

**A HYBRID EXPERT SYSTEM APPROACH FOR
EVALUATION SYSTEMS**

**Ph.D. Thesis by
Veysi ÖZTÜRK, M.Sc.**

Department: Computer Engineering

Programme: Computer Engineering

NOVEMBER 2005

PREFACE

I would like to express my sincere gratitude to Assoc. Prof. Coşkun Sönmez for his invaluable guidance, suggestions, comments and helps during the preparation of this dissertation.

I am honored by the presence of Prof. Eşref ADALI, Assoc. Prof. Şakir Kocabaş in the dissertation monitoring committee. I also express my thankfulness for reading the dissertation, and for their comments and suggestions.

I would like to express my deepest gratitude to Prof. Dr. Ercan Öztemel for his invaluable comments and suggestions during the work in TÜBİTAK, and being in the dissertation jury.

I am grateful to all my friends in TÜBİTAK and RTP 11.13 “Realising the Potential of European Networked Simulation” project team, especially Burak Selçuk Soyer, Savaş Öztürk, Bernd Rollesbroich, Timo Hartikainen, Ali Görçin, Ali Gürbüz, Kemal Kiran, and Col. Ziya Palıgu for all kinds of their helps and their valuable support.

Last but not least I would like to thank my wife for enjoying the life with love.

November 2005

Veysi Öztürk

CONTENTS

ABBREVIATIONS	vi
LIST OF TABLES	viii
LIST OF FIGURES	ix
LIST OF SYMBOLS	xi
SUMMARY	xii
ÖZET	xiv
1. INTRODUCTION	1
1.1 Purposes of Thesis	1
1.2 Problem Definition and The Reason of Using AI for Evaluation	2
1.3 Concept Of Evaluation And Artificial Intelligence	4
1.3.1 Verification, validation and evaluation	6
1.3.2 Evaluation methods	6
1.3.3 Data collection for evaluation	7
1.3.4 The classification of evaluation	7
1.3.5 Processes including evaluation steps	10
1.3.5.1 Synthetic Environments (SEs)	10
1.3.5.2 High Level Architecture (HLA)	12
1.3.5.3 FEDEP (Federation Development and Execution Process)	13
1.3.5.4 SEDEP	15
1.3.6 Artificial Intelligence techniques	19
1.3.6.1 Expert systems	21
1.3.6.2 Fuzzy logic	24
1.3.6.3 Natural language processing	26
1.3.6.4 Machine learning	27
1.3.6.5 Genetic algorithms	27
1.3.6.6 Artificial Neural Networks (ANNs)	27
1.3.6.7 Intelligent agents	28
1.4 Summary of Previous Evaluation Studies	28
1.4.1 SIMULTAAN PASS	28
1.4.2 RTP 11.12 PES (Performance Evaluation System)	30
1.4.3 RTP 11.13 Project tools (EDST+EDT+EET)	30
1.4.4 Evaluation Expert System developed by ESTA	30

1.4.5	ESSE (Expert System for Software Evaluation)	31
1.5	The Aim of Thesis for the Lack of Previous Studies	31
1.6	Study Methodology	32
2.	CONCEPT OF COMMON EVALUATION PROCESS AND MODEL	35
2.1	Common Evaluation Process (CEP)	36
2.2	Common Evaluation Model (CEM)	39
2.3	Mapping CEP to SEDEP and FEDEP	41
2.4	Requirements Of Evaluation Tools	43
2.4.1	The features of evaluation tools	43
2.4.2	Applicable AI techniques respect to evaluation tools requirements	46
3.	INTELLIGENT EVALUATION SYSTEM (INES)	48
3.1	INES ES Knowledge Base	50
3.1.1	Knowledge elicitation and acquisition	51
3.1.2	Knowledge representation and Knowledge Base architecture	53
3.2	INES ES Inference Engine	58
3.2.1	Developing Inference Engine	59
3.2.2	Components of INES Inference Engine	61
3.2.2.1	Objective Identifier (OI)	62
3.2.2.2	Evaluation Executer (EE)	64
3.3	INES Fuzzy Logic Module	65
3.4	INES Implementation On Various Cases	65
3.4.1	AD (Air Defence) system evaluation	65
3.4.1.1	INES implementation	66
3.4.1.2	Evaluation of the hit ratio of the AD system	66
3.4.1.3	Evaluation of the AD system reaction time	67
3.4.1.4	Evaluation of the damage level in the sheltered area	68
3.4.1.5	AD System overall evaluation	71
3.4.2	Instructor evaluation	75
3.4.2.1	Instructor overall evaluation	78
3.4.3	Pilot evaluation	82
3.4.4	Personnel selection	85
3.5	Distributed Evaluation	86
3.5.1	Advantages of distributed evaluation	88
3.5.2	Knowledge Representation	89
3.6	Overall Results of Implementation Studies	90
4.	COMPARISON, (DIS)ADVANTAGES OF INES WITH SIMILAR TOOLS	92
4.1	Comparison of INES with Similar Tools	92
4.2	Advantages / Disadvantages of INES With Respect to Similar Tools	94
5.	CONCLUSIONS AND FUTURE WORK	96
5.1	Conclusions	96
5.2	Future Work	99
	REFERENCES	101

APPENDIX A USE OF INES	107
A.1 INSTALLATION INSTRUCTIONS	107
A.2 INFERENCE ENGINE	107
A.3 KNOWLEDGE BASE (KB) EDITOR	111
A.4 Evaluation Database Part	125
Sorting and searching the objectives	125
Manipulating the evaluation rules	125
Manipulating the evaluation measures	125
A.5 Options	125
A.6 The use of Knowledge Base Editor	125
APPENDIX B GLOSSARY	125
CIRRICULUM VITAE	125

ABBREVIATIONS

AD	: Air Defence
AI	: Artificial Intelligence
ANN	: Artificial Neural Network
BSc	: Bachelor of Science
CATPEF	: Computer Aided Trainee Performance Evaluation Framework
CEP	: Common Evaluation Process
CGF	: Computer Generated Forces
COTS	: Commercial Off the Shelf
CT	: Collective Training
DIS	: Distributed Interactive Simulation
DoD	: Department of Defense
DMSO	: Defense Modeling and Simulation Office
EDST	: Evaluation Definition Selection Tool
EDT	: Evaluation Definition Tool
EE	: Evaluation Executer
EET	: Evaluation Execution Tool
ES	: Expert System
ESSE	: Expert System for Software Evaluation
ESTA	: Expert System for Text Animation
FEDEP	: Federation Development and Execution Process
FL	: Fuzzy Logic
FOM	: Federation Object Model
GUI	: Graphical User Interface
GOTS	: Government Off-The-Shelf
HLA	: High Level Architecture
IE	: Inference Engine
IEEE	: Institute of Electrical and Electronics Engineers
INES	: INtelligent Evaluation System
INES ES	: INtelligent Evaluation System Expert System
INES IE	: INtelligent Evaluation System Inference Engine
INES KB	: INtelligent Evaluation System Knowledge Base
KB	: Knowledge Base
MAM	: Marmara Research Center
MR	: Mission Rehearsal
M&S	: Modelling and Simulation
MSc	: Master of Science
NLP	: Natural Language Processing
N/A	: Not Applicable
OMT	: Object Model Template
OI	: Objective Identifier
Q'n'C	: Questionnaire and Checklist
PASS	: Performance Assessment SubSystem

PES	: Performance Evaluation System
RTI	: HLA Runtime Infrastructure
RTP	: Research and Technology Project
SAT	: Systems Approach to Training
SBA	: Simulation Based Acquisition
SE	: Synthetic Environment
SEEP	: Synthetic Environment Evaluation Process
SEDEP	: Synthetic Environment Development and Exploitation Process
SME	: Subject Matter Expert
SNE	: Synthetic Natural Environment
SOM	: Simulation Object Model
STEP	: Simulation, Test and Evaluation Process
SW	: Software
TRADOC	: Training and Doctrine Command
TÜBİTAK	: The Scientific and Technical Research Council of Turkey
XML	: eXtensible Markup Language
WEAG	: Western European Armament Group

LIST OF TABLES

	<u>Page No</u>
Table 1.1 Comparison of Previous Evaluation Studies	29
Table 2.1 Comparison of CEP with SEDEP and FEDEP	42
Table 2.2 Comparison of applicable AI technologies and conventional computing with respect to some requirements of Evaluation Tools	46
Table 3.1 The rules of AD System Fuzzy Evaluation System	72
Table 3.2 The rules of Instructor Fuzzy Evaluation System	79
Table 3.3 INES' execution time for different applications	91
Table 4.1 Comparison of INES with Resembling Previous Studies	93

LIST OF FIGURES

	<u>Page No</u>
Figure 1.1	Live, virtual and constructive simulation (From www.dmsomil) 11
Figure 1.2	Federation Development and Execution Process (FEDEP) 13
Figure 1.3	Step 7- Analyze data and evaluate results [40] 14
Figure 1.4	SEDEP steps 15
Figure 1.5	“Perform Evaluation” step in SEDEP [43] 17
Figure 1.6	Expert system components and human interfaces) 22
Figure 1.7	Temperature membership functions for {Cold, Hot} 24
Figure 1.8	Main components of a fuzzy logic system 25
Figure 1.9	ESTA’s main components 31
Figure 2.1	Common Evaluation Process 37
Figure 2.2	Common Evaluation Model (adopted from [65]) 40
Figure 2.3	An instant of Common Evaluation Model (adopted from [67]) 41
Figure 3.1	The components of INES 49
Figure 3.2	The related table fields in evaluation KB [64] 56
Figure 3.3	Activity diagram of INES Inference Engine 60
Figure 3.4	An example of INES IE evaluation objectives 62
Figure 3.5	An example of INES evaluation selection 63
Figure 3.6	Air Defence System (adopted from [78]) 66
Figure 3.7	Two dimensional damage model of sheltered area and cruise missile (adopted from [78]) 68
Figure 3.8	The effective destruction in the sheltered area (adopted from [78]) 70
Figure 3.9	Screenshot of INES ES for AD System evaluation 70
Figure 3.10	Screenshot of INES for AD Systems comparison 71
Figure 3.11	Screenshot of INES ES for AD System evaluation 71
Figure 3.12	Overall results of INES ES and Fuzzy Logic Module for different membership functions 73
Figure 3.13	Membership function for input variable “HitRatio” 73
Figure 3.14	Membership function for input variable “Reaction Time” 74
Figure 3.15	Membership function for input variable “Destroyed Area” 74
Figure 3.16	AD System Fuzzy Evaluation output variable “Performance” 74
Figure 3.17	Overall result of AD System evaluation 75
Figure 3.18	Screenshot of INES ES for instructor evaluation 77
Figure 3.19	Screenshot of INES ES for instructor evaluation 78
Figure 3.20	Screenshot of INES ES for instructor evaluation 78
Figure 3.21	Screenshot of INES ES for instructor evaluation 78
Figure 3.22	The results generated by INES ES and Fuzzy Logic for

	different membership functions	80
Figure 3.23	Membership Function for input variable “Publications”	80
Figure 3.24	Membership Function for input variable “StudentTraining”	80
Figure 3.25	Membership Function for input variable “Projects”	81
Figure 3.26	Membership Function for output variable “Performance”	81
Figure 3.27	An example result of Instructor Evaluation Fuzzy System	82
Figure 3.28	Screenshot of INES’ ES for pilot performance evaluation	85
Figure 3.29	Screenshot of INES results for a person evaluation	86
Figure 3.30	Distributed evaluation using master and evaluation agents	87
Figure 3.31	Distributed evaluation using master and evaluation agents	88
Figure 3.32	The main components of master agent	90
Figure A.1	INES Inference Engine screenshot	107
Figure A.2	Options form	109
Figure A.3	Results form	110
Figure A.4	Evaluation results graphic form	111
Figure A.5	KB Editor main window	112
Figure A.6	Buttons for updating the hierarchical evaluation objectives	113
Figure A.7	Evaluation objectives	114
Figure A.8	Database navigator bar	115
Figure A.9	Sorting, searching and displaying all records of evaluation objectives	116
Figure A.10	Evaluation rules	116
Figure A.11	Evaluation measures	117
Figure A.12	Evaluation methods and parameters	119
Figure A.13	Questionnaires and checklists form	120
Figure A.14	Options form	121

LIST OF SYMBOLS

h	: altitude of detonation
R	: Distance between the center of sheltered area and Destruction Center
R_{SA}	: Radius of Sheltered Area,
R_{CM}	: Destruction Radius of Cruise Missile,
R_{DR}	: cruise missile destroy range
α, β	: the angle to calculate circle segment of the Sheltered Area

HYBRID EXPERT SYSTEM APPROACH FOR EVALUATION SYSTEMS

SUMMARY

Evaluation of systems, synthetic environments and human performance are generally complicated and time-consuming tasks. Existing evaluation systems are domain dependent and generally don't provide explanation on how the system reaches the evaluation results. Elicited new evaluation information cannot be updated to the system easily. Expertise is needed for evaluation process. In this thesis, "Common Evaluation Process" and "Common Evaluation Model", which simplify, speed up evaluation process and decrease evaluation cost, were proposed and developed.

The study indicates that it is possible to put knowledge related to evaluation into a structured format. In this scope, a methodology was developed to handle the heuristic knowledge of experts from different domains and information from different sources for evaluation purposes. In this method, the knowledge was represented as reference model of evaluation objectives, production rules, measures, methods and parameters. Using "Common Evaluation Model" was decreased the number of evaluation rules, measures and parameters to represent evaluation knowledge.

A hybrid expert-fuzzy system, called "Intelligent Evaluation System", which can be used for evaluation of trainees, instructors, job applicants, synthetic environments such as simulators, computer generated forces as well as real systems was developed based on "Common Evaluation Process", "Common Evaluation Model" and evaluation needs. As the evaluation includes uncertainty in some aspects, fuzzy logic was incorporated with expert system for reasoning. However, it was realised that Fuzzy Logic could be used to perform high level (abstract) evaluation, instead of low level evaluation. In other words, fuzzy logic can be more beneficial and more easily used for overall evaluation of main objective instead of all aspects of evaluation. Because a lot of parameters are required for a complete evaluation and writing a lot of rules for these parameters in fuzzy logic is not an efficient way. As more rules are needed for complex systems, it becomes increasingly difficult to relate these rules to the system. Therefore, fuzzy system was used at an abstract level of evaluation in this study.

Intelligent Evaluation System can be used at different domains and provides the following benefits:

- Speeding up evaluation process and decreasing evaluation cost.
- flexible structure for modeling evaluation knowledge.
- explanation on how the system reaches evaluation results.
- Allowing to update evaluation knowledge base without changing source code.

- sharing and reusability of knowledge used in evaluation process.
- Reducing the complexity associated with the evaluation.
- Modelling the uncertainty about overall evaluation and providing reasoning on linguistic variables.

“Intelligent Evaluation System” was implemented for the first time in various areas such as evaluation of Air Defence System, instructor performance, pilot performance and personnel selection.

DEĞERLENDİRME SİSTEMLERİ İÇİN MELEZ UZMAN SİSTEM YAKLAŞIMI

ÖZET

Sistemlerin, sentetik ortamların ve insan başarısının değerlendirilmesi genellikle karmaşık olup çok zaman gerektirmektedir. Mevcut değerlendirme sistemleri belli bir alana yöneliktir ve genellikle sistemin değerlendirme sonuçlarına nasıl ulaştığını açıklamazlar. Elde edilen yeni değerlendirme bilgilerinin, sistemde güncellenmesi kolay değildir. Değerlendirme süreci için uzmanlık bilgisi gerekmektedir. Bu tezde değerlendirme sürecini kolaylaştıran ve hızlandıran “Genel Değerlendirme Süreci” ve “Genel Değerlendirme Modeli” önerilmiş ve geliştirilmiştir.

Bu çalışma, değerlendirmeyle ilgili bilgilerin yapısal bir formata konabileceği göstermektedir. Farklı alanlardaki uzmanlardan elde edilen sezgisel bilgilerin ve farklı kaynaklardan elde edilen verilerin işlenebilmesi için bir yöntem geliştirildi. Bu yöntemde, değerlendirme bilgileri değerlendirme amaçlarının, değerlendirme kurallarının, ölçümlerin, metotların ve parametrelerin referans modeli olarak ifade edildi. Genel Değerlendirme Modeli'nin kullanılması, değerlendirme bilgilerinin ifade edilmesi için gerekli kuralların, ölçümlerin ve parametrelerin sayısını azalttı.

Melez uzman sistem ve bulanık mantıktan meydana gelen “Zeki Değerlendirme Sistemi”, öğrencileri, öğretmenleri, işe başvuranları, bilgisayar tarafından meydana getirilmiş kuvvetler gibi sentetik kuvvetleri değerlendirdiği gibi gerçek sistemleri de değerlendirebilmekte olup “Genel Değerlendirme Süreci”ne, “Genel Değerlendirme Modeli”ne ve değerlendirme ihtiyaçlarına göre geliştirildi. Değerlendirme bazı açılardan belirsizlik içerdiğinden, karar vermede bulanık mantıkla uzman sistemler birlikte kullanıldı. Fakat bulanık mantığın alt seviye değerlendirme yerine, üst seviye (özet) değerlendirmede kullanılabilmesi görüldü. Başka bir deyişle, değerlendirmenin temel amacına göre genel olarak yapılmasında bulanık mantığın kullanılması, tüm kısımlarına göre yapılmasına göre daha faydalıdır ve daha kolay kullanılabilir. Bunun nedeni, komple bir değerlendirmede birçok parametrenin gerekmesi ve bulanık mantıkta tüm bu parametrelerle ilgili olarak birçok kural yazmanın etkili ve kolay bir yol olmamasındandır. Karmaşık sistemler için birçok kural gerekeceğinden, sistemle bu kurallar arasında ilgi kurmanın zorluğu çok fazla artacaktır. Bunlardan dolayı bulanık mantık, bu çalışmada üst seviye değerlendirmede kullanıldı.

Zeki Değerlendirme Sistemi'nin farklı alanlarda kullanılabilir ve aşağıdaki faydaları sağlar:

- Değerlendirme sürecini hızlandırır ve değerlendirme maliyetini azaltır.
- Değerlendirme bilgilerinin modellenmesinde esnek bir yapı sağlar.

