame1combo7.Clear  
frm3.frm3frame1combo8.Clear  
frm3.frm3frame1combo9.Clear  
frm3.frm3frame1combo10.Clear  
frm3.frm3frame1combo11.Clear  
frm3.frm3frame1combo12.Clear  
  
End Sub
```

### SAVE ACTOR LIST BUTTON

```
Private Sub uccmd3_Click()
```

```
'This is a way to copy the list of a listbox to all comboboxes
```

```
Dim i As Integer
```

```
For i = 0 To frm2list1.ListCount - 1
```

```
ucframeCombo1.AddItem frm2list1.List(i)  
ucframeCombo2.AddItem frm2list1.List(i)  
ucframeCombo3.AddItem frm2list1.List(i)  
ucframeCombo4.AddItem frm2list1.List(i)  
ucframeCombo5.AddItem frm2list1.List(i)  
ucframeCombo6.AddItem frm2list1.List(i)  
ucframeCombo7.AddItem frm2list1.List(i)  
ucframeCombo8.AddItem frm2list1.List(i)  
ucframeCombo9.AddItem frm2list1.List(i)  
ucframeCombo10.AddItem frm2list1.List(i)  
ucframeCombo11.AddItem frm2list1.List(i)  
ucframeCombo12.AddItem frm2list1.List(i)  
ucframeCombo13.AddItem frm2list1.List(i)  
ucframeCombo14.AddItem frm2list1.List(i)  
ucframeCombo15.AddItem frm2list1.List(i)  
ucframeCombo16.AddItem frm2list1.List(i)  
ucframeCombo17.AddItem frm2list1.List(i)  
ucframeCombo18.AddItem frm2list1.List(i)  
ucframeCombo19.AddItem frm2list1.List(i)  
ucframeCombo20.AddItem frm2list1.List(i)  
ucframeCombo21.AddItem frm2list1.List(i)  
ucframeCombo22.AddItem frm2list1.List(i)  
ucframeCombo23.AddItem frm2list1.List(i)  
ucframeCombo24.AddItem frm2list1.List(i)
```

```
frm3.frm3frame1combo1.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo2.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo3.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo4.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo5.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo6.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo7.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo8.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo9.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo10.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo11.AddItem frm2.frm2list1.List(i)  
frm3.frm3frame1combo12.AddItem frm2.frm2list1.List(i)
```

```
frm4.frm4list1.AddItem frm2list1.List(i)
```

```
Next
```

```
End Sub
```

### DELETE ACTOR BUTTON

```
Private Sub uccmd2_Click()  
  
frm2list1.RemoveItem frm2list1.ListIndex ' REMOVE AN ITEM FROM THE  
LIST  
  
frm2frame2txt0.Text = "" 'Clearing the text box  
  
frm2frame2txt0.SetFocus 'Get the focus back to the  
text box  
  
End Sub
```

### CODE FOR ALL PROPERTIES BUTTONS

```
Private Sub ucframe10cmd13_Click()  
frm3.Show  
frm3.frm3frame1txt7.Text = ucframe10.Caption  
frm4.bpwstep9.Text = frm2.ucframeCombo19.Text + " " +  
frm2.frm2frame2txt10.Text + " " + frm2.ucframeCombo20.Text  
  
End Sub
```

```
Private Sub ucframe11cmd14_Click()  
frm3.Show  
frm3.frm3frame1txt7.Text = ucframe11.Caption  
frm4.bpwstep10.Text = frm2.ucframeCombo21.Text + " " +  
frm2.frm2frame2txt11.Text + " " + frm2.ucframeCombo22.Text  
  
End Sub
```

```
Private Sub ucframe12cmd15_Click()  
frm3.Show  
frm3.frm3frame1txt7.Text = ucframe12.Caption  
frm4.bpwstep20.Text = frm2.ucframeCombo23.Text + " " +  
frm2.frm2frame2txt12.Text + " " + frm2.ucframeCombo24.Text  
  