- Deęerlendirme sonularına nasıl ulaşıldığını açıklar.
- Deęerlendirme için gerekli olan bilgilerin kaynak kodu deęiştirilmeden güncellenmesine olanak sağlar.
- Deęerlendirme sürecinde kullanılan bilgilerin paylaşılmasını ve tekrar kullanılmasını sağlar.
- Deęerlendirmeyle ilgili karmaşıklığı azaltır.
- Üst seviye deęerlendirmeyle ilgili olarak belirsizliklerin modellenmesinde ve sözel ifadelerle çıkarımda kullanılabilir.

“Zeki Deęerlendirme Sistemi”, Hava Savunma Sistemi, öęretmen başarımı, pilot başarımı deęerlendirmesi ve eleman seçimi gibi çeşitli alanlara ilk defa uygulandı.

1. INTRODUCTION

Evaluation of systems, synthetic environments, human performance (e.g. instructors and trainees) is generally complicated and time-consuming [1]. Besides finding out the knowledge and formulating it, there is not a structured approach developed so far helping the evaluators to make the required evaluation. Although, a number of different evaluation methodologies exist, there is still no general "evaluation methodology" [2]. Defining such a process and methodology, which simplifies and speeds up the evaluation of synthetic environments, systems, human performances, job applicants, can obviously save cost, time and provide reusability. In this study, a common evaluation process and a methodology were developed to handle the heuristic knowledge of experts from different domains and information from different sources for evaluation purposes using Artificial Intelligence Techniques.

Artificial intelligence (AI) is a very powerful decision-making technology. It has been widely applied to military applications and non-military applications [3]. Application of AI for evaluation purposes can be beneficial in various aspects. Some of these benefits are given in the following section.

1.1 Purposes of Thesis

The purposes of thesis are as follows:

- To define a common evaluation process and methodology that simplifies and speeds the evaluation of synthetic environments, systems, and human performance.
- To develop an intelligent evaluation software based on common evaluation process and methodology for the following benefits:
 - Using domain independently for evaluation purposes
 - Providing automated intelligent evaluation
 - Reducing the time and cost required to accomplish the evaluation tasks

- Reducing complexity of evaluation of systems, synthetic environments and humans
- Providing explanation on how the system reaches the evaluation results
- Providing flexibility with regard to the applied evaluation criteria
- To implement developed evaluation software on various cases to test developed evaluation process and evaluation tool

1.2 Problem Definition and The Reason of Using AI for Evaluation

The complexity of accomplished tasks by systems, humans, and synthetic environments is increasing day by day. Evaluation is needed nearly for all engineering tasks and the obstacles related with evaluation are increased proportional with complexity. It is necessary to investigate new techniques to automate manual evaluation and to overcome the obstacles related with evaluation that cannot be solved (or very difficultly solved) with conventional computing. Some of these obstacles are as follows:

- **Systems and Synthetic Environments (SEs) are getting more complex day by day**, and it is difficult to evaluate those manually or using conventional methods. For example, trainees are performing more complex tasks that cannot be viewed at a glance to determine their state. If an instructor has several trainees to monitor, it is difficult to track what each trainee is doing at any given time. One solution to this is to offload the instructor by interpreting and evaluating the trainee actions via computer programs. Intelligent systems (e.g. rule-based) appear to be a good means to interpret what the student action is, and how it relates to the goals of the training [4].
- **Expertise is needed for evaluation process. But there are very few Subject Matter Experts (SMEs) being able to evaluate systems and SEs efficiently, especially for complex tasks.** “An SME is an individual who, by virtue of position, education, training, or experience, is expected to have greater than normal expertise or insight relative to a particular technical or operational discipline, system, or process.” SMEs are essential for simulation evaluation. SME usage occurs in all parts of simulation lifecycle [5].

Small community of SE experts is a limiting factor in the wider application of SEs [6].

Sandeep reported that Fallasen points out an experiment that was designed to determine differences in information usage by tactical planners indicated that 78% of critical facts identified by the experts were missed by the non-experts [7]. The facts missed by non-experts included timing information, actions of adjacent units, changes in boundaries, enemy activities, terrain constraints, mobility, engineering capabilities, and logistical loads.

- **Evaluation Process is ill defined by nature. It changes according to the SMEs.**

Generally, evaluations are made via the subjective observations and assessment of SMEs [8]. For example, there are no well-accepted methods for storing and manipulating simulation execution logs [9].

Sandeep reported that Tolcott points out that another critical difference between experts and non-experts is the use of uncertain information. Experts were more aware of uncertain assumptions and made explicit predictions of events that would confirm their expectations and thus confirm or disconfirm assumptions [7].

- It is important to **provide evaluation results with an understanding of the source of the problem** instead of only judgements on outcome [10]. This may help instructors, evaluators and trainees to understand where to focus future training and evaluation.
- There is a need to **objectively evaluate** systems, trainees, simulation based training scenarios [11], trainee performance [12], SEs, etc. But manual methods are generally subjective by nature.
- **Existing evaluation systems are domain dependent.** For example, pilot evaluation system cannot be used for evaluation of job applicant. A common evaluation methodology or system can provide reusability.
- As **new information can be elicited over time**, updating the required knowledge for evaluation should be done easily without changing source code.

In this study it was shown (provided) that AI technology especially expert systems and integrated “expert system and fuzzy logic” could be used to overcome these obstacles related with evaluation that can’t be solved (or partially solved) by conventional computing and manual evaluation.

1.3 Concept Of Evaluation And Artificial Intelligence

Evaluation is a general term referring to the collection and processing of information and data in order to compare events, which have taken place (e.g. effects caused by a new technology) to a set of normative criteria or goals. This can be done in a number of different ways and with regard to a number of different classes of objects (e.g. technologies, projects, policies etc.).

Evaluation is a means to strengthen development, whether it is human, economic, or other forms of development. Sansers reported from Scriven “*Without such a process, there is no way to distinguish the worthwhile from the worthless.*” [13].

Some more definitions of evaluation are as follows:

“Evaluation is the process of determining significance or worth, usually by careful appraisal and study.” [14].

“Evaluation is the process of determining the worth or value of something. This involves assigning values to the thing or person being evaluated.” [14].

“Evaluation is the systematic acquisition and assessment of information to provide useful feedback about some object. Term 'object' refers to a program, policy, technology, person, need, activity, synthetic environment and so on. The definition emphasizes acquiring and assessing information rather than assessing worth or merit because all evaluation work involves collecting and sifting through data, making judgements about the validity of the information and of inferences we derive from it, whether or not an assessment of worth or merit results. The generic goal of most evaluations is to provide "useful feedback" to a variety of audiences including sponsors, donors, client-groups, administrators, staff, and other relevant constituencies. Most often, feedback is perceived as "useful" if it aids in decision-making” [15].

There could be two major uses of evaluation [16]:

- Determination of the best method of delivering training, mission rehearsal, etc

- Assessment of the progress made by trainees

Different levels of evaluation may include [81]:

1. **simple** display of data without analysis (e.g. charts, diagrams, lists)
2. **analysis of data**, (e.g. Number of hits and failure)
3. **evaluation of data**, (e.g. using AI (fuzzy logic, neural networks, etc), comparison)
4. **assessment**: The highest level would include the judging of the data, (e.g. ‘this trainee had an excellent (or good) performance’ etc.)

Keeping these in mind, the evaluation comprises:

- **Evaluation Objectives** indicating what is going to be evaluated. For example: Trainee performance, effectiveness and efficiency evaluation etc.
- **The data** indicating the type of data and their precision (if applicable), units (if applicable).
- **The rules, measures and methods** to perform evaluation objectives.
- **Evaluation results** to execute evaluation criteria, measure, methods and parameters with real or synthetic data.
- **User interface** to present the evaluation results to the user.

Some examples of evaluation that could be found in the literature are as follows:

- Evaluation of event management performances of senior police officers [17],
- Pilot evaluation [18],
- Simulation based training scenarios evaluation [11],
- Collaborative virtual environments performance evaluation [19],
- HLA Run-Time Infrastructure (RTI) implementations evaluation [20],
- Software evaluation [21], etc.

1.3.1 Verification, validation and evaluation

“Verification”, “validation” and “evaluation” terms are generally confused.

“*Verification* of a system is the task of determining that the system is built according to its specifications. *Validation* is the process of determining that the system actually fulfills the purpose for which it was intended. *Evaluation* reflects the acceptance of the system by the end users and its performance in the field.” [22].

Shortly, verification is to build the system right, validation is to build the right system and evaluation of system is to assess the system.

For an example, the definition of “SE (Synthetic Environment) Verification”, “SE validation” and “SE evaluation” are given below.

“SE validation is the process of determining the degree to which a model or simulation is an accurate representation of the real world from the perspective of the intended uses of the model or simulation.” [23].

“SE verification is the process of determining that a model or simulation implementation accurately represents the developer's conceptual description and specification. Verification also evaluates the extent to which the model or simulation has been developed using sound and established software engineering techniques” [23].

And evaluation of SEs is the process of determining SEs’ significance, worth or value by careful appraisal and study to provide useful feedback.

1.3.2 Evaluation methods

The following methods are used to accomplish evaluation through analysing the collected data [24]:

- Processing data using mathematical algorithms
- Comparing (raw data or processed data) with the data from former SEs, executions and other systems
- Comparing data (raw data or processed data) to SME’s target values (e.g. standards)
- Verifying the right patterns of operation through check lists

- Evaluation by SMEs: Some evaluation objectives are difficult or impossible to measure (e.g. verbal communication) and thus to evaluate. These measures are evaluated by SMEs

The evaluation method(s) depend(s) on [24]:

- Application Domain and Type
- Evaluation Objectives
- Syllabus: Correlate the defined events with the actions of Operator or Actor

1.3.3 Data collection for evaluation

In general, data collection activity is associated with most systems, whereby data is collected to fulfill some predefined evaluation requirements for analysis. The analysis may be related with evaluating performance or behaviour of the system users, or concerned with the performance or reliability of the system itself. The data collection activity depends on the nature of the SE application and the requirements determined by the application managers [25]. Generally the data required for evaluation is collected with triggered events (e.g. shooting time), by time (repetitive measure) or automatic systems such as sensors during the SE execution.

Evaluators can also collect information from surveys/questionnaires, checklists, structured interviews, and on-site observations [26].

1.3.4 The classification of evaluation

There are many categories of evaluation. They depend on the object being evaluated, the methods used for evaluation and the purpose of the evaluation. One classification is based on the nature of evaluation; qualitative and quantitative.

Qualitative evaluation:

“Qualitative evaluation is an assessment process that answers the question, ‘How well did we do?’” [14].

It can be conducted in a qualitative way, simply by recording incidental comments of trainees, instructors, evaluators and SMEs. This can be combined with formal methods, e.g. structured interviews, checklists, questionnaires [16].

Here are some examples of qualitative evaluations and questions that can be asked for collecting related data [14]:

- Content, quality, and relevance of a program, exercise or lesson
 - What was learned?
 - Are the learners using their new knowledge? If so, how?
- Attitudes and achievements of the learners
 - What do the trainees think about, the instructors, and the materials?
- Selection, training, attitude, and ability of instructors and other personnel
 - Did the instructors do a good job of communicating the new information?
 - Did they respect and support the learners?
- Quality of resources
 - Do people in the community like the materials?
 - Do they think the materials are appropriate for each group of learners?
 - Do they find information they want to learn in the resources?
- Efficiency of strategies and activities
 - Do people in the community think the program is successful?
 - Which activities do they think are good or not good?
- Costs in relation to what was achieved
 - Do people in the community think the results of the program are worth the cost and energy that were necessary to get the program started and to keep it going?

Quantitative evaluation:

“Quantitative evaluation is an assessment process that answers the question, ‘How much did we do?’” [14].

Here are some examples of quantitative evaluations:

- User performance
 - Test scores
 - Time to complete some operation (i.e. reaction time)
 - Number of errors and where they occurred
 - Transfer of training
- Costs
 - In relation to number of students, material, and instructors
 - SE developing costs
- Time
 - SE development time
 - Training time

Another classification explained below is based on method of judging the worth of a program.

“Formative evaluation is a method of judging the worth of a program while the program activities are forming or happening. Formative evaluation focuses on the process.” [27].

Here are some examples of formative evaluation;

- Collecting continuous feedback from participants in a program in order to revise the program as required
- To assess an exercise and trainee as the exercise progresses

“Summative evaluation is a method of judging the worth of a program at the end of the program activities. The focus is on the outcome.” [27]. Summative evaluation is typically quantitative, using numeric scores or results to assess achievement. For example, learner A’s transfer of training is % 90.

“Pretraining evaluation is a method of judging the worth of a program before the program activities begin.” [28].

Here are some objectives of pretraining evaluation:

- To determine the appropriateness of the context of an activity
- To help you define relevant objectives

1.3.5 Processes including evaluation steps

In this thesis, evaluation of synthetic environments was firstly focused and synthetic environment evaluation processes were investigated. FEDEP (Federation Development and Execution Process) and SEDEP (Synthetic Environment Development & Exploitation Process) evaluation steps were examined in detail and given in the following sections. SEDEP and FEDEP aim to develop, execution and evaluation synthetic environments.

1.3.5.1 Synthetic Environments (SEs)

The definition of SEs provided by DMSO (Defence Modelling & Simulation Office) is as follows [23]:

“Synthetic Environments (SE) are internetted simulations that represent activities at a high level of realism from simulations of theaters of war to factories and manufacturing processes. These environments may be created within a single computer or a vast distributed network connected by local and wide area networks and augmented by super-realistic special effects and accurate behavioral models. They allow visualization of and immersion into the environment being simulated.”

In UK MoD glossary [29], Synthetic Environment Technology is defined as follows:

“Synthetic Environment Technology links a combination of models, simulations, people and real equipment into a common representation of the world. Internetted simulations that represents activities at a high level of realism from simulations of theaters of war to factories and manufacturing processes.”

Typical SEs include the following elements [30]:

- Computer Generated Forces (CGF)
- Simulators

- Synthetic Natural Environment (SNE)
- Run time infrastructures and Networks e.g. DIS, HLA

SEs can combine three different types of participants in representative play as shown Figure 1.1. These are [31]:

- **Live Simulation:** A simulation involving real people operating real systems.
- **Virtual Simulation:** A simulation involving real people operating simulated systems.
- **Constructive Model or Simulation:** Models and simulations that involve simulated people operating simulated systems. Real people stimulate (make inputs) to such simulations, but are not involved in determining the outcomes.

Using SEs has several advantages [32]:

- Enable participants to rehearsal their missions, before being engaged.
- Reduce costs by avoiding real deployment. If the simulation is “realistic” enough, there is little associated decrease in the effectiveness of the training [33].
- Record missions for learning purposes.
- Attempt diverse combinations for evaluation purposes, e.g. assessing the decisions impact for several times and within different contexts.
- SEs can reduce time, risk and cost throughout the procurement process from concept definition, through development, production and acceptance testing [34].

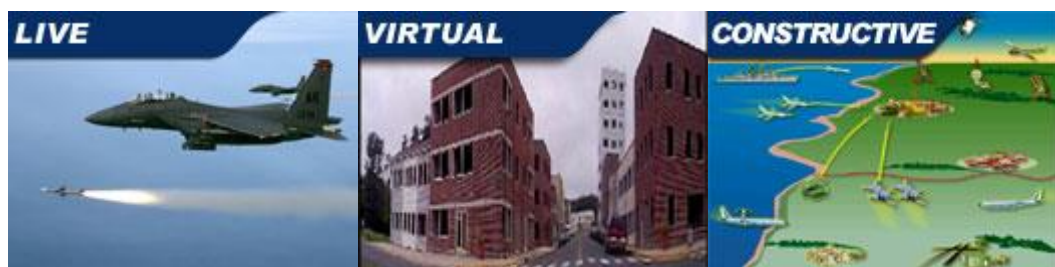


Figure 1.1 Live, virtual and constructive simulation (From www.dmsomil)

- SEs permit operational or engineering tests, which would not be possible (due to safety, environmental or cost limitations) in the real world [34].

Three types of SE users can be identified as [35]:

- Problem setters – “the individuals or group that pose the question to be answered in the SE. i.e. the customer who is responsible for defining the problem and for funding the means to obtain the solution.”
- Problem solvers – “the individuals or group that investigates the SE solution to the problem i.e. the system engineers that define the SE architecture.”
- SE Implementers – “the individuals or group that develop and integrate the various elements of an SE i.e. the specialist engineers who provide an operational and fully tested SE.”

1.3.5.2 High Level Architecture (HLA)

The High Level Architecture (HLA) is a general-purpose architecture for reuse and interoperation of simulations [36]. “The HLA is based on the premise that no single simulation can satisfy the requirements of all uses and users.” [37]. An individual simulation or set of simulations developed for one purpose can be applied to another application under the HLA concept of the federation: a composable set of interacting simulations.

HLA defines some terms such as [38]:

- A *federation* is the combined simulation system developed from the constituent simulations.
- Each simulation that is combined to form a federation is a *federate*.
- A *federation execution* is a session of a federation executing together.

The HLA has wide applicability, across a full range of simulation application areas, including education and training, analysis, engineering and even entertainment, at a variety of levels of resolution.

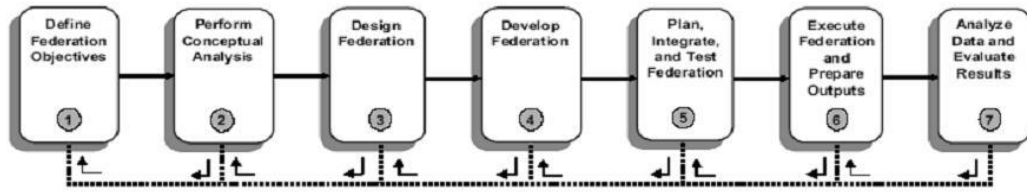


Figure 1.2 Federation Development and Execution Process (FEDEP)

1.3.5.3 FEDEP (Federation Development and Execution Process)

From the earliest HLA profederation experiences through current federation implementations, two needs have been evident— guidance in the process of developing and automating HLA federations and automation to support that process.

To meet the first need, the Federation Development and Execution Process (FEDEP) documentation was developed [39].

The types and sequence of low-level activities required to develop analysis-oriented federations is likely to be quite different from those required to develop distributed training exercises. However, at a more abstract level, it is possible to identify a sequence of seven very basic steps that all HLA federations should follow to develop and execute their federations. Figure 1.2 illustrates each of these steps and is listed as follows [40]:

Step 1: Define federation objectives.

Step 2: Perform conceptual analysis.

Step 3: Design federation.

Step 4: Develop federation.

Step 5: Plan, integrate, and test federation.

Step 6: Execute federation and prepare outputs.

Step 7: Analyze data and evaluate results.

FEDEP Step 7 is related with evaluation and explained in detail in this section. The purpose of step 7 of the FEDEP is to analyze and evaluate the data acquired during the federation execution (Step 6), and to report the results back to the user/sponsor. This evaluation is necessary to ensure that the federation fully satisfies the requirements of the user/sponsor. The results are fed back to the user/sponsor so that

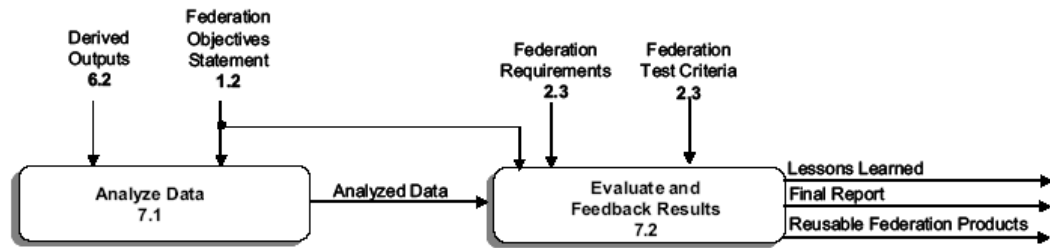


Figure 1.3 Step 7- Analyze data and evaluate results [40]

they can decide if the federation objectives have been met, or if further work is required. In the latter case, it will be necessary to repeat some of the FEDEP steps again with modifications to the appropriate federation products. Figure 1.3 illustrates the key activities in step 7 of the FEDEP. “Analyze data” and “Evaluate and feedback results” activities are explained below:

Activity 7.1 (Analyze data) [40]:

“The main purpose of this activity is to analyze the derived outputs from Step 6. This data may be supplied using a range of different media (e.g., digital, video, audio), and appropriate tools and methods will be required for analyzing the data. These may be commercial off-the-shelf (COTS) or government off-the-shelf (GOTS) tools or specialized tools developed for a specific federation. The analysis methods used will be specific to a particular federation and can vary between simple observations (e.g., determining how many targets have been hit) to the use of complex algorithms (e.g., regression analysis or data mining). In addition to data analysis tasks, this activity also includes defining appropriate “pass/fail” evaluation criteria for the federation execution and defining appropriate formats for presenting results to the user/sponsor.”

Activity 7.2 (Evaluate and feedback results) [40]:

“The purpose of this activity is to determine if federation objectives have been met and to archive reusable federation products. There are two main tasks in this activity. In the first task, the derived results from the previous activity are evaluated to determine if all federation objectives have been met. This requires a retracing of execution results to the measurable set of federation requirements originally generated during conceptual analysis (Step 2) and refined in subsequent steps. Step 7 also includes evaluating the results against the federation test criteria. In the vast majority of cases, any impediments to fully satisfying federation requirements have

already been identified and resolved during the earlier federation development and integration phases. Thus, for well-designed federations, this task is merely a final check. In those rare cases in which certain federation objectives have not been fully met at this late stage of the overall process, corrective actions must be identified and implemented. This may necessitate revisiting previous steps of the FEDEP and regenerating federation results.

The second task in this activity, assuming all federation objectives have been achieved, is to store all reusable federation products in an appropriate archive for general reuse within the domain or broader HLA community.”

1.3.5.4 SEDEP

SEDEP (Synthetic Environment Development & Exploitation Process) is another important process, developed for federation development and execution. SEDEP has a close relation with FEDEP [41]. Euclid RTP11.13 (Realising the Potential of Networked Simulations in Europe) project team, which is comprises of 23 European companies across 13 Nations developed SEDEP based on FEDEP. The aim of the project was to “overcome the obstacles that prevent SEs being exploited in Europe by developing a process and an integrated set of prototype tools, which will reduce the cost and timescale of specifying, creating and utilising SEs for collective training, mission rehearsal and simulation based acquisition” [30, 41, 42]. The main

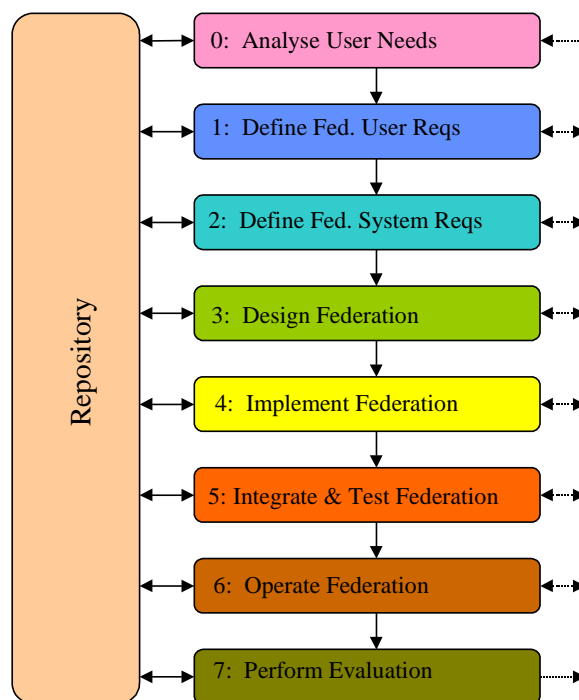


Figure 1.4 SEDEP steps

difference between SEDEP and FEDEP v1.5 is the addition of two more steps to FEDEP (Step0: Analyse User's Needs and Step7: Perform Evaluation). FEDEP IEEE Std 1516.3 has a step of "Analyze data and evaluate results" which corresponds "Perform Evaluation" of SEDEP. There are also some other minor differences in other steps.

The different activities of the SEDEP are organized in steps sequentially along the whole process. Each step covers a specific phase of the SE lifecycle. The step representation is shown in Figure 1.4 [42].

The purpose of step7 is to post-process the outputs acquired during the Federation execution, analyse them, and evaluate the results. The results are then fed back to the user to decide if the problem has been solved. This step provides support to the User to analyse and process the federation raw outputs, in order to provide the required indicators, metrics, criteria to evaluate the application domain object (trainee, mission, system).

The evaluation of synthetic environments as a SEDEP step is shown in Figure 1.5 [43]. The definitions of input items are as follows [43]:

“Evaluation User Requirements: Requirements for parameters that must be evaluated in the evaluation step, e.g. record of number of missile used against each target.

Evaluation System Requirements: The Evaluation System Requirements introduces the different metrics and indicators to be used.

Execution Outputs: Reorganized, sorted and formatted Federation Execution Outputs prepared for the evaluation post-process.”

The definitions of output items are as follows [43]:

“Execution Evaluation Results: These results provide the required information to evaluate the performance and characteristics necessary to find a solution to the problem to be solved.

Federation Evaluation Results: These results evaluate the federation capability to address the problem to be solved.

Corrective Actions: Actions to be identified and implemented when certain federation objectives have not been fully met. This may necessitate revisiting previous steps of the SEDEP and regenerating evaluation results.

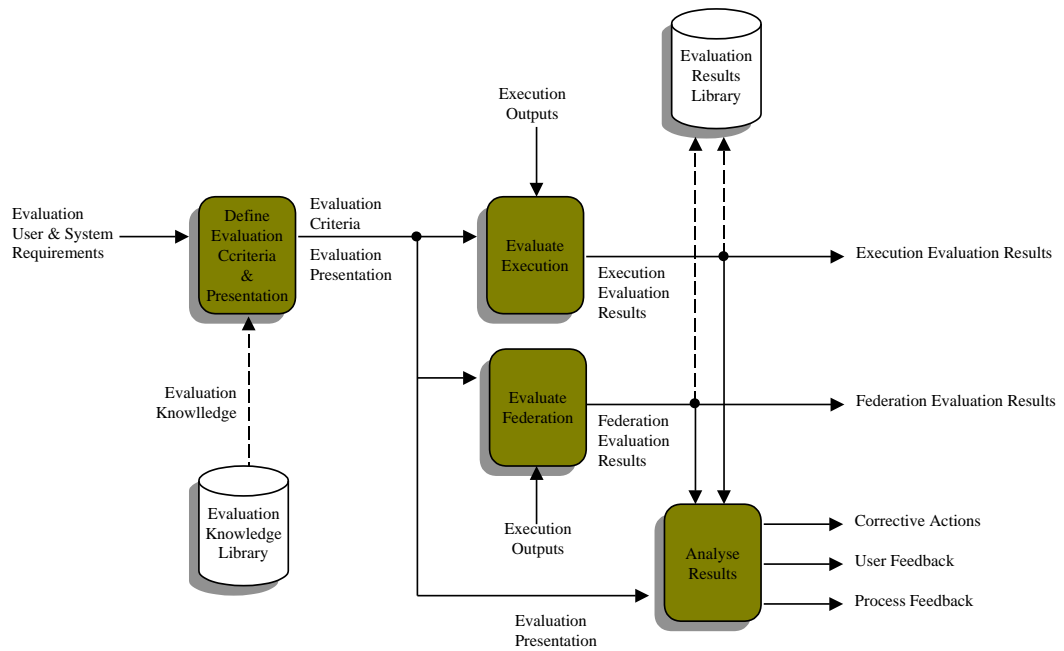


Figure 1.5 “Perform Evaluation” step in SEDEP [43]

User Feedback: Feedback to the user with the desired information defined by the previously stated Evaluation Objectives.

Process Feedback: Feedback to other process phases.”

The definitions of internal items are as follows [43]:

“Evaluation Criteria: Criteria, rules and processes to analyse and evaluate the Prepared Execution Outputs to meet the Evaluation Objectives.

Evaluation Presentation: Definition of the way how to present results to the user (graphical, textual etc.).”

The activities of “perform evaluation” step are as follows [43]:

“Define Evaluation Criteria & Presentation: The purpose of this activity is to transform (high-level) Evaluation Objectives (Evaluation User & System Requirements) into corresponding (low-level) Evaluation Criteria along with a Results Presentation definition. The Evaluation Criteria will be used in the next activity to analyse the data gathered during federation execution. The definition on how Evaluation Results will be presented to the user (textual, graphical etc.) is used in the step Distribute Results.

Evaluation Knowledge from the Evaluation Library can be used to support and simplify this activity. The library contains information – acquired previously - that links Evaluation Objectives to criteria and presentations, allowing the reuse of previously definitions.

The activity involves the following steps:

1. Selection/Identification of Evaluation Objectives for the federation respect for evaluation of data. For this step, information from User's Needs and User Requirements analysis should be considered. The Evaluation Objectives can be generic, i.e. independent of the SE application, or specific for the SE application like Collective Training.
2. Transfer of the Evaluation Objectives into Evaluation Criteria. If necessary, this involves an iterative process to get a more detailed definition of Evaluation Objectives.
3. Selection/identification of means to present results (Results Presentation).

Evaluate Execution: The purpose of this activity is to process (analyse) the outputs from the federation execution, introducing Evaluation Criteria, defined during the previous activity, to evaluate solution(s) to the problem to be solved. Such processing normally requires the application of appropriate statistical measures and other data reduction methods to transform output data into derived results (Execution Evaluation Results).

Commercial or government off-the-shelf (COTS/GOTS) statistical analysis tools and other post-processing tools are often applicable here.

1. Apply appropriate statistical measures and other data reduction methods to transform output data into derived results taking into account the Evaluation Objectives.
2. Determine if all federation objectives have been met. This requires a retracing of execution results to the measurable set of federation requirements originally generated during conceptual analysis (process step 2) (and refined in subsequent steps). This may necessitate revisiting previous steps of the SEDEP and regenerating federation results.

Evaluate Federation: The purpose of this activity is to evaluate and measure the capability of the federation to address the problem to be solved.

Analyse Results: The purpose of this activity is to compile, and present the Evaluation Results coming from the previous activity.

This activity identifies and makes public Corrective Actions that are necessary when certain federation objectives are not met.

This activity provides feedback to the user with information about the success of the federation execution in general and with information specific to the SE application, e.g. training. The feedback depends on the given Evaluation Objectives.

This activity provides feedback to the process, including information about the success of the federation execution in general and identified reusable federation products. These are made available through the Repository.”

There are also some other processes, which include evaluation such as Simulation, Test and Evaluation Process (STEP). STEP is defined as “*an iterative process that integrates both simulation and test for the purpose of evaluating the performance, military worth or effectiveness of systems to be acquired.*” [44].

AI techniques, especially expert systems and fuzzy, which are explained in the following section, were investigated in order to find the most beneficial and proper technique(s) for the development of an intelligent evaluation system.

1.3.6 Artificial Intelligence techniques

“Artificial Intelligence is the branch of computer science that is concerned with the automation of intelligent behaviour” [3].

Intelligent systems have following advantages. They can [45]:

- reduce workload.
- support objectivity in analysis and evaluation.
- reduce distraction from original task.
- provide expert/advisory assistance. Typical examples are diagnostic expert systems.

- process complex information with many parameters and/or complex dependencies between parameters. Example: Sensors data processing, process optimisation.
- support a more structured approach to solve a problem.
- provide flexibility in terms of used knowledge and/or algorithms. Especially evolving algorithms can modify themselves, allowing a system to adapt its algorithms dynamically to changing requirements and optimise its algorithms.
- be used when no exact model exists for ill-defined problems (applying rules-of-thumb in contrast to classical methods).
- explain its reasoning how it came to the provided solution (or why it did not come to a solution).
- support user friendly communication between user and computerised systems.
- improve effectiveness [46].
- increase system reliability [46].

However, there are some difficulties, challenges, shortcomings and disadvantages some of which are as follows [45]:

- Knowledge engineering (i.e. transferring human knowledge into a computer system) is difficult, time and cost consuming. Knowledge acquisition is difficult, because knowledge is highly dynamic and experiences are difficult to capture and to quantify.
- Knowledge is usually limited to special field. Therefore intelligent systems are heavily domain dependent.
- These systems generally provide satisfying but no optimal solutions as there is no mathematical model to use the problem and heuristics are to be employed.

- Depending on the selected technique (such as expert systems), intelligent systems may not be as flexible as requested, but confined to a narrow task (focus only on one specific subject).
- Intelligent systems lack common-sense-reasoning. Even highly specialised knowledge depends on general knowledge about the world.
- It is still difficult to analyse human behaviour and to model that in a computer system.
- Coupling of several expert systems to exchange knowledge is still limited.

There are several AI techniques developed and used successfully in industrial problems such as Expert Systems (ES), Fuzzy Logic (FL), artificial neural networks (ANNs), genetic algorithms (GA), Intelligent Agents (IA), robotics, computer vision, Natural Language Processing (NLP). ES, FL, ANNs, GA, IA and NLP techniques, which can be candidate of to be used for evaluation, are investigated in the following sections. Among those, ES and FL are investigated and given in detail.

1.3.6.1 Expert systems

“Expert Systems are computerized advisory programs that attempt to imitate the reasoning processes and knowledge of experts in solving specific types of problems.” [47]. “An expert system (ES) encodes deep expertise in a narrow domain of human speciality. Several expert systems have been constructed whose behavior surpasses that of humans.” [46].

Expert systems have a number of major system components and interface with individuals in various roles. These are illustrated in Figure 1.6 (adopted from [48]).

The major components of an ES are as follows [48]:

- Knowledge base - Detailed knowledge about the respective area (the "domain") is necessary for developing the AI tools. Therefore, it is necessary to collect required knowledge, to transform elicited knowledge into a machine-readable format, often in IF THEN rules, and to store the related knowledge in a structured way inside the so-called a “Knowledge Base”.

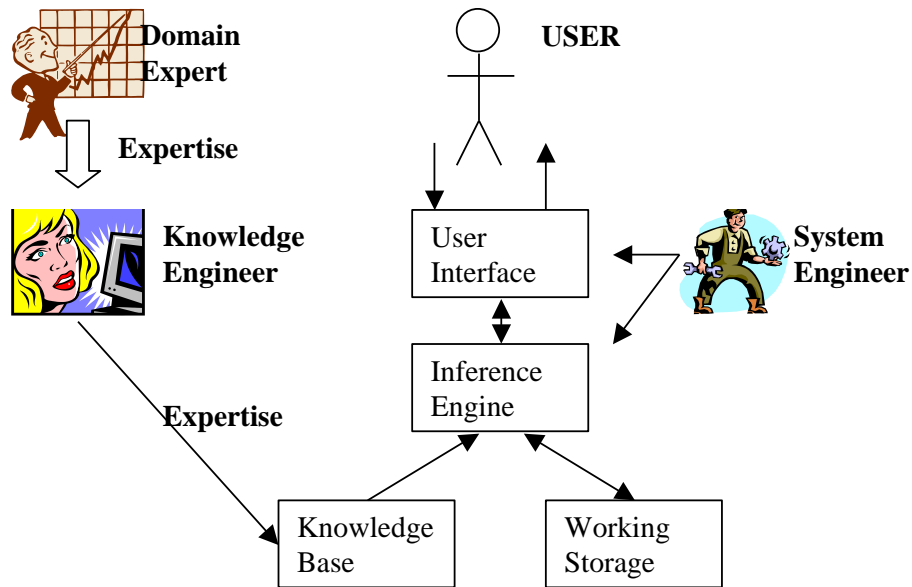


Figure 1.6 Expert system components and interfaces

A knowledge base can be represented by several different knowledge representation methods such as [49, 50]:

1. Frames
2. Semantic nets
3. Lists
4. Scripts
5. Logic (e.g. Predicate Calculus)
6. Production rules

Some of the major advantages of rule representation are as follows [50]:

- Rules are easy to understand.
- Inference and explanations are easily derived.
- Modifications and maintenance are relatively easy.
- Uncertainty is easily combined with rules.
- Each rule is usually independent of all others.