End Sub
```

```
Private Sub ucframe1cmd4_Click()  
frm3.Show  
frm3.frm3frame1txt7.Text = ucframe1.Caption  
frm4.bpwstep1.Text = frm2.ucframeCombo1.Text + " " +  
frm2.frm2frame2txt1.Text + " " + frm2.ucframeCombo2.Text  
  
End Sub
```

```
Private Sub ucframe2cmd5_Click()  
frm3.Show  
frm3.frm3frame1txt7.Text = ucframe2.Caption  
frm4.bpwstep2.Text = frm2.ucframeCombo3.Text + " " +  
frm2.frm2frame2txt2.Text + " " + frm2.ucframeCombo4.Text
```

```

End Sub

Private Sub ucframe3cmd6_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe3.Caption
frm4.bpwstep3.Text = frm2.ucframeCombo5.Text + " " +
frm2.frm2frame2txt3.Text + " " + frm2.ucframeCombo6.Text

End Sub

Private Sub ucframe4cmd7_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe4.Caption
frm4.bpwstep4.Text = frm2.ucframeCombo7.Text + " " +
frm2.frm2frame2txt4.Text + " " + frm2.ucframeCombo8.Text

End Sub

Private Sub ucframe5cmd8_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe5.Caption
frm4.bpwstep5.Text = frm2.ucframeCombo9.Text + " " +
frm2.frm2frame2txt5.Text + " " + frm2.ucframeCombo10.Text

End Sub

Private Sub ucframe6cmd9_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe6.Caption
frm4.bpwstep6.Text = frm2.ucframeCombo11.Text + " " +
frm2.frm2frame2txt6.Text + " " + frm2.ucframeCombo12.Text

End Sub

Private Sub ucframe7cmd10_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe7.Caption
frm4.bpwstep7.Text = frm2.ucframeCombo13.Text + " " +
frm2.frm2frame2txt7.Text + " " + frm2.ucframeCombo14.Text

End Sub

Private Sub ucframe8cmd11_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe8.Caption
frm4.bpwstep8.Text = frm2.ucframeCombo15.Text + " " +
frm2.frm2frame2txt8.Text + " " + frm2.ucframeCombo16.Text

End Sub

Private Sub ucframe9cmd12_Click()
frm3.Show
frm3.frm3frame1txt7.Text = ucframe9.Caption
frm4.bpwstep30.Text = frm2.ucframeCombo17.Text + " " +
frm2.frm2frame2txt9.Text + " " + frm2.ucframeCombo18.Text

End Sub

```



### **SHOW BORM BUTTON**

```
Private Sub Command1_Click()

frm4.Show
frm4.Text4.Text = frm2.frm2frame1txt4.Text
frm4.Text2.Text = frm2.frm2frame1txt5.Text

frm4.Text1.Text = frm2.ucframecomb1.Text + " " +
frm2.frm2frame2txt1.Text + " " + frm2.ucframeCombo2.Text
'frm4.Text3.Text = frm2.ucframeCombo17.Text + " " +
frm2.frm2frame2txt9.Text + " " + frm2.ucframeCombo18.Text

End Sub
```

### **EXIT BUTTON CODE**

```
Private Sub Command3_Click()

Me.Hide

End Sub
```

## **FORM 3: USE CASE STEP DETAILS FORM**

### **CHECKBOXES' CODE**

```
Private Sub frm3Check1_Click()
If frm3Check1.Value = 1 Then
frm3Check2.Enabled = False
Else: frm3Check2.Enabled = True
End If

End Sub

Private Sub frm3Check10_Click()
If frm3Check10.Value = 1 Then
frm3Check9.Enabled = False
Else: frm3Check9.Enabled = True
End If
End Sub

Private Sub frm3Check11_Click()

If frm3Check11.Value = 1 Then
frm3Check12.Enabled = False
Else: frm3Check12.Enabled = True
```

```

End If

End Sub

Private Sub frm3Check12_Click()

If frm3Check12.Value = 1 Then
frm3Check11.Enabled = False
Else: frm3Check11.Enabled = True
End If

End Sub

Private Sub frm3Check2_Click()
If frm3Check2.Value = 1 Then
frm3Check1.Enabled = False
Else: frm3Check1.Enabled = True
End If

End Sub

Private Sub frm3Check3_Click()
If frm3Check3.Value = 1 Then
frm3Check4.Enabled = False
Else: frm3Check4.Enabled = True
End If
End Sub

Private Sub frm3Check4_Click()
If frm3Check4.Value = 1 Then
frm3Check3.Enabled = False
Else: frm3Check3.Enabled = True
End If
End Sub

Private Sub frm3Check5_Click()
If frm3Check5.Value = 1 Then
frm3Check6.Enabled = False
Else: frm3Check6.Enabled = True
End If
End Sub

Private Sub frm3Check6_Click()
If frm3Check6.Value = 1 Then
frm3Check5.Enabled = False
Else: frm3Check5.Enabled = True
End If
End Sub

Private Sub frm3Check7_Click()
If frm3Check7.Value = 1 Then
frm3Check8.Enabled = False
Else: frm3Check8.Enabled = True
End If
End Sub

Private Sub frm3Check8_Click()
If frm3Check8.Value = 1 Then
frm3Check7.Enabled = False
Else: frm3Check7.Enabled = True
End If

```

```

End Sub

Private Sub frm3Check9_Click()

If frm3Check9.Value = 1 Then
frm3Check10.Enabled = False
Else: frm3Check10.Enabled = True
End If
End Sub

```

```

Private Sub frm3cmd1_Click()
frm2.Show
frm3.Hide

```

```

End Sub

```

### BACK BUTTON CODE

```

Private Sub frm3cmd1_Click()
frm2.Show
frm3.Hide

```

```

End Sub

```

### SAVE BORM DATA BUTTON CODE (SENDS BORM DATA TO FINAL FORM 4)

```

Private Sub frm3cmd2_Click()
If frm3frame1txt7.Text = frm2.ucframe1.Caption Then

frm4.bpwstep11.Text = frm3.frm3frame1combo1.Text + " " +
frm3.frm3frame1txt1.Text + " " + frm3.frm3frame1combo2.Text
frm4.bpwstep12.Text = frm3.frm3frame1combo3.Text + " " +
frm3.frm3frame1txt2.Text + " " + frm3.frm3frame1combo4.Text
frm4.bpwstep13.Text = frm3.frm3frame1combo5.Text + " " +
frm3.frm3frame1txt3.Text + " " + frm3.frm3frame1combo6.Text
frm4.bpwstep14.Text = frm3.frm3frame1combo7.Text + " " +
frm3.frm3frame1txt4.Text + " " + frm3.frm3frame1combo8.Text
frm4.bpwstep15.Text = frm3.frm3frame1combo9.Text + " " +
frm3.frm3frame1txt5.Text + " " + frm3.frm3frame1combo10.Text
frm4.bpwstep16.Text = frm3.frm3frame1combo11.Text + " " +
frm3.frm3frame1txt6.Text + " " + frm3.frm3frame1combo12.Text
End If

```

```

If frm3frame1txt7.Text = frm2.ucframe2.Caption Then

frm4.bpwstep21.Text = frm3.frm3frame1combo1.Text + " " +
frm3.frm3frame1txt1.Text + " " + frm3.frm3frame1combo2.Text
frm4.bpwstep22.Text = frm3.frm3frame1combo3.Text + " " +
frm3.frm3frame1txt2.Text + " " + frm3.frm3frame1combo4.Text
frm4.bpwstep23.Text = frm3.frm3frame1combo5.Text + " " +
frm3.frm3frame1txt3.Text + " " + frm3.frm3frame1combo6.Text
frm4.bpwstep24.Text = frm3.frm3frame1combo7.Text + " " +
frm3.frm3frame1txt4.Text + " " + frm3.frm3frame1combo8.Text
frm4.bpwstep25.Text = frm3.frm3frame1combo9.Text + " " +
frm3.frm3frame1txt5.Text + " " + frm3.frm3frame1combo10.Text

```