The major limitations of rule representation are as follows [50]:

- Complex knowledge requires many, many (thousands of) rules.

- Systems with many rules may have a search limitation in the control of program.
- Working storage - the data that is specific to a problem being solved.
- Inference engine - the code that is developed to make the computer to utilise knowledge and make knowledge-based intelligent decisions, search the knowledge base to find out relevant knowledge for the user's problem, and generate recommendations and solutions for the problem provided.
- User interface - the code that controls the dialog between the user and the system.

To understand expert system design, it is also necessary to understand the major roles of individuals who interact with the system. These are:

- Domain experts - the individual or individuals who currently are experts possessing the expertise and solving the problems the system is intended to solve;
- Knowledge engineer - the individual who encodes the expert's knowledge in a declarative form that can be interpreted by the expert system. The knowledge engineer has to have the knowledge of Knowledge based ES technology and should know how to develop an expert system using a development environment (Prolog, C++, Pascal, etc) or an expert system development shell [51].
- User - the individual who will be consulting with the system to get advice, which would have been provided by the expert.
- System engineer - the individual who builds the user interface, designs the declarative format of the knowledge base, and implements the inference engine.

1.3.6.2 Fuzzy logic

“Fuzzy Logic is a logical system, which aims at a formalization of approximate reasoning in a narrow sense. In a wide sense, it is coextensive with fuzzy set theory. Today, the term fuzzy logic is used predominantly in its wide sense.” [52]. “Fuzzy sets are sets in which members are presented as ordered pairs that include information on degree of membership.” [53].

Fuzzy Logic (FL) is basically developed to handle uncertainties in computer-based problem solving or decision-making. It can be used to solve highly complex problems where a mathematical model is too difficult or impossible to be formulated due to uncertainties.

Classic logic deals with true and false. But, there are statements, which cannot be stated with such certainty. Simply, let’s try to decide if today is cold or hot. In classic logic there is a threshold for hotness (for example 15 °C). If the weather is 14.9 °C then the weather is cold. If the weather is 15.1 °C then the weather is hot. In fuzzy logic, the weather can belong 0.5 degree of membership of {hot} set and 0.5 degree of membership of {cold} set as temperature membership functions for {hot, cold} is shown Figure 1.7. Fuzzy logic deals with propositions that can be true to a certain degree—somewhere from 0 to 1. Therefore, a proposition’s truth value indicates the degree of certainty about which the proposition is true. The degree of certainty sounds like a probability (perhaps subjective probability), but it is not quite the same [53].

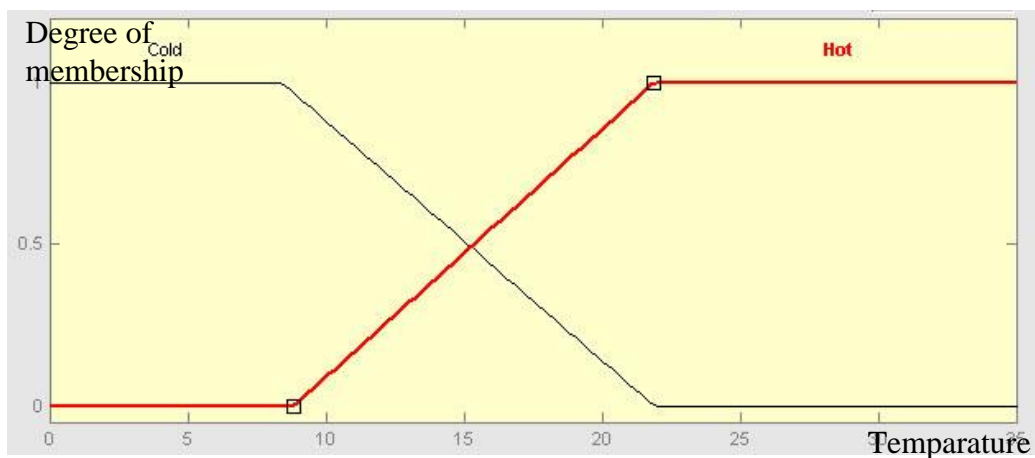


Figure 1.7 Temperature membership functions for {Cold, Hot}

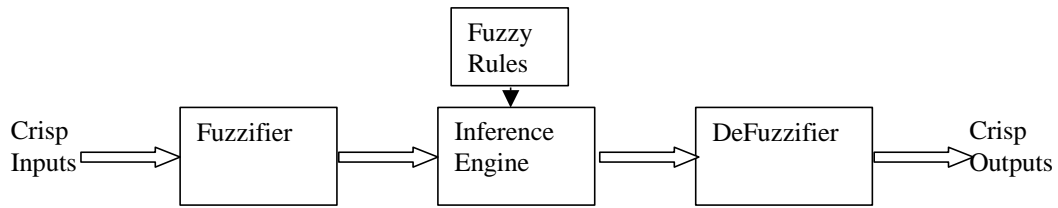


Figure 1.8 Main components of a fuzzy logic system

A fuzzy logic system consists of three main components as shown in Figure 1.8:

- **Fuzzifier** maps crisp numbers to fuzzy sets. It is needed in order to activate fuzzy rules, which are in linguistic variables and have fuzzy sets associated with them.
- **Inference engine** generates fuzzy outputs corresponding to fuzzified inputs, with respect to the fuzzy rules. Fuzzy Rules are “IF ...THEN...” form and combines inputs to outputs.
- **Defuzzifier** maps output sets into crisp numbers. For example, such a number corresponds to a voltage value for a control application.

Fuzzy sets and logic is a relatively new discipline that has proved itself successful in automated reasoning of expert systems [54].

Advantages of Fuzzy Logic

The advantages of fuzzy logic are as follows [55]:

- Fuzzy logic converts complex problems into simpler problems using approximate reasoning. The system is described by fuzzy rules and membership functions using human type language and linguistic variables. System behavior is generally defined by using knowledge of domain experts.
- A fuzzy logic description can effectively model the uncertainty and nonlinearity of a system. It is extremely difficult to develop a mathematical model of a complex system to reflect nonlinearity, uncertainty, and variation over time. Fuzzy logic avoids the complex mathematical modeling.
- Fuzzy logic is easy to implement using both software on existing microprocessors or dedicated hardware. Fuzzy logic based solutions are cost effective for a wide range of applications (such as home appliances) when compared to traditional methods.

Disadvantages of Fuzzy Logic

The disadvantages of fuzzy logic are as follows [55]:

- As the system complexity increases, it becomes more difficult to determine the correct set of rules and membership functions to describe system behavior. A significant time of investment is needed to correctly tune membership functions and adjust rules to obtain a good solution. For complex systems, more rules are needed, and it becomes increasingly difficult to relate these rules. The capability to relate the rules typically diminishes when the number of rules exceeds approximately 15. For many systems, it is impossible to find a sufficient working set of rules and membership functions.
- In addition, the use of fixed geometric-shaped membership functions in fuzzy logic limits system knowledge more in the rule base than in the membership function base. This results in requiring more system memory and processing time.
- Fuzzy logic uses heuristic algorithms for defuzzification, rule evaluation. Heuristic algorithms can cause problems mainly because heuristics do not guarantee satisfactory solutions that operate under all possible conditions. The generalization capability is important in order to handle unforeseen circumstances.
- Once the rules are determined, they remain fixed in the fuzzy logic inference engine, which is unable to learn (except in adaptive fuzzy systems, which allow some limited flexibility).
- Conventional fuzzy logic cannot generate rules (users cannot write rules) that will meet a pre-specified accuracy. Accuracy is improved only by trial and error.
- Conventional fuzzy logic does not incorporate previous state information (very important for pattern recognition, like speech recognition) in the rule base. A recurrent fuzzy logic (described later) incorporates the past information and hence is more effective for context sensitive information systems.

1.3.6.3 Natural language processing

Natural language processing (NLP) can be defined, in a very general way, as the discipline having as its ultimate, very ambitious goal that of enabling people to interact with machines using their "natural" faculties and skills [56].

NLP includes areas such as automatic text generation, text processing, machine translation, speech synthesis and analysis, grammar and style analysis of text etc.

For evaluation, NLP can be used for questionnaire analysis and automatic speech analysis of trainees such as pilots especially during team training.

1.3.6.4 Machine learning

Learning is an inherent characteristic of the human beings. By virtue of this, people, while executing similar tasks, acquire the ability to improve their performance. The principle of learning that can be adhered to machines to improve their performance is usually referred to as 'machine learning' [54].

Learning techniques can be used for developing Self-Learning Evaluation Systems.

1.3.6.5 Genetic algorithms

Genetic Algorithms are an algorithmic concept based on a survival-of-the-fittest strategy with sexual reproduction, where stronger individuals in the population have a higher chance of creating an offspring. A genetic algorithm is implemented as a computerized search and optimization procedure.

GAs are best used to find nonlinear solutions where there does not exist any previously developed mathematical or heuristic approach [56].

GA has three major applications, namely, intelligent search, optimization and machine learning [54].

GA can be used to find relevant evaluation information or knowledge from huge amount of information or data.

1.3.6.6 Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANN) are massively parallel distributed processor made up simple processing units which has a natural propensity experiential knowledge and making it available for use [57]. An ANN is an adaptive, most often nonlinear system that learns to perform a function (an input/output map) from data. Adaptive means that the system parameters are changed during operation, normally called the training phase. After the training phase the ANN parameters are fixed and the system is deployed to solve the problem at hand (testing phase) [58].

ANNs techniques can be used for evaluation of complex situations, behaviours where the relationship between input and output is unknown. For example, it can be used for tactical analysis, pilot behaviour analysis.

There are evaluation examples made by evaluators and ANNs can be beneficial for getting information from evaluation examples.

1.3.6.7 Intelligent agents

“An agent is anything that can be viewed as perceiving the environment through sensors and acting upon environment through effectors” [3]. “An intelligent agent is one that is capable of flexible autonomous action in order to meet its design objectives” [59].

More information about intelligent agents can be found in [60, 61].

Comparison of applicable AI technologies and conventional computing with respect to some requirements of Evaluation Systems is given in Section 2.4.2.

1.4 Summary of Previous Evaluation Studies

Some resembling previous studies found in the literature are as follows:

- SIMULTAAN PASS [62]
- RTP 11.12 Performance Evaluation System [63]
- RTP 11.13 EDST+EDT+EET [64, 65]
- ESTA Evaluation [66]
- ESSE (Expert System for Software Evaluation) [21]

The comparison of these studies is given in Table 1.1.

1.4.1 SIMULTAAN PASS

Arend and Jansen reported that, the SIMULTAAN project resulted into flexible and re-useable simulator/component architecture. An essential part of the SIMULTAAN architecture is the Performance Assessment SubSystem (PASS), which advises the Scenario Manager in choosing the best scenario that leads to achieving the training objectives. To accomplish this goal PASS automatically analyses and judges the trainee and team performance. The automatic performance assessment in PASS is based on a generic framework with scenario-specific expected actions and action-related judgement rules. SIMULTAAN PASS has generic HLA-based simulator

Table 1.1 Comparison of Previous Evaluation Studies

	Resembling Previous Studies				
	EDST +EDT +EET	SIMULT- AAN PASS	PES	ESTA Evaluation	ESSE
Artificial Intelligence	Partially	No	No	Yes	Yes
Application area	General (Training, Rehearsal, SBA,..)	Training	Pilot Evaluation	General (Training, Rehearsal, SBA,..)	Software Evaluation
Main purpose	Evaluation of Synthetic Environments	Automatic analysis and assessment of trainee and team performance	Evaluation of Pilot Performance	Developing expert systems	Software problem solving and software attribute assessment
Knowledge Details	Evaluation Objectives, criteria, methods and rules	Scenario- specific actions and action-related judgement rules	Pilot Evaluation Objectives & indexes, rules	Knowledge Bases	Multiple criteria, Software attributes
Knowledge Representation	Rule Based	Action-related judgement rules	Algorithmic	Rule Based	Rule Based
Updating Knowledge without source code change	Yes	Yes	No	Yes	Partially
Explanation of Results	Not Provided	Not Provided	Not Provided	Provided	Partially Provided
Maintenance and update	Easy		Difficult	Difficult	
Reasoning Capability	Yes		No	Yes	
Structure	Manually Inferencing		Control integrated with information		
Usability	Difficult		Easy	Difficult	
Learning Itself	No	No	No	No	No

architecture and thus interfaces with HLA compliant tools and simulation components [62].

Inside the training scenario the performance of the trainee or team of trainees is evaluated using a list of expected actions. Expected actions are operations or tasks that have to be performed. For each training unit (trainee or team of trainees) a list of expected actions is created. The expected actions are automatically checked off whenever all related judgement rules are true. If the trainee has to perform an action that cannot be judged automatically, the instructor has to check off this expected action manually whenever the instructor observes the correct performance of this action. This way the performance of this action is incorporated in the score.

In SIMULTAAN project, only a demonstration version was developed.

1.4.2 RTP 11.12 PES (Performance Evaluation System)

PES was developed based on the CATPEF (Computer Aided Trainee Performance Evaluation Framework). CATPEF is a set of processes describing how the performance of a pilot should be assessed after training on a certain mission on mainly simulators in a distributed simulation environment [63].

1.4.3 RTP 11.13 Project tools (EDST+EDT+EET)

In RTP 11.13 project, three tools were developed for evaluation of synthetic environments. These are:

- EDST (Evaluation Definition Selection Tool)
- EDT (Evaluation Definition Tool)
- EET (Evaluation Execution Tool)

1.4.4 Evaluation Expert System developed by ESTA

ESTA (Expert System for Text Animation) is an expert system shell, which is developed in Visual Prolog. ESTA has two main components (Control System Inference Machine and User Interface) and set of KBs as shown in Figure 1.9. By providing it with a knowledge base for a certain subject area, ESTA can be used to create an expert system for that subject [66]:

ESTA + Knowledge Base = Expert System

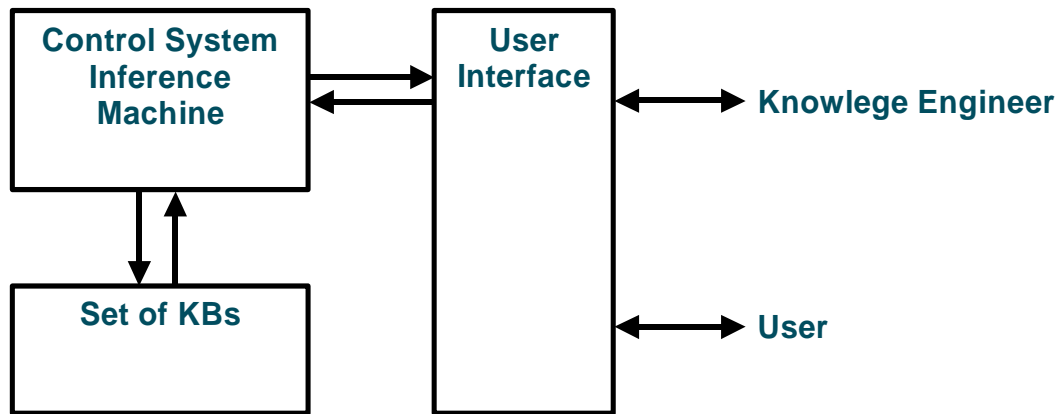


Figure 1.9 ESTA's main components

Each knowledge base contains rules for a specific domain.

ESTA has all facilities to write the rules that will make up a knowledge base. Further, ESTA has an inference engine, which can use the rules in the knowledge base to determine which advice is to be given to the expert system user or to initiate other actions. ESTA also features the ability for the expert system user to obtain answers to questions such as 'how' and 'why', etc [66].

A simple expert system for evaluation of synthetic environments was developed in ESTA (See Table 4.1 for comparison).

1.4.5 ESSE (Expert System for Software Evaluation)

ESSE is an expert system designed for software problem solving and software attribute assessment [21].

1.5 The Aim of Thesis for the Lack of Previous Studies

The lack of the previous studies and the aim of thesis for these are as follows:

- There are a lot of processes defined to standardize and speed up the tasks accomplished by synthetic environments, systems and humans. These processes generally include evaluation as a step of the whole process (e.g. FEDEP, SEDEP). In this thesis, defining a common process, which only focuses on the evaluation, was aimed. This process could be used on a wide range for evaluation purposes.
- Nearly all evaluation systems were developed according the expertise of experts. Generally, experts have heuristic evaluation knowledge of domain.

Existing evaluation systems don't provide enough reusability, knowledge share, automated evaluation for handling the heuristic knowledge of experts. For this reason, another aim of this study is developing a methodology to handle the heuristic knowledge of experts from different domains and information from different sources for evaluation purposes.

- Existing evaluation systems are domain dependent. In this thesis, developing a domain independent evaluation system is aimed. At least, the developed system can be used in a wide range.
- The evaluation tools found in the literature has limited facilities. For example, they don't provide explanation on how the system reaches to the results of evaluation. The comparison of some evaluation tools found in the literature is given in Table 1.1. Another aim of the thesis is to develop an intelligent evaluation tool for solving the following problems and obstacles of evaluation tools:
 - Providing explanation on how the system reaches the evaluation results. This problem can be solved by using AI techniques (e.g. expert systems)
 - Domain dependency
 - Difficulty in updating knowledge of system.
 - complexity of evaluation of systems, synthetic environments and humans
 - Providing flexibility with regard to the applied evaluation criteria

1.6 Study Methodology

In this study, "Common Evaluation Process" (CEP), which can be used at the assessment of synthetic environments as well as real systems and trainees, is developed. During the development of CEP, SEEP (Synthetic Environment Evaluation Process) [67], SEDEP (Synthetic Environment Development and Exploitation Process) [41], FEDEP (Federation Development and Execution Process) [40] and engineering procedures were considered. CEP is a generalized form of SEEP.

Most applicable AI technologies were investigated in order to find the most proper technique for evaluation purposes. Firstly, high-level requirements of evaluation systems are determined. AI techniques were compared with each other according to accomplishing identified evaluation systems requirements.

A hybrid expert-fuzzy software, so called INtelligent Evaluation System (INES), which can be used for evaluation of trainees, instructors, job applicants, Synthetic Environments such as simulators, Computer Generated Forces (CGF) as well as real systems was developed based on predefined evaluation needs, CEP and CEM (Common Evaluation Model). A brief description of similar previous studies and comparison of them with INES are also presented in the thesis. The detailed advantages of INES with respect to the similar tools is given in Section 4.2.