```

frm4.bpwstep102.Text = frm3.frm3frame1combo3.Text + " " +
frm3.frm3frame1txt2.Text + " " + frm3.frm3frame1combo4.Text
frm4.bpwstep103.Text = frm3.frm3frame1combo5.Text + " " +
frm3.frm3frame1txt3.Text + " " + frm3.frm3frame1combo6.Text
frm4.bpwstep104.Text = frm3.frm3frame1combo7.Text + " " +
frm3.frm3frame1txt4.Text + " " + frm3.frm3frame1combo8.Text
frm4.bpwstep105.Text = frm3.frm3frame1combo9.Text + " " +
frm3.frm3frame1txt5.Text + " " + frm3.frm3frame1combo10.Text
frm4.bpwstep106.Text = frm3.frm3frame1combo11.Text + " " +
frm3.frm3frame1txt6.Text + " " + frm3.frm3frame1combo12.Text

End If

```

```

If frm3frame1txt7.Text = frm2.ucframe12.Caption Then

```

```

frm4.bpwstep201.Text = frm3.frm3frame1combo1.Text + " " +
frm3.frm3frame1txt1.Text + " " + frm3.frm3frame1combo2.Text
frm4.bpwstep202.Text = frm3.frm3frame1combo3.Text + " " +
frm3.frm3frame1txt2.Text + " " + frm3.frm3frame1combo4.Text
frm4.bpwstep203.Text = frm3.frm3frame1combo5.Text + " " +
frm3.frm3frame1txt3.Text + " " + frm3.frm3frame1combo6.Text
frm4.bpwstep204.Text = frm3.frm3frame1combo7.Text + " " +
frm3.frm3frame1txt4.Text + " " + frm3.frm3frame1combo8.Text
frm4.bpwstep205.Text = frm3.frm3frame1combo9.Text + " " +
frm3.frm3frame1txt5.Text + " " + frm3.frm3frame1combo10.Text
frm4.bpwstep206.Text = frm3.frm3frame1combo11.Text + " " +
frm3.frm3frame1txt6.Text + " " + frm3.frm3frame1combo12.Text

End If

```

```

If frm3frame1txt7.Text = frm2.ucframe9.Caption Then

```

```

frm4.bpwstep301.Text = frm3.frm3frame1combo1.Text + " " +
frm3.frm3frame1txt1.Text + " " + frm3.frm3frame1combo2.Text
frm4.bpwstep302.Text = frm3.frm3frame1combo3.Text + " " +
frm3.frm3frame1txt2.Text + " " + frm3.frm3frame1combo4.Text
frm4.bpwstep303.Text = frm3.frm3frame1combo5.Text + " " +
frm3.frm3frame1txt3.Text + " " + frm3.frm3frame1combo6.Text
frm4.bpwstep304.Text = frm3.frm3frame1combo7.Text + " " +
frm3.frm3frame1txt4.Text + " " + frm3.frm3frame1combo8.Text
frm4.bpwstep305.Text = frm3.frm3frame1combo9.Text + " " +
frm3.frm3frame1txt5.Text + " " + frm3.frm3frame1combo10.Text
frm4.bpwstep306.Text = frm3.frm3frame1combo11.Text + " " +
frm3.frm3frame1txt6.Text + " " + frm3.frm3frame1combo12.Text

End If

```

```

If frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check1.Value = 1 Then
frm4.bpwstatus11.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then
frm4.bpwstatus11.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check2.Value = 1 Then
frm4.bpwstatus11.Text = "S"

End If

```

```

End If

```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3Check3.Value = 1 Then
frm4.bpwstatus12.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then
frm4.bpwstatus12.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check4.Value = 1 Then
frm4.bpwstatus12.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check5.Value = 1 Then
frm4.bpwstatus13.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then
frm4.bpwstatus13.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check6.Value = 1 Then
frm4.bpwstatus13.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check7.Value = 1 Then
frm4.bpwstatus14.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then
frm4.bpwstatus14.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check8.Value = 1 Then
frm4.bpwstatus14.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check9.Value = 1 Then
frm4.bpwstatus15.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
frm4.bpwstatus15.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check10.Value = 1 Then
frm4.bpwstatus15.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check11.Value = 1 Then
frm4.bpwstatus16.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then
frm4.bpwstatus16.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe1.Caption And
frm3.frm3Check12.Value = 1 Then
frm4.bpwstatus16.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus21.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus21.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check2.Value = 1 Then  
frm4.bpwstatus21.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus22.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus22.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus22.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check5.Value = 1 Then  
frm4.bpwstatus23.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then  
frm4.bpwstatus23.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check6.Value = 1 Then  
frm4.bpwstatus23.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check7.Value = 1 Then  
frm4.bpwstatus24.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then  
frm4.bpwstatus24.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check8.Value = 1 Then  
frm4.bpwstatus24.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check9.Value = 1 Then  
frm4.bpwstatus25.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then  
frm4.bpwstatus25.Text = ""
```



```
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check10.Value = 1 Then  
frm4.bpwstatus25.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check11.Value = 1 Then  
frm4.bpwstatus26.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then  
frm4.bpwstatus26.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe2.Caption And  
frm3.frm3Check12.Value = 1 Then  
frm4.bpwstatus26.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus31.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus31.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check2.Value = 1 Then  
frm4.bpwstatus31.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus32.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus32.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus32.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check5.Value = 1 Then  
frm4.bpwstatus33.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then  
frm4.bpwstatus33.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check6.Value = 1 Then  
frm4.bpwstatus33.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check7.Value = 1 Then
```

```
frm4.bpwstatus34.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then  
frm4.bpwstatus34.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check8.Value = 1 Then  
frm4.bpwstatus34.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check9.Value = 1 Then  
frm4.bpwstatus35.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then  
frm4.bpwstatus35.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check10.Value = 1 Then  
frm4.bpwstatus35.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check11.Value = 1 Then  
frm4.bpwstatus36.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then  
frm4.bpwstatus36.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe3.Caption And  
frm3.frm3Check12.Value = 1 Then  
frm4.bpwstatus36.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus41.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus41.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And  
frm3.frm3Check2.Value = 1 Then  
frm4.bpwstatus41.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus42.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus42.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus42.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check5.Value = 1 Then
frm4.bpwstatus43.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then
frm4.bpwstatus43.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check6.Value = 1 Then
frm4.bpwstatus43.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check7.Value = 1 Then
frm4.bpwstatus44.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then
frm4.bpwstatus44.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check8.Value = 1 Then
frm4.bpwstatus44.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check9.Value = 1 Then
frm4.bpwstatus45.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
frm4.bpwstatus45.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check10.Value = 1 Then
frm4.bpwstatus45.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check11.Value = 1 Then
frm4.bpwstatus46.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then
frm4.bpwstatus46.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe4.Caption And
frm3.frm3Check12.Value = 1 Then
frm4.bpwstatus46.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And
frm3.frm3Check1.Value = 1 Then
frm4.bpwstatus51.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then
frm4.bpwstatus51.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And
frm3.frm3Check2.Value = 1 Then
frm4.bpwstatus51.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus52.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus52.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus52.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check5.Value = 1 Then  
frm4.bpwstatus53.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then  
frm4.bpwstatus53.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check6.Value = 1 Then  
frm4.bpwstatus53.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check7.Value = 1 Then  
frm4.bpwstatus54.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then  
frm4.bpwstatus54.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check8.Value = 1 Then  
frm4.bpwstatus54.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check9.Value = 1 Then  
frm4.bpwstatus55.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then  
frm4.bpwstatus55.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check10.Value = 1 Then  
frm4.bpwstatus55.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check11.Value = 1 Then  
frm4.bpwstatus56.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then  
frm4.bpwstatus56.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe5.Caption And  
frm3.frm3Check12.Value = 1 Then
```

```

frm4.bpwstatus56.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check1.Value = 1 Then
frm4.bpwstatus61.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then
frm4.bpwstatus61.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check2.Value = 1 Then
frm4.bpwstatus61.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3Check3.Value = 1 Then
frm4.bpwstatus62.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then
frm4.bpwstatus62.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check4.Value = 1 Then
frm4.bpwstatus62.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check5.Value = 1 Then
frm4.bpwstatus63.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then
frm4.bpwstatus63.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check6.Value = 1 Then
frm4.bpwstatus63.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check7.Value = 1 Then
frm4.bpwstatus64.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then
frm4.bpwstatus64.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check8.Value = 1 Then
frm4.bpwstatus64.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check9.Value = 1 Then
frm4.bpwstatus65.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
frm4.bpwstatus65.Text = ""