Integrated “Reference Model of Evaluation Objectives” and “Evaluation Definition Knowledge” was used to represent evaluation knowledge in INES KB. The evaluation knowledge was represented as reference model of evaluation objectives, production rules, measures, methods and parameters. Using CEM was decreased the number of evaluation rules. CEM simplifies the representation of evaluation knowledge. The method employed in building evaluation knowledge base is explained in 3.1.1.

As the evaluation includes uncertainty in some aspects, fuzzy logic was incorporated with expert system for reasoning.

INES was implemented for the first time in various areas from different domains such as evaluation of Air Defence System, instructor performance, pilot performance and personel selection.

This thesis consists of five chapters in addition to three appendixes. In Chapter 2, concept of “Common Evaluation Process” (CEP), “Common Evaluation Model”, the requirements of general purpose evaluation tools and the comparison of applicable AI technologies and conventional computing with respect to these high level requirements is explained.

In Chapter 3, INtelligent Evaluation System (INES), developed according to the “Common Evaluation Model” and the requirements of evaluation tools, is explained. The components of INES, implementation of INES on various cases and the concept

of distributed evaluation are also given in this section. This chapter is finished with the overall results of implementation studies.

In Chapter 4, the comparison, advantages and disadvantages of INES with similar tools is given.

The last chapter, conclusions of the current work, and recommendations, includes scientific contributions of the thesis, some benefits and test condition of INES, experience gained in developing INES and future work for improving INES and developing general purpose evaluation tools.

2. CONCEPT OF COMMON EVALUATION PROCESS AND MODEL

“An information system monitoring and evaluation methodology consists of the following primary phases [68]:

1. Determination of generic monitoring/evaluation objectives.
2. Translation of generic objectives into specific measurement parameters.
3. Design and implementation of the monitoring facility (the data collection mechanisms to collect data corresponding to the identified specific measurement parameters). Activation of this facility within the information system will result in the collection or generation of appropriate measurement data.
4. Selection (if available) or design and implementation of the data analysis and data presentation tools necessary for performing the required data analyses and reporting their results.
5. Design and conduct of appropriate experiments to collect the data to be analyzed.
6. Based on whatever time periodicity is dictated by the evaluation objectives, perform the data analyses, making evaluations and drawing conclusions from the results.
7. Identify system improvements and enhancements as implied by the results of the analyses and forward such to the information system staff for action.
8. Identify monitor improvements and enhancements as implied by the results of the analyses and forward such to the information system staff for action.
9. Identify experimental design improvements and enhancements as implied by the results of the analyses and forward such to the monitoring experiment staff for action.”

Common Evaluation Process (CEP) involves the phases of the methodology given above in a higher level or in detailed. CEP was developed for simplifying and speeding up the evaluation of systems, synthetic environments, and humans.

2.1 Common Evaluation Process (CEP)

CEP has four steps as shown in Figure 2.1 and is explained in detail below. CEP can be used iteratively, which means it may be initiated several times for a particular system, program, project, human and that successive iterations build on the information already available. There are also feedback loops where it may be necessary to revisit an earlier step as a result of actions performed in later ones.

Step 1 Define Evaluation

The purpose of this step is to determine evaluation definition, which includes user evaluation objectives, evaluation criteria (rules), evaluation measures, evaluation methods, evaluation parameters, questionnaires and checklists wherever applicable. Evaluation objectives are the goals of the user for performing evaluation. User evaluation objectives can be elicited from the user needs/requirements or system requirements. User evaluation objectives can be defined hierarchically as main goals and their sub-goals. Sub-goals are a set of goals to accomplish the main evaluation objective. In this step, the evaluation criteria (rules) related with the evaluation objective(s) should also be defined. Evaluation parameters indicate the type of data, their precision (if applicable) and units (if applicable) used in rules and methods. Evaluation rules are criteria used to assess the collected parameters or calculated evaluation measures. Evaluation parameters are variables needed for applying rules or calculating the result of methods. The results of methods are defined as measures in order to simplify the evaluation rules and provide reusability. Evaluation methods are the algorithms for analyzing the collected parameters or/and calculating measures used in the rules. Questionnaires and checklists are used to collect related evaluation parameters or measures values in some situations.

A typical evaluation definition is given below.

- Main Evaluation Objective is Pilot Performance Evaluation
- Evaluation Sub Objective is Pilot Destroy Success
- Measure is Destroy Ratio

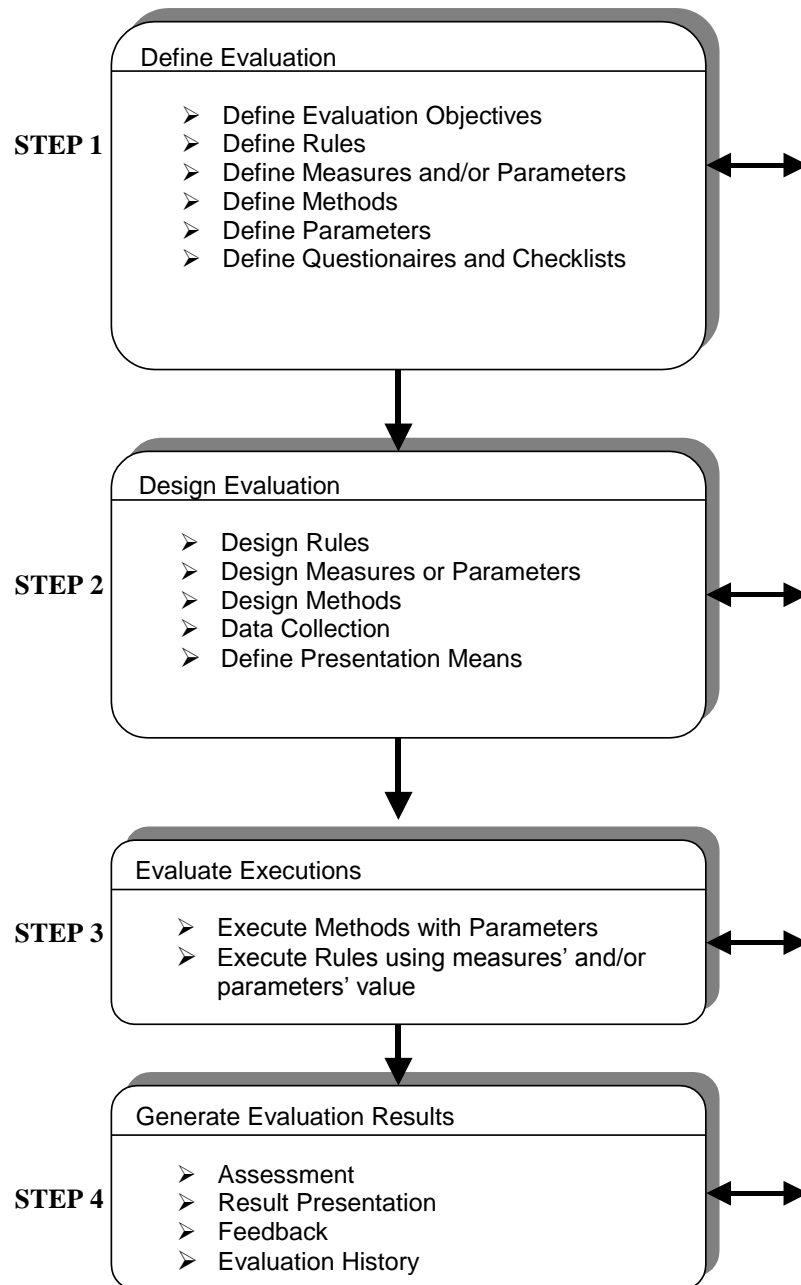


Figure 2.1 Common Evaluation Process

- Method is the ratio between number of destroyed threats and total number of threats
- Evaluation parameters are number destroyed threats and total number of threats
- Evaluation rules for this example are
 - If Destroy Ratio is smaller than 40 then Destroy Success is poor

- If Destroy Ratio is between 40 and 60 then Destroy Success is average
- If Destroy Ratio is greater than 60 then Destroy Success is good

Step 2 Design Evaluation

The purpose of this step is to design evaluation rules, measures, methods and parameters that have to be applied in the evaluation execution (Step 3) step by software means. Commercial or government off-the-shelf (COTS/GOTS) analysis tools and other post-processing tools are often applicable here. Specialized tools developed for a specific environment can also be used in this step.

A general representation for the example on evaluation definition given above is as follows:

- Evaluation Measure is Destroy_Ratio
- Evaluation Method is $\text{Destroy_Success} = (\# \text{ destroyed threats} / \text{total number of threats}) * 100$
- Related evaluation parameters are # destroyed threats, total number of threats
- Respective evaluation rules are
 - if $\text{Destroy_Ratio} < 40$ then $\text{Destroy_Success} = \text{poor}$
 - If $40 \leq \text{Destroy_Ratio} < 60$ then $\text{Destroy_Success} = \text{average}$
 - If $\text{Destroy_Ratio} \geq 60$ then Destroy_Success is good

Note that the representation of the rules, measures, methods and parameters can be changed according to the development environment such as Matlab, C++ and Delphi.

Step 3 Evaluate Executions

In this step, the execution of rules, algorithms and other data reduction or collection methods to transform output data into parameters on a given problem is required. Suitable questionnaires and checklists can also be used to collect evaluation parameters or measures.

Evaluation execution of the example given above is as follows:

- Used Evaluation Parameters;

destroyed threats = 5 total number of threats =10

- Used Method;

Destroy_Ratio= (# destroyed threats / total number of threats) *100 = 50

- Fired Evaluation rule is

- If $40 \leq \text{Destroy_Ratio} < 60$ then Destroy_Success=average

Step 4 Generate Evaluation Results

The purpose of this step is to assess the results of execution, to generate feedback to the user and to keep the history of evaluation results.

The users of SEs and systems require timely feedback on their performance for effective training and mission rehearsal [69].

The results are fed back to the user so that he/she can decide if the evaluation objectives have been met, or if further work is required.

Commercial or government off-the-shelf (COTS/GOTS) statistical and graphical tools are often applicable in this step. Specialized tools developed for a specific domain and environment can also be used in this step.

2.2 Common Evaluation Model (CEM)

A model is a simplified representation or abstraction of reality. Real problems and solutions are generally complex. The representations of systems and problems through models can be done at various degrees of abstraction [47]. Hierarchical representation is one of the methodologies to represent complex data. The importance of hierarchical data representation is so much that we could represent any present day system models based on it. Metadata representation in the hierarchy could tell a lot about the data itself [70]. Using hierarchical representation shows graphically the relationships of the problem and solution. It can deal with more complex situations in a compact form. The Common Evaluation Model developed in this thesis is a knowledge representation of the CEP and is shown in Figure 2.2. In this model, the CEP and the relationship between evaluation objectives, rules, measures, methods and parameters are taken into account. The hierarchical structure

represents the relations between objectives, which end up in a hierarchy from high-level to low-level objectives. High-level objectives are the main branches of the tree whereas the low level objectives (sub objectives) are stored as lower level branches. Evaluation objectives describe the goals of the evaluation to be performed. These can be derived from the user's needs and user/system requirements.

Each evaluation objective has related evaluation rules and different evaluation objectives can use the same evaluation rules in order to prevent duplication. In the same way, each evaluation rule is related with evaluation measures (or parameters) and different evaluation rules can use the same evaluation measures (or parameters) in order to prevent duplication of measures (or parameters). As similar, each evaluation measure is related with evaluation methods and different evaluation measures can use the same evaluation method in order to prevent duplication of

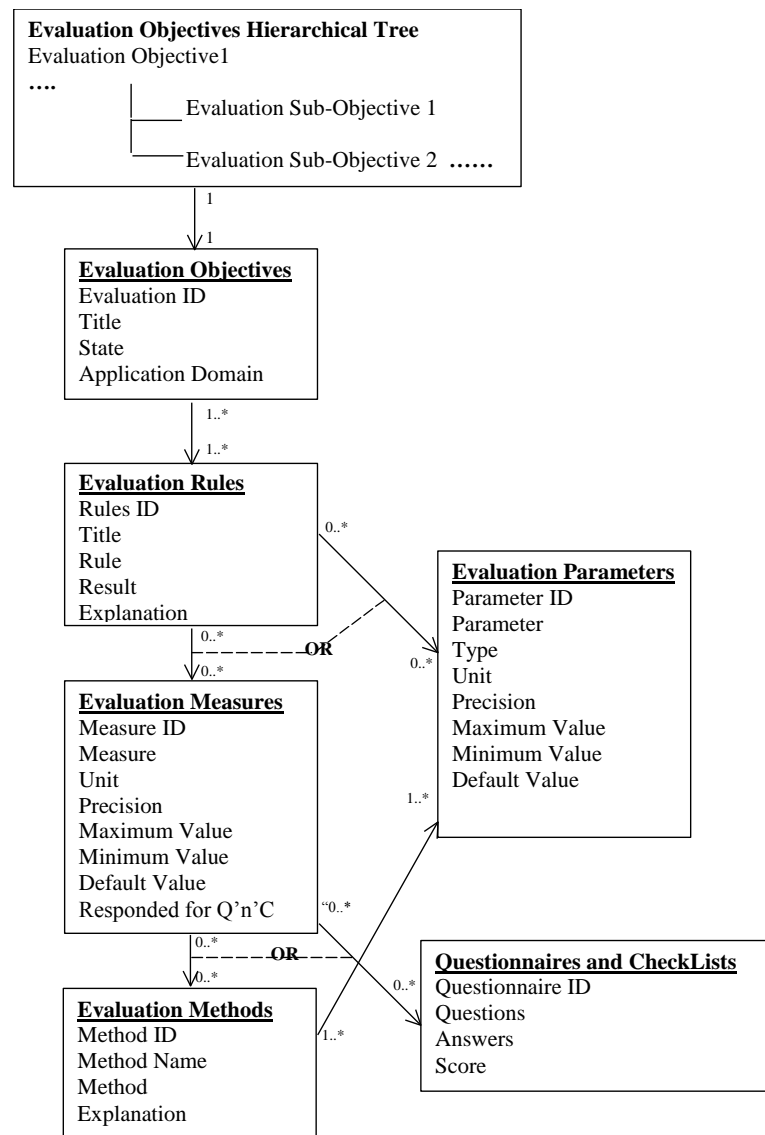


Figure 2.2 Common Evaluation Model (adopted from [65])

methods. Each evaluation method or rule has one or more related evaluation parameters and different evaluation methods (or rule) can use the same related evaluation parameter in order to prevent duplication of parameters. In simple evaluations, the evaluation knowledge will be defined by using evaluation objectives, rules and parameters. In complex evaluations, the evaluation knowledge will be defined by using evaluation objectives, rules, measures, methods and parameters.

As an instant of CEM, the relationship between evaluation objective 1 and related rules, measures, methods and parameters are shown in bold in Figure 2.3.

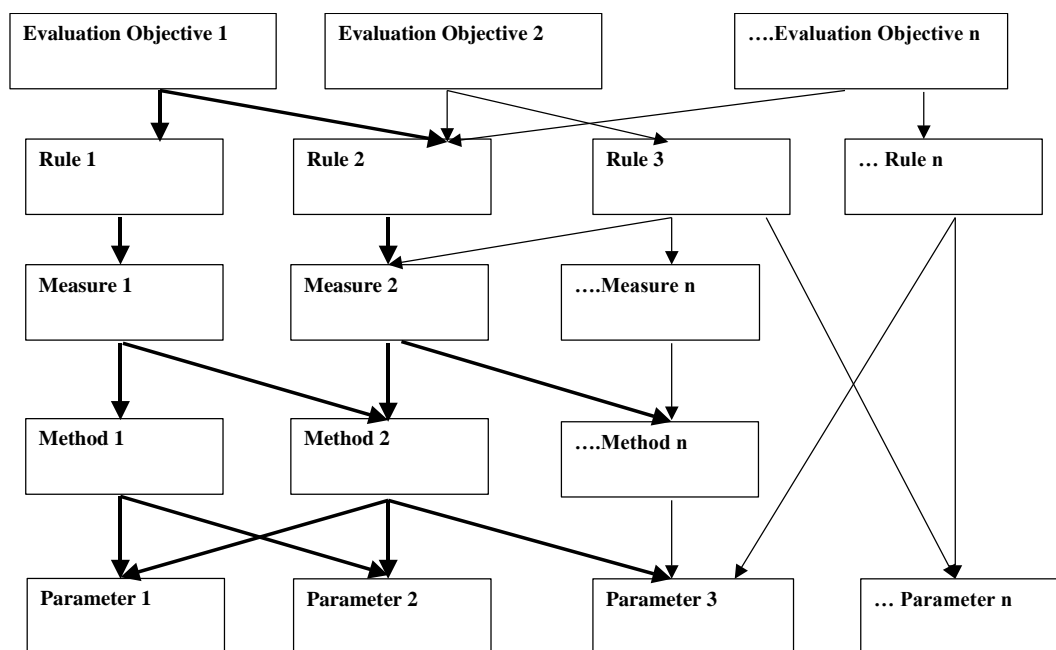


Figure 2.3 An instant of Common Evaluation Model (adopted from [67])

2.3 Mapping CEP to SEDEP and FEDEP

The mapping of CEP to SEDEP and FEDEP is shown in Table 2.1. The detailed information about the related processes can be found in the sections 2.1 (CEP), 1.3.5.4 (SEDEP), 1.3.5.3 (FEDEP). As seen from Table 2.1 SEDEP's and FEDEP's some steps and their some activities are related with evaluation and don't cover all aspects of evaluation, because SEDEP and FEDEP focus on Federation Development. Besides CEP focuses on only evaluation. Note that FEDEP's previous version didn't include evaluation aspects as the new one.

Table 2.1 Comparison of CEP with SEDEP and FEDEP

CEP	SEDEP (V1.0)	FEDEP (IEEE 1516)
Step 1 (Define Evaluation) Evaluation Objective Definition	Step 7-Selection/Identification of Evaluation Objectives for the federation resp.	Step 1: Define federation objectives. ➤ Sponsor needs ➤ Develop objectives
Step 1 (Define Evaluation) ➤ Rules Definition ➤ Measures or Parameters Definition ➤ Methods Definition ➤ Parameters Definition ➤ Questionnaires and Checklists Definition	Step 7- Transfer of the Evaluation Objectives into Evaluation Criteria	
Step 2 (Design Evaluation) ➤ Rules Design ➤ Measures or Parameters Design ➤ Methods Design ➤ Parameters Design ➤ Questionnaires and Checklists		Step 3: Design federation.
Step 3 (Execute Evaluation) • Execute methods with Parameters • Execute Rules using measures' value	Step 7 (Evaluate Execution Task): Apply appropriate statistical measures and other data reduction methods to transform output data into derived results. Evaluate Federation Task: The purpose of this activity is to evaluate and measure the capability of the federation to address the problem to be solved.	Step7: Analyze data and evaluate results. Apply analysis methods and tools to data.
Step 4 (Evaluation Results) Assessment	Step 7 Analyse Results: The purpose of this activity is to compile, and present the Evaluation Results coming from the previous activity.	Step 7: Analyze data and evaluate results. Apply analysis methods and tools to data.
Step 4 (Generate Evaluation Results) Results Presentation		Step 7: Analyze data and evaluate results. —Define appropriate presentation formats. — Prepare data in chosen formats.
Step 4 (Evaluation Results) Feedback		Step 7: Analyze data and evaluate results. Activity 7.2: Evaluate and feedback results
Step 4 (Evaluation Results) Evaluation History		

2.4 Requirements Of Evaluation Tools

Evaluation is generally time-consuming and difficult task. Evaluators and designers must have software tools or at least some toolkit to help them. Having tools available for evaluation (especially for performance evaluation) increases not only by saving time but also decreases the production error rate [71].

2.4.1 The features of evaluation tools

The high level requirements of Evaluation Tools are as follows [24],

- **The Evaluation Tool shall be flexible enough regarding the applied evaluation process (source code shouldn't be changed for different applications).**

All information dealing with the evaluation, which might be subject to change has easily to be put into changeable format (such database or local file system).

This has the advantage that the tool is easy to be maintainable and adaptable.

- **The Evaluation Tool shall be flexible enough regarding the applied evaluation rules (source code shouldn't be changed for different types of rules).**

All information dealing with the evaluation rules (criteria), which might be subject to change has easily to be put into changeable format (such database or local file system). By this way, the tool can be updated without for example re-compiling.

The evaluation rules, identified through for example interrogating SMEs, will be transformed into a machine-readable format. Since this rule base will be altered (i.e. probably extended) during the day-by-day use of the tool, it must be easy to modify the existing rules or add new ones.

- **The Evaluation Tool shall perform evaluation in an acceptable time.**

Note the acceptable time depends on applications. For real time application, the acceptable time is very short where for offline applications is longer.

- **The Evaluation Tool shall allow the user to perform evaluation in different domains, such as Synthetic Environments, trainees, instructors and real systems.**

- **The Evaluation Tool shall support the user in defining the data to be recorded for evaluation purposes.**

In general, there is some data collection activity associated with most systems, whereby data is collected to fulfill some predefined requirement for analysis. The analysis may be concerned with evaluating performance or behavior of the system users, or concerned with the performance or reliability of the system itself [25].

- **The Evaluation Tool shall be able to read (or to transform into a readable format) the data from the Data Logger.**
- **The Evaluation Tool shall search Knowledge base or database to find evaluation objectives relevant to the user needs.**
- **The Evaluation Tool shall allow the user to add new Knowledge and to modify existing ones (no code changes necessary).**
- **The Evaluation Tool shall derive low level Evaluation criteria (rules), measures, methods and parameters from high level Evaluation needs or Objectives.**
- **The Evaluation Tool shall allow the user to edit new Evaluation Objectives.**

This should either done by starting "from scratch" or by editing an already defined Evaluation Objective from the pool of Objectives.

- **The Evaluation Tool may be able to resolve inconsistencies in the Evaluation Objectives.**

Inconsistencies and contradictions in the selected Evaluation Objectives may be found explicitly (e.g. by respective routines) or implicitly by e.g. structuring the available objectives in a way, which prevents contradictions. Example could be to use hierarchical ordered tree structure.

- **The Evaluation Tool shall take mission effectiveness into account for evaluation.**

Mission effectiveness can be expressed by parameters like number of hits, number of losses etc. These parameters will be determined in cooperation with subject matter experts.

- **The Evaluation tool shall take mission efficiency into account for evaluation.**

Efficiency would be measured in values like "hits / fired ammunition" or "loses / total resources".

- **The Evaluation Tool shall be able to display and/or print evaluation results.**
- **The Evaluation Tool shall present the results in a suitable graphical manner.**
- **The Evaluation Tool shall allow comparison of results from different executions of evaluation for the same exercise.**
- **The Evaluation Tool shall be able to generate results, which are impartial and unbiased.**

It is a great asset of the automatic evaluation by the tool that the results generated this way are unbiased which tend to influence the assessment by humans.

It is for example known for a long time that human instructors tend to focus their evaluation results summary on the trainee's errors (and more or less take the trainee's correct actions for granted), which may lead to incorrect assumption about the trainee's overall performance.

- **The Evaluation Tool shall explain the reason of the results.**

The explanation of how the system reached a generated conclusion.

- **The Evaluation Tool shall save the results of the evaluation.**

Appropriate knowledge are stored and archived in a meaningful manner that supports the development of lessons learned and future exercises [72]. The results can be saved in databases or file systems. The saved evaluation results can provide comparison with previous evaluations, learning ratio, the necessity of repetition, etc.

- **The Evaluation Tool shall be able to generate/display and/or print questionnaires and checklists**

For some evaluation topics questionnaires are suggested, so the tool must be able to create them. Questionnaires can be used to gather information, which is not available by other means i.e. comments about the fidelity of the SE as a whole.

- **The Evaluation Tool shall be able to analyze data from questionnaires.**

The questionnaires provide information from instructors, trainees and other participants about the executed SE. This information (although subjective) is valuable for the Evaluation, especially for topics like "fitness-for-purpose". The Tool will take this information into account for the Evaluation process.

- **The Evaluation Tool shall support the use of COTS tools.**

The use of COTS tools within the Evaluation Tool can improve cost-effectiveness and speed up the development.

2.4.2 Applicable AI techniques respect to evaluation tools requirements

As seen from the previous section, some evaluation requirements are related with AI technology and some not. The comparison of applicable AI techniques and conventional computing with respect to some high level requirements of Evaluation Tools is shown in Table 2.2. The requirements given in the Table 2.2 are especially AI related.

In this study, ANNs and GAs were out of scope, because they don't have explanation capability. It is not easy to add new knowledge and to modify existing knowledge using ANNS and GAs.

The other techniques can also be partially beneficial for evaluation purposes, but out of scope of this thesis.

In this study, ANNs and GAs were out of scope, because they don't have explanation capability. It is not easy to add new knowledge and to modify existing knowledge using ANNS and GAs.

The other techniques can also be partially beneficial for evaluation purposes, but out of scope of this thesis.

Table 2.2 Comparison of applicable AI technologies and conventional computing with respect to some requirements of Evaluation Tools

Requirements of Evaluation Tools	AI Techniques					Conv. Comp.
	Expert Systems	Fuzzy Logic	Neural Network	Genetic Algorithm	Intelligent Agents	
Be flexible with regard to the applied evaluation process and rules (source code shouldn't be changed for different applications) [24]	Yes	Yes, but difficult	No	No	Yes	No
Perform evaluation in an acceptable time (Computational time)	Short	Short	Training is Long	Long	Short	Short
Allow user to evaluate from different domains Synthetic Environments, trainees, instructors and real systems	Yes	Yes	Yes, but needs training	Partially	Yes	Yes, but very difficult
Support user in defining the data to be recorded for evaluation purposes.	Yes	Partially	No	No	Yes	Yes
Search Knowledge base or database to find evaluation objectives relevant to the user needs	Yes	No	No	No	Yes	Yes
Allow user to add new Knowledge and to modify existing ones (without changing source code).	Yes	Yes	Partially but needs training	No	Yes	No
Derive low level Evaluation criteria (rules), measures, methods and parameters from high level Evaluation needs or Objectives.	Yes	No	No	No	Yes	Yes
Allow the user to edit new Evaluation Objectives.	Yes	Partially	No	No	Yes	
Resolve inconsistencies in the Evaluation Objectives	Yes	Partially	No	No	Yes	Yes
Take mission effectiveness and efficiency into account for evaluation.	Yes	Yes	Yes	No	Yes	
Generate results, which are impartial and free of bias.	Yes	Yes	Yes	Yes	Yes	
Explain the reason of the results	Yes	No	No	No	Yes	Generally No
Develop system in an acceptable time (Development time)	Medium	Medium	Short	Medium	Long	

3. INTELLIGENT EVALUATION SYSTEM (INES)

Intelligent Evaluation System (INES) was developed according to the “Common Evaluation Model” and the requirements of evaluation tools. The INES is a rule-based intelligence software tool including a special designed Expert System and Fuzzy Logic Toolbox to

- assist the user in the evaluation definition phase by providing information on which criteria, measures, methods, parameters and questionnaires need to be used in the evaluation.
- allow a direct access to captured knowledge of Subject Matter Experts. This increases confidence on the knowledge utilised in the evaluation process and improve the idea transfer and knowledge transfer among the evaluators
- execute evaluation definition and generate evaluation results
- present, and save results in textual and graphical form with the reason of inferencing
- reduce complexity associated with the evaluation
- reduce time and cost required to accomplish the evaluation tasks
- model the uncertainty about overall evaluation and provide reasoning on linguistic variables.

Human intelligence helps in identifying the right piece of knowledge at the appropriate instances of decision making. In AI studies, simulation of human intelligence on a system, so as to make the system efficient to identify and use the right piece of “Knowledge” at a given step of solving a problem is essential [54]. In this scope, the intelligency of INES for evaluation can be explained as follows:

- INES guides people (especially the people who are not SMEs) for evaluation process and shows them what they must look after and do.
- It performs evaluation similar to human SMEs.

- It can also evaluate other systems in case the Knowledge Base is filled with required knowledge.

INES was developed for Microsoft Windows platforms according to the requirements defined in previous chapter, the CEP and some other requirements related with GUI, architecture, etc. An iterative process was used to develop INES using Borland Delphi 5 and Matlab 6.5. With this process, firstly a small prototype was developed and enhanced by time. The evaluators and the other users should learn using the program in order to use INES. The software use of INES is given in Appendix A.

The INES can be used by evaluators such as instructors, etc. to evaluate trainees, instructors, job applicants, Synthetic Environments such as simulators, Computer Generated Forces (CGF) as well as real systems.

As indicated in Figure 3.1, The INES is mainly composed of three components. These are:

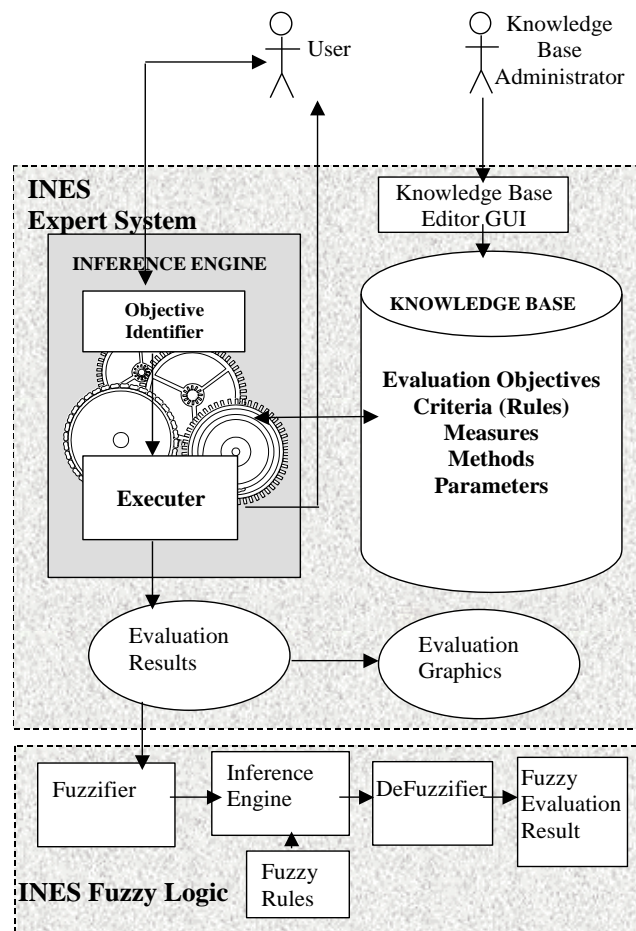


Figure 3.1 The components of INES

- Knowledge Base (KB) for storing the domain knowledge.
- ES Inference Engine for performing reasoning in accordance with the evaluation objectives defined by the user.
- Fuzzy Logic (FL) for doing overall assessment of results of executing ES in the highest level. FL was used to model the uncertainty about overall evaluation and provides reasoning on linguistic variables.

The ES Knowledge Base was separated from the ES Inference Engine. This is an important feature of ES, which makes design, and development of intelligent systems much easier. Besides, this separation allows the user to populate and improve the level of knowledge in the knowledge base easily when more knowledge is available over time.

3.1 INES ES Knowledge Base

The knowledge base is a data structure, which contains knowledge about the problem domain. INES KB (INES Knowledge Base) contains the knowledge and expertise of SMEs performing evaluation. The INES Knowledge Base contains knowledge about:

- Evaluation Objective Definition: The information about the title, state, definition of evaluation objectives and relationship of evaluation objectives each other.
- Evaluation rules: The knowledge about the criteria that is used for assessments such as successful/unsuccessful.
- Evaluation measures: The knowledge about the variables used in evaluation rules.
- Evaluation methods: The algorithms to calculate measures used in the rules.
- Evaluation parameters: The data about the variables used in measures or methods.

INES KB provides knowledge for the INES IE (INES Inference Engine) to make selections, and reasoning.

An example that can exist in the INES Knowledge Base is given below:

- Evaluation Objective is Pilot Destroy Success

- Evaluation Measure is Destroy_ratio
- Evaluation Method is $\text{Destroy_ratio} = (\# \text{ destroyed threats} / \text{total number of threats}) * 100$
- Evaluation Parameters = # destroyed threats, total number of threats
- Evaluation rules are
 - ❖ If $\text{Destroy_Ratio} < 40$ then $\text{Destroy_Success} = \text{poor}$
 - ❖ If $40 \leq \text{Destroy_Ratio} < 60$ then $\text{Destroy_Success} = \text{average}$

3.1.1 Knowledge elicitation and acquisition

Knowledge acquisition and elicitation is one of the main bottlenecks in AI studies. Since there is no formulation or algorithms for solving the problems requiring heavy expertise and skills, the knowledge is not as easy as to be collected and formulated. Note that it is not only important to formulate the knowledge, but finding out the right source of knowledge is also essential. This may also require a lot of effort from the developers to locate and understand the nature of knowledge required to solve problems requiring expertise and specific domain knowledge.

In this study, “Knowledge Base Editor” was developed in order to collect required knowledge and transform elicited knowledge into a machine-readable format.

For building and populating the INES Knowledge Base, basically, the following sources were identified:

- Literature survey (journal, conference papers, etc)
- Knowledge from SMEs of the respective fields,
- Military reports, policies and guidelines, instructions
- Search results on Internet web sites.

The approach used at the beginning of the acquisition process can be described as “top-down” approach involving the following steps [65]:

- 1) Determining application domain (i.e. instructor evaluation, pilot evaluation, collective training, mission rehearsal, SBA) and identify evaluation objectives. The objective describes what is going to be evaluated.

- 2) Seeking for rules (criteria) that are used e.g. to assess the synthetic environments, persons or systems.
- 3) Identifying detailed evaluation measures that are determined in evaluation rules.
- 4) Identifying detailed evaluation methods. Methods are formulas and algorithms that provide analysis results, which have to be assessed by using the rule.
- 5) Finding out detailed evaluation parameters that are used in the evaluation methods or rules and generally are collected after SE or exercise execution.

The required knowledge was elicited from SMEs and other sources to make INES to be able to make reasoning on previously selected evaluation objectives. It seemed to be easier for the SME to provide all information that they currently use to analyse an exercise at the first step. As a second step information that was really relevant for the evaluation was identified together with the algorithms and formulas used. Then also the definition of criteria to produce evaluation results was much easier for the experts because they had a set of measures (key factors) that they always use to judge the quality of an exercise or execution of a scenario. This resulted in a new or additional knowledge acquisition approach [65]:

- 1) Identify all data that are
 - provided by existing systems
 - required by experts to analyse an exercise, mission, experiment, person, system, etc
- 2) Reduce the huge amount of information to produce a set of key factors relevant for decision-making.
- 3) Identify a set of rules, which includes the different key factors to produce evaluation results.
- 4) Determine detailed evaluation measures that are used in evaluation rules.
- 5) Identify detailed evaluation methods.
- 6) Determine detailed evaluation parameters.

The set of key factors have to be included in several criteria which results provide meaningful information for the assessment of the exercise or experiment.

3.1.2 Knowledge representation and Knowledge Base architecture

Knowledge captured from experts and other sources must be organised in such a fashion that a computer inferencing program will be able to handle the captured knowledge [50]. Integrated “Reference Model of Evaluation Objectives” and “Evaluation Definition Knowledge” was used to represent evaluation knowledge in INES KB:

- Evaluation Objectives Hierarchical Tree (Reference Model of Evaluation Objectives) includes all evaluation objectives and their relationship including the dependencies. The hierarchical structure represents the relations between objectives, which end up in a hierarchy from high-level to low-level objectives. The hierarchical structure supports the user in finding suitable objectives and in defining a complete set of objectives. High-level objectives are the main branches of the tree whereas the low level objectives (sub objectives) are stored as lower level branches. Evaluation Objectives describe the (high-level) objectives of the evaluation to be performed. These can be derived from the User’s Needs and User / System Requirements Analysis.
- Evaluation Definition Knowledge, where evaluation objectives, rules, measures, methods, parameters, questionnaires, and their relationships are stored. Problem solvers like engineers mostly define this data. The work of the problem solver gets much easier because the evaluation definition data is linked with the objectives selected by the problem setter.