```

```
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And  
frm3.frm3Check10.Value = 1 Then  
frm4.bpwstatus65.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And  
frm3.frm3Check11.Value = 1 Then  
frm4.bpwstatus66.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And  
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then  
frm4.bpwstatus66.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe6.Caption And  
frm3.frm3Check12.Value = 1 Then  
frm4.bpwstatus66.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus71.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus71.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check2.Value = 1 Then  
frm4.bpwstatus71.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus72.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus72.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus72.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check5.Value = 1 Then  
frm4.bpwstatus73.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then  
frm4.bpwstatus73.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check6.Value = 1 Then  
frm4.bpwstatus73.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check7.Value = 1 Then
```

```
frm4.bpwstatus74.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then  
frm4.bpwstatus74.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check8.Value = 1 Then  
frm4.bpwstatus74.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check9.Value = 1 Then  
frm4.bpwstatus75.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then  
frm4.bpwstatus75.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check10.Value = 1 Then  
frm4.bpwstatus75.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check11.Value = 1 Then  
frm4.bpwstatus76.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then  
frm4.bpwstatus76.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe7.Caption And  
frm3.frm3Check12.Value = 1 Then  
frm4.bpwstatus76.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus81.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus81.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check2.Value = 1 Then  
frm4.bpwstatus81.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus82.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus82.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus82.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check5.Value = 1 Then  
frm4.bpwstatus83.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then  
frm4.bpwstatus83.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check6.Value = 1 Then  
frm4.bpwstatus83.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check7.Value = 1 Then  
frm4.bpwstatus84.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then  
frm4.bpwstatus84.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check8.Value = 1 Then  
frm4.bpwstatus84.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check9.Value = 1 Then  
frm4.bpwstatus85.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then  
frm4.bpwstatus85.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check10.Value = 1 Then  
frm4.bpwstatus85.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check11.Value = 1 Then  
frm4.bpwstatus86.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then  
frm4.bpwstatus86.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe8.Caption And  
frm3.frm3Check12.Value = 1 Then  
frm4.bpwstatus86.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus91.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus91.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And  
frm3.frm3Check2.Value = 1 Then
```



```

frm4.bpwstatus91.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3Check3.Value = 1 Then
frm4.bpwstatus92.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then
frm4.bpwstatus92.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check4.Value = 1 Then
frm4.bpwstatus92.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check5.Value = 1 Then
frm4.bpwstatus93.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then
frm4.bpwstatus93.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check6.Value = 1 Then
frm4.bpwstatus93.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check7.Value = 1 Then
frm4.bpwstatus94.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then
frm4.bpwstatus94.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check8.Value = 1 Then
frm4.bpwstatus94.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check9.Value = 1 Then
frm4.bpwstatus95.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
frm4.bpwstatus95.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check10.Value = 1 Then
frm4.bpwstatus95.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check11.Value = 1 Then
frm4.bpwstatus96.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then
frm4.bpwstatus96.Text = ""