Figure 2.2 shows the detailed and complex information about evaluation objectives, rules, methods, measures, parameters and questionnaires, which are stored in the INES Knowledge Base.

The INES KB uses production rules as a knowledge representation method for defining evaluation rules (See Section 1.3.6.1 for advantages / disadvantages of rule representation). Evaluation rules are defined in a form of condition-action pairs: “IF a condition (or premise) occurs, THEN some action (or result, etc) will occur. A production rule is a statement having the following form [74]:

<production rule> ::= if <antecedent> then <consequent> *
 <antecedent> ::= <disjunction> {and <disjunction>}*

<disjunction>	::= <condition> {or <condition >}*
<consequent>	::= <conclusion> {also <conclusion>}*
<condition>	::= <predicate> (<variable>, <constant>)
<conclusion>	::= <action> (<variable>, <constant>)
<predicate>	::= same notsame greaterthan ...
<action>	::= display add greaterthan ...

A condition is built from a predicate and two associated arguments: a variable and a constant. By means of its predicate, a condition expresses a comparison between the specified constant value and the actual value(s) the specified variable has adopted. In the context of production systems, a predicate is a function which upon evaluation returns either the truth value true or the value false.

In this study the expression of condition is extended as follows

<condition>	::= <predicate> (<measure> or <parameter>, <constant>)
<measure>	::= <calculation> (<parameter>)

Evaluation parameters are the variables needed for applying rules or calculating the result of methods. The results of methods are defined as measures, which are used to simplify the evaluation rules and provide reusability. Evaluation methods are the algorithms for analyzing the collected parameters. Suitable questionnaires and checklists are also be used to collect some related evaluation parameters or measures

An example of a “production rule” used in INES is as follows,

IF (Instructor_Publications_Score <= 400) **AND** (Instructor_Publications_Score > 200) **THEN** Publications Success is Good.

Instructor_Publications_Score is defined as measure and the following method was used for calculating Instructor_Publications_Score,

Instructor_Publications_Score = (SCI_Papers * 30)+ (SCI_Books * 40) + (International_Conference_Papers*8) + (National_Conference_Papers*4) + (ReferredPapers *1) + (SCI_Other*10) + (Journals_Referee*15)

SCI_Papers, SCI_Books, International_Conference_Papers, ..etc was defined as parameters.

See section 3.4.2 for the details of this production rules and see section 3.4.1, 3.4.3 and 3.4.4 for other production rule examples.

Large numbers of rules can be a problem for expert systems [50, 75]. Using “Reference Model of Evaluation Objectives” and “Common Evaluation Model” was decreased the number of evaluation rules. If we represent Evaluation Objectives with rules, we must use a lot of rules to represent evaluation objectives and relationship of each of them with other. CEM simplifies the representation of evaluation knowledge.

The relationship between evaluation objectives, rules, measures, methods, and parameters are expressed in relation databases [65]. The relation of tables’ fields is shown with black arrows in Figure 3.2.

The detailed information about evaluation objectives is defined and stored in evaluation table. Evaluation Objectives table includes information about

- Objective ID: Unique identifier for each single objective
- Objective title: Meaningful name of the objective
- Description: Additional text that describes the objective more understandable
- State (mandatory, optional): Text that identifies whether an objective is mandatory or optional
- Keywords: Keywords that can be searched for by the user

Each evaluation objective is related with evaluation rule(s). There is a master-slave relation between evaluation objectives table and evaluation rules table as shown in Figure 2.2 and Figure 3.2 (evaluation objectives is the master).

The detailed information about evaluation rules is defined and stored in rules table. This table includes information about

- Related objective ID: Used to find rules that belong to a specific objective
- Rule ID: Unique identifier for each single rule
- Rule title: Meaningful name of the rule

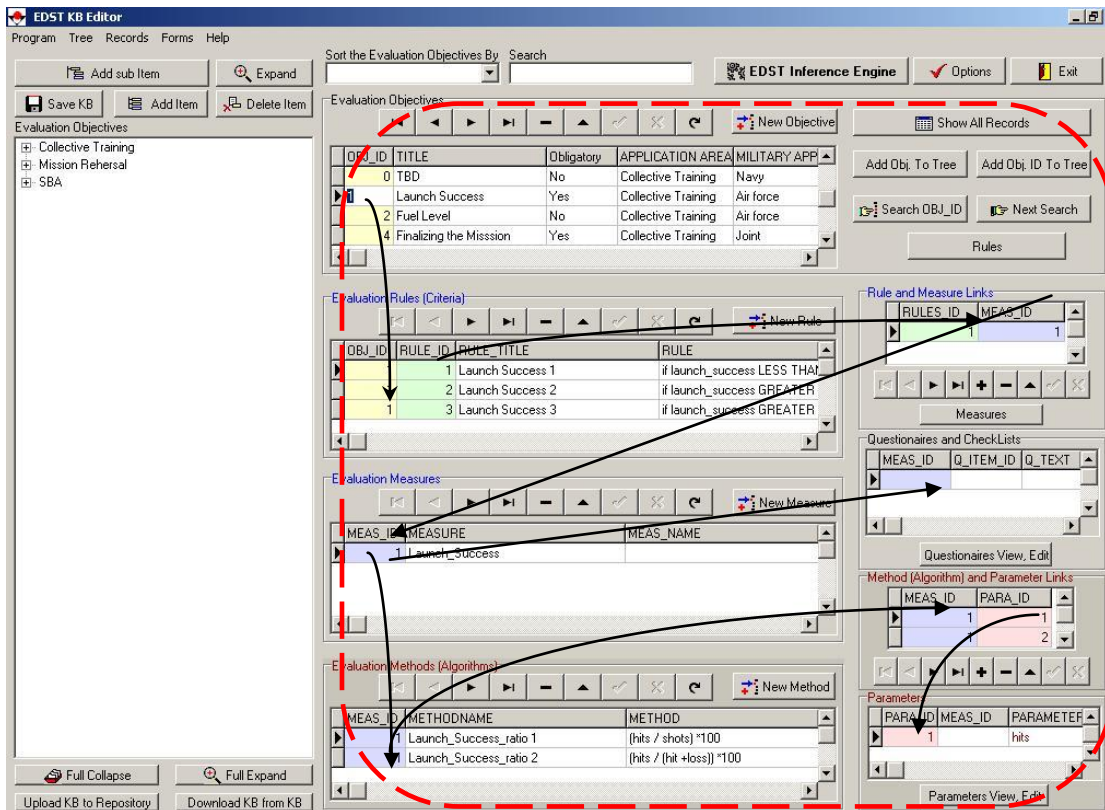


Figure 3.2 The related table fields in evaluation KB [64]

- Rule expression: Definition of the rule (IF part of the rule)
- Result of the rule: Result that should be used, if the output of the rule is true
- Explanation about rule: Additional text that describes the rule.

Each evaluation rule is related with evaluation measure(s). There is a master-slave relation between evaluation rules table and evaluation measures as shown in Figure 2.2 and Figure 3.2 (evaluation rules is the master).

Furthermore, the detailed information about evaluation measures is defined and stored in measures table. This table includes information about:

- Measure ID: Unique identifier for each single measure
- Measure title: Meaningful name of the measure
- Measure unit: Unit of the measure like seconds or meter
- Precision: Required precision of the measure
- Maximum value, Maximum value of the measure

- Minimum value, Minimum value of the measure
- Default value of measure Default value of the data
- The respondent of questionnaires. If a questionnaire is used for evaluation, this field describes the role of the respondent like pilot, instructor etc.

Note that there are also some Questionnaire and Checklists about the result of training, which is generally filled by trainees. The questionnaires are considered as a kind of measures in this study. The Questionnaires defined in Questions and Answers tables, which includes information about questions and explanations, related answers and scores.

Methods and questionnaires are related with measures as shown in Figure 2.2 and in Figure 3.2. Evaluation methods table includes information about:

- Measure ID: Unique identifier for each single measure which the method is related
- Method Name: Meaningful name of the method
- Method's expression: Definition of the method
- Explanation: Additional description and explanation
- Pre conditions: Required precondition for a method, e.g. to avoid dividing by zero

Parameters that are basic elements of the methods are also taken into account and they linked with the respective methods as shown in Figure 2.2 and in Figure 3.2. Parameters includes information about the variables of a method and parameters table includes information about:

- Parameter ID: Unique identifier for each single parameter
- Parameter expression: Parameter name
- Explanation: Additional description
- Type: Term that describes the type of parameter like constant, variable, simulator output etc.
- Unit: Unit of the measure like seconds or meter

- Precision: Required precision of the data
- Maximum value: Maximum value of the parameter
- Minimum value: Minimum value of the parameter
- Default value: Default value of the parameter

The INES Knowledge Base (KB) was designed to be managed, updated and maintained by a KB Administrator.

3.2 INES ES Inference Engine

The IE is essentially a computer program that handles the knowledge stored in the knowledge base and generates evaluation results for the user's evaluation objectives.

The core of the INES is its inference engine, which is also known as the control structure or the rule interpreter [50]. Finding a rule and executing it to generate knowledge or decisions is called rule firing. Various inferencing mechanisms are already developed. Three main strategies can be listed as below.

Backward chaining: In this strategy, the IE starts with a goal and tries to seek for the knowledge and domain facts satisfying the goal in question. If no knowledge is provided to satisfy the goal then the system asks the user to provide the required knowledge in order to make the decisions. The direction of inferencing is from the goal down to the related facts of the domain.

Forward chaining: In this strategy, the IE starts with the facts and available knowledge and try to define if there is a goal satisfied with the existing knowledge and facts on hand. If there is no goal satisfied with the knowledge then the user is informed about this. The direction of inferencing is from the facts of the domain to the goal.

Hybrid strategies: This strategy is the combination of both strategies explained above.

There are also various strategies developed for searching and rule firing. The details of these strategies can be found in [3, 76].

Backward chaining strategy was used for INES's IE inferencing. INES' Inference Engine working mechanism is as follows. User enters keyword(s) and Inference Engine of INES searches all the KB for possible evaluation objectives

using Depth-first search algorithm. The solution of evaluation objectives is presented in a tree-like structure with the successor and the predecessor of the solution. The database of INES is also searched in order to find the other solutions, which is not listed at the above solution. After getting users selections, INES Inference Engine performs the analysis of user evaluation objectives, finds necessary information for the evaluation execution such as criteria (rules), measures, methods, questionnaires, parameters and their relationships according to the selected evaluation objectives and put the collected data from the exercise to the related methods, measures, rules in order to calculate the results of evaluation.

3.2.1 Developing Inference Engine

INES ES Inference Engine was developed according to the activity diagram shown in Figure 3.3. Each activity represents the performance of a group of “actions” in a workflow. The brief explanation of INES IE activities is as follows:

Read evaluation keywords: This activity receives evaluation keywords from the user in order to present the user the possible evaluation objectives from Evaluation Knowledge Base (KB).

Search evaluation objectives tree: This activity searches evaluation keywords in the evaluation Objectives tree.

Search evaluation database for keywords: This activity searches evaluation keywords in the evaluation database.

Generate evaluation objectives results: This activity generates and presents the results of search in a hierarchical form.

Edit Knowledge Base: This activity allows user to update, modify and add knowledge into Knowledge Base when the user cannot find his evaluation objectives in the results of search.

Select evaluation objectives among results: This activity receives user’s evaluation objectives selections from the user.

Find evaluation rules related with the selected evaluation objectives: This activity finds evaluation rules related with the selected evaluation objectives from evaluation KB.

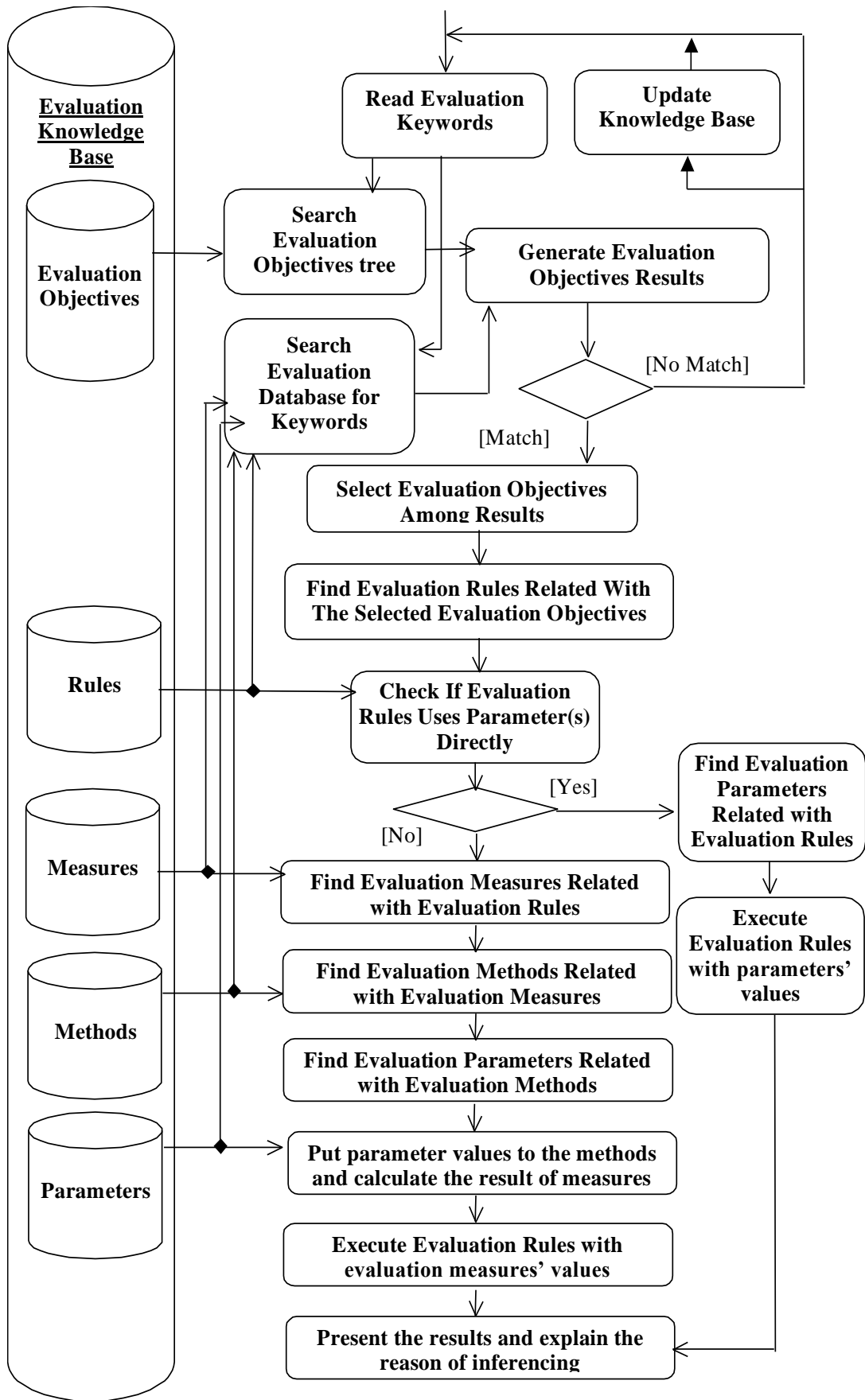


Figure 3.3 Activity diagram of INES Inference Engine

Check if evaluation rules use parameter(s) directly: This activity checks if the evaluation rules in the KB is defined by evaluation measures or evaluation parameters.

Find evaluation parameters related with evaluation rules: If the evaluation rules in the KB are defined by evaluation parameters, this activity will find Evaluation Parameters related with evaluation rules.

Find evaluation measures related with evaluation rules: If the evaluation rules in the KB are defined by evaluation measures, this activity will find evaluation measures related with evaluation rules.

Find evaluation methods related with evaluation measures: If the evaluation rules in the KB are defined by evaluation measures, this activity will find evaluation methods related with evaluation rules.

Find Evaluation Parameters Related with Evaluation Methods: If the evaluation rules in the KB are defined by evaluation measures, this activity will find Evaluation Parameters related with evaluation rules.

Put parameter values to the methods and calculate the result of measures: This activity gets parameter values from external or internal (e.g. from logger system) and puts these values to the related methods in order to calculate the result of measures.

Execute evaluation rules with the values of evaluation measures: This activity executes evaluation rules with the calculated the values of the evaluation measures.

Execute evaluation rules with the values of parameters: If the evaluation rules in the KB are defined by evaluation parameters, this activity will put the values of the parameter to the rules and calculate the result of rules.

Present the results and explain the reason of inferencing: This activity presents the results of the rule execution and explains the reason of inferencing.

3.2.2 Components of INES Inference Engine

The inference engine supports user to select his evaluation objectives and generates evaluation results with explanation of the reasoning.

As shown in Figure 3.1, there are two main components of the INES IE including, objective identifier and evaluation executer.

3.2.2.1 Objective Identifier (OI)

This component gives the opportunity to the user to identify his evaluation objectives from predefined set of objectives. First, the user enters evaluation Keywords. INES searches the KB for these keywords and generates a set of evaluation objectives complying with the keywords from which the user may make the selection. Depth First Search Algorithm was implemented to search keywords in evaluation objectives tree. Depth-first search always expands one of the nodes at the deepest level of the tree. Only when the search hits a dead end (a non-goal node with no expansion) does the search go back and expand nodes at shallower levels [3]. Note that INES IE continues searching until finishing to search the entire decision tree as stored in the knowledge base. Each result node and its successors are generated in the above “Found Evaluation Objectives” treeview as shown in Figure 3.4 (dashed rectangle-top treeview). The user can obtain detailed information about the objectives stored in the KB. The details provided are explained below.