```

```
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe10.Caption And  
frm3.frm3Check12.Value = 1 Then  
frm4.bpwstatus96.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check1.Value = 1 Then  
frm4.bpwstatus101.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then  
frm4.bpwstatus101.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check2.Value = 1 Then  
frm4.bpwstatus101.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check3.Value = 1 Then  
frm4.bpwstatus102.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then  
frm4.bpwstatus102.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check4.Value = 1 Then  
frm4.bpwstatus102.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check5.Value = 1 Then  
frm4.bpwstatus103.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then  
frm4.bpwstatus103.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check6.Value = 1 Then  
frm4.bpwstatus103.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check7.Value = 1 Then  
frm4.bpwstatus104.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then  
frm4.bpwstatus104.Text = ""  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check8.Value = 1 Then  
frm4.bpwstatus104.Text = "S"
```

```
End If
```

```
If frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check9.Value = 1 Then  
frm4.bpwstatus105.Text = "A"  
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And  
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
```

```

frm4.bpwstatus105.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And
frm3.frm3Check10.Value = 1 Then
frm4.bpwstatus105.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And
frm3.frm3Check11.Value = 1 Then
frm4.bpwstatus106.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then
frm4.bpwstatus106.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe11.Caption And
frm3.frm3Check12.Value = 1 Then
frm4.bpwstatus106.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check1.Value = 1 Then
frm4.bpwstatus201.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then
frm4.bpwstatus201.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check2.Value = 1 Then
frm4.bpwstatus201.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check3.Value = 1 Then
frm4.bpwstatus202.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then
frm4.bpwstatus202.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check4.Value = 1 Then
frm4.bpwstatus202.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check5.Value = 1 Then
frm4.bpwstatus203.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then
frm4.bpwstatus203.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check6.Value = 1 Then
frm4.bpwstatus203.Text = "S"

End If

If frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check7.Value = 1 Then
frm4.bpwstatus204.Text = "A"

```

```
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then
frm4.bpwstatus204.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check8.Value = 1 Then
frm4.bpwstatus204.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check9.Value = 1 Then
frm4.bpwstatus205.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
frm4.bpwstatus205.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check10.Value = 1 Then
frm4.bpwstatus205.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check11.Value = 1 Then
frm4.bpwstatus206.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then
frm4.bpwstatus206.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe12.Caption And
frm3.frm3Check12.Value = 1 Then
frm4.bpwstatus206.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check1.Value = 1 Then
frm4.bpwstatus301.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check1.Value = 0 And frm3.frm3Check2.Value = 0 Then
frm4.bpwstatus301.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check2.Value = 1 Then
frm4.bpwstatus301.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3Check3.Value = 1 Then
frm4.bpwstatus302.Text = "A"
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check3.Value = 0 And frm3.frm3Check4.Value = 0 Then
frm4.bpwstatus302.Text = ""
ElseIf frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check4.Value = 1 Then
frm4.bpwstatus302.Text = "S"
```

End If

```
If frm3.frm3frame1txt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check5.Value = 1 Then
```

```

frm4.bpwstatus303.Text = "A"
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check5.Value = 0 And frm3.frm3Check6.Value = 0 Then
frm4.bpwstatus303.Text = ""
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check6.Value = 1 Then
frm4.bpwstatus303.Text = "S"

End If

If frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check7.Value = 1 Then
frm4.bpwstatus304.Text = "A"
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check7.Value = 0 And frm3.frm3Check8.Value = 0 Then
frm4.bpwstatus304.Text = ""
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check8.Value = 1 Then
frm4.bpwstatus304.Text = "S"

End If

If frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check9.Value = 1 Then
frm4.bpwstatus305.Text = "A"
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check9.Value = 0 And frm3.frm3Check10.Value = 0 Then
frm4.bpwstatus305.Text = ""
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check10.Value = 1 Then
frm4.bpwstatus305.Text = "S"

End If

If frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check11.Value = 1 Then
frm4.bpwstatus306.Text = "A"
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check11.Value = 0 And frm3.frm3Check12.Value = 0 Then
frm4.bpwstatus306.Text = ""
ElseIf frm3.frm3frameltxt7.Text = frm2.ucframe9.Caption And
frm3.frm3Check12.Value = 1 Then
frm4.bpwstatus306.Text = "S"

End If

End Sub

```

**FORM 4: BORM MODEL FORM**

**VALIDATE BORM MODEL BUTTON**

```
Private Sub bpwValidateBM_Click()  
  
Dim validateResponse As Integer  
  
validateResponse = MsgBox("BORM function must be superset of process  
and action include all workflow steps", vbOKCancel + vbInformation,  
"Final Model Validation")  
  
If validateResponse = vbCancel Then  
  
frm2.Show  
frm2.frm2frame1txt4.Text = ""  
frm2.frm2frame1txt5.Text = ""  
frm2.frm2frame1txt4.SetFocus  
  
End If  
  
End Sub
```