INES IE performs two types of search. First, the inference engine search the evaluation objectives stored in the INES KB using keywords as provided by the user. If respective objectives are found they are listed for the user in the dashed rectangle-top treeview of “Found evaluation objectives”. The objectives in this treeview are definitely related to the user request. However, there could be some objectives which

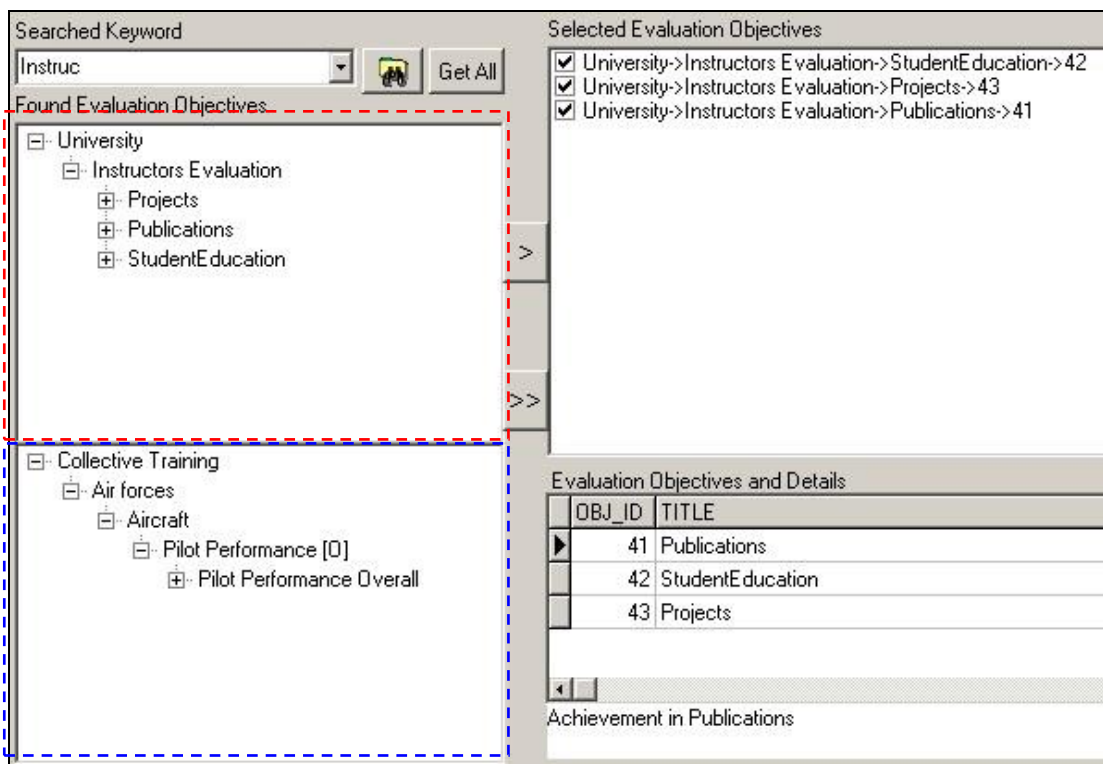


Figure 3.4 An example of INES IE evaluation objectives

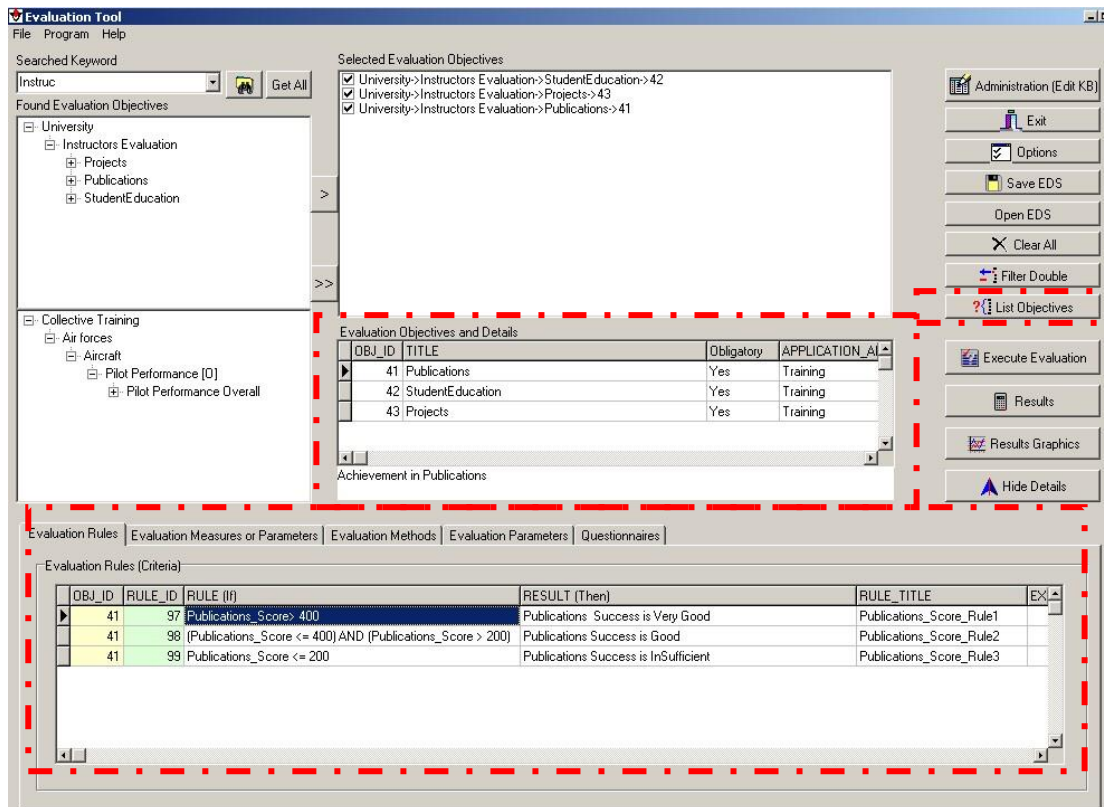


Figure 3.5 An example of INES evaluation selection

are related to the user goal but do not contain the keywords in the objective title as defined in the objective tree. In this case, second search took place to search the whole database for the specified keywords. If the searched keyword is found in the fields connected to the related evaluation objectives, then the objective is listed for the user to check if it suits for his goal.

In other words, database tables of “Evaluation KB” are searched by using SQL to find relevant evaluation objectives. When relevant data (based on provided keywords) are found in database tables (rather than the title in the objective tree), they are identified and checked if they are already found in the first type of search using “Search Depth Algorithm”. If yes, then the respective objective is ignored (as already being in the selected list) to prevent duplication. If no, then hierarchical solution of related evaluation objective is generated in the bottom treeview of “Found Evaluation Objectives” as shown in Figure 3.4 (dashed rectangle-bottom treeview). Detailed information about the evaluation objectives is also shown to the user. The information provided includes;

- Title of the objectives
- Application area (Collective Training, Mission Rehearsal or SBA)

- Description
- Obligatory (Yes or No)
- Keywords

This component also supports user to select suitable evaluation objectives and generates the list of selected evaluation objectives and related evaluation criteria, measures, methods, parameters and questionnaires as shown in Figure 3.5 in squares with dashed lines.

3.2.2.2 Evaluation Executer (EE)

This component first finds evaluation rules related with the selected evaluation objectives from Evaluation KB and checks if the evaluation rules in the KB are defined by evaluation measures or evaluation parameters. If the evaluation rules in the KB are defined by evaluation parameters, “Evaluation Executer” finds Evaluation Parameters related with evaluation rules. If the evaluation rules in the KB are defined by evaluation measures, EE finds Evaluation measures, methods and parameters related with evaluation rules.

After that, EE gets parameter values from external (i.e. from a local file) or internal (default values) and puts these values to the related methods in order to calculate the result of measures. If the evaluation rules in the KB are defined by evaluation parameters, this component will put parameter values to the rules and calculate the result of rules. If the evaluation rules in the KB are defined by evaluation parameters, this component executes evaluation rules with the calculated values of the evaluation measures. After all these activities, EE presents the results of the rules and provides explanation about the inferencing.

INES Inference Engine is capable of generating the following output files, which provides other software to export the evaluation knowledge:

- EvaluationObjectives.xml (Evaluation Objectives)
- EvaluationDefinition.xml (Evaluation rules, methods, measures, parameters)
- EvaluationResults.xml (current exercise results)
- EvaluationDefinition.eds (current work save)

XML is a general-purpose, meta-markup language for documents containing structured information that supports the definition of customized markup components [77]. XML is widely used and is standardized, as a result the language supports re-use of the knowledge bases. As mentioned in Section 2.2, hierarchical representation is one of the oldest methodology to represent complex data. The concept of XML and its widespread use is because of the easiest way it could represent such data. The importance of hierarchical data representation is so much that we could represent any present day system models based on it. Metadata representation in the hierarchy could tell a lot about the data itself [70]. In our case, the document is the evaluation KB. The generated XML files can be used by other computer programs to elicit evaluation knowledge for a specific exercise.

3.3 INES Fuzzy Logic Module

INES Fuzzy Logic Module consists of three main components as shown in Figure 3.1:

- Fuzzifier maps the output of ES evaluation results to fuzzy sets.
- **Fuzzy Inference engine** generates fuzzy outputs corresponding to fuzzified inputs, with respect to the fuzzy rules. Fuzzy Rules are in “IF...THEN...” form and combines inputs to outputs. Fuzzy rules for two different evaluation examples are shown in Table 3.1 and Table 3.2.
- **Defuzzifier** maps output fuzzy sets into fuzzy evaluation results.

3.4 INES Implementation On Various Cases

INES was used for evaluation purposes at different domains. In this section, evaluation of Air Defence (AD) System, instructor performance, pilot performance and personnel selection is explained in detailed.

3.4.1 AD (Air Defence) system evaluation

Air Defense System is used to protect some region from all air threats especially from guided munitions such as missiles as shown in Figure 3.6.

System performance measurement and evaluation process must be viewed as an essential supportive component of effective information system functioning and improvements of the system [68].

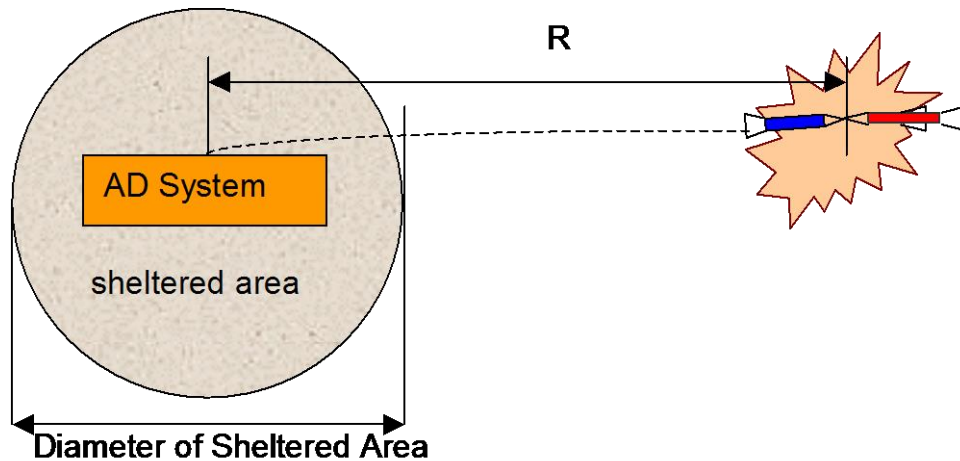


Figure 3.6 Air Defence System (adopted from [78])

The performance of the AD system is determined by the interception capability. AD system's interception capability of the threat, which is generally a cruise missile was investigated in this application.

3.4.1.1 INES implementation

Following evaluation objectives were identified for evaluation of AD System:

1. Evaluate the hit ratio of the AD system
2. Evaluate the AD system reaction time
3. Evaluate the damage level in the sheltered area
4. Overall Performance

The performance of AD System is evaluated by applying several rules that use various measures. The measures for AD System evaluation are Reaction_Time, Hit_Ratio, Destroyed_Sheltered_Area_Ratio, and Overall_Performance.

3.4.1.2 Evaluation of the hit ratio of the AD system

The following rules can be used as criteria to evaluate the "Hit ratio":

- If Hit ratio is greater equal 85% then Hit ratio is sufficient
- If Hit ratio is less than 85% then Hit ratio is insufficient

A 100% hit ratio is not essential because the AD System usually should be able to launch a second AD missile, if necessary.

The method for calculating the Hit_Ratio is

$$\text{Hit_Ratio} = (\text{SUM of missiles Hit} / \text{SUM of missiles launched}) * 100 \quad (3.1)$$

3.4.1.3 Evaluation of the AD system reaction time

An important characteristic of the AD System is the reaction time. The reaction time is the total time needed from detection of an object by the AD radar to the launching of the AD missile.

The method used for calculating the Reaction time is as follows:

$$\text{reaction time [ms]} = \text{detect_ident_time [ms]} + \text{ident_class_time [ms]} \quad (3.2a)$$

$$+ \text{class_alloc_time [ms]} + \text{alloc_firing_time [ms]} \quad (3.3b)$$

Where

- detect_ident_time: time between detection and identification of the object
- ident_class_time: time between identification and classification of the object
- class_alloc_time: time between classification and allocation to a suitable AD weapon system
- alloc_firing_time: time between allocation and launching the AD missile

Typical reaction times of AD Systems range between 5 and 6.5 seconds. Realistic rules could be:

- If the reaction time is less than or equal 5000ms then the reaction time is very good.
- If the reaction time is greater than 5000ms and less than 6500ms then the reaction time is good.
- If the reaction time is greater than or equal 6500ms then the reaction time is insufficient.

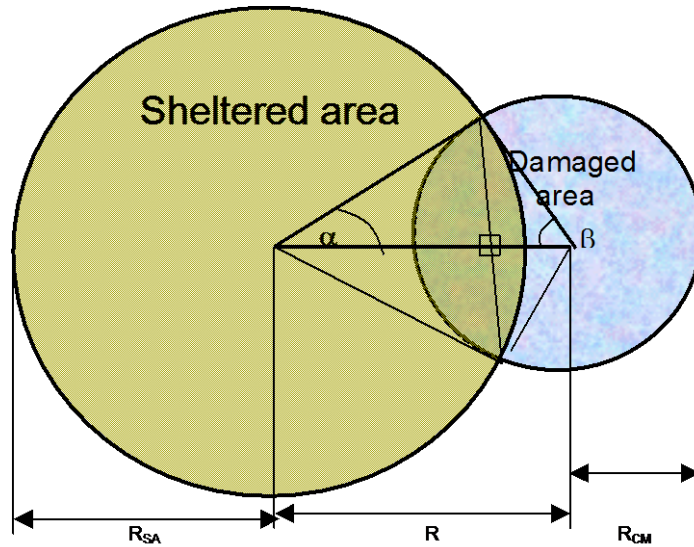


Figure 3.7 Two dimensional damage model of sheltered area and cruise missile (adopted from [78])

3.4.1.4 Evaluation of the damage level in the sheltered area

Damage caused by the cruise missile as a function of the distance at which it explodes from the sheltered area is evaluated.

Two Dimensional Damage Model of Sheltered Area and Cruise Missile is shown in Figure 3.7. There, R is the distance between the center of sheltered area and Destruction Center of Cruise Missile, R_{SA} is the radius of Sheltered Area, R_{CM} is Destruction Radius of Cruise Missile, α and β are the angle to calculate circle segment of the Sheltered Area and the Cruise Missile.

The following rules were used to evaluate the destroyed_sheltered_area

- If R is greater equal than $(R_{SA}+R_{CM})$ then Defenced Area is undamaged
- If R is smaller than R_{SA} then Defenced Area is damaged
- Else If Destroyed_Area_Ratio is less than 10% then Damage on Defenced Area is Acceptable
- Else If Destroyed_Area_Ratio is greater equal 10% then Damage on Defenced Area is Unacceptable

For calculation of Destroyed_Area_Ratio,

$$\text{DestroyedArea} = 2 * (\text{circle segment area with } \alpha - \text{triangle area with } \beta) \quad (3.4a)$$

$$+ 2 * (\text{circle segment area with } \beta - \text{triangle area with } \alpha) \quad (3.5b)$$

$$= 2((\pi R_{SA}^2 \alpha / 2\pi) - (R_{SA}^2 \sin 2\alpha / 4)) + 2((\pi R_{CM}^2 \beta / 2\pi) - (R_{CM}^2 \sin 2\alpha / 4)) \quad (3.6)$$

$$= \pi R_{SA}^2 \alpha - (R_{SA}^2 \sin 2\alpha / 2) + R_{CM}^2 \beta - (R_{CM}^2 \sin 2\alpha / 2) \quad (3.7)$$

$$\text{DestroyedArea Ratio (\%)} = 100 * (\text{DestroyedArea} / \text{Sheltered Area}) \quad (3.8)$$

$$= 100 * (\pi R_{SA}^2 \alpha - (R_{SA}^2 \sin 2\alpha / 2) + R_{CM}^2 \beta - (R_{CM}^2 \sin 2\alpha / 2)) / \pi R_{SA}^2 \quad (3.9)$$

$$= 100 * (\alpha - (\sin 2\alpha / 2) + (R_{CM} / R_{SA})^2 \beta - ((R_{CM} / R_{SA})^2 \sin 2\alpha / 2)) / \pi \quad (3.10)$$

$$= 100 * (\alpha - (\sin 2\alpha / 2) + (R_{CM} / R_{SA})^2 (\beta - \sin 2\alpha / 2)) / \pi \quad (3.11)$$

Where

$$R_{CM}^2 = R_{SA}^2 + R^2 - 2R_{SA}R \cos(\alpha) \quad \text{Law of cosines} \quad (3.12)$$

$$\alpha = \arccos((R_{SA}^2 + R^2 - R_{CM}^2) / 2R_{SA}R) \quad (3.13)$$

$$R_{SA}^2 = R_{CM}^2 + R^2 - 2R_{CM}R \cos(\alpha) \quad \text{Law of cosines} \quad (3.14)$$

$$\beta = \arccos((R_{CM}^2 + R^2 - R_{SA}^2) / 2R_{CM}R) \quad (3.15)$$

Overall Performance is calculated as a function of Reaction_Time, Hit_Ratio, Destroyed_Sheltered_Area_Ratio.

Note, it was assumed that R_{CM} and R (cruise missile destroy range) is equal each other in this model. We can also take the height of the cruise missile explosion into account. Then

$$R_{DR}^2 = R_{CM}^2 + h^2 \rightarrow R_{CM} = \sqrt{(R_{DR}^2 - h^2)} \quad (\text{see Figure 3.8}) \quad (3.16)$$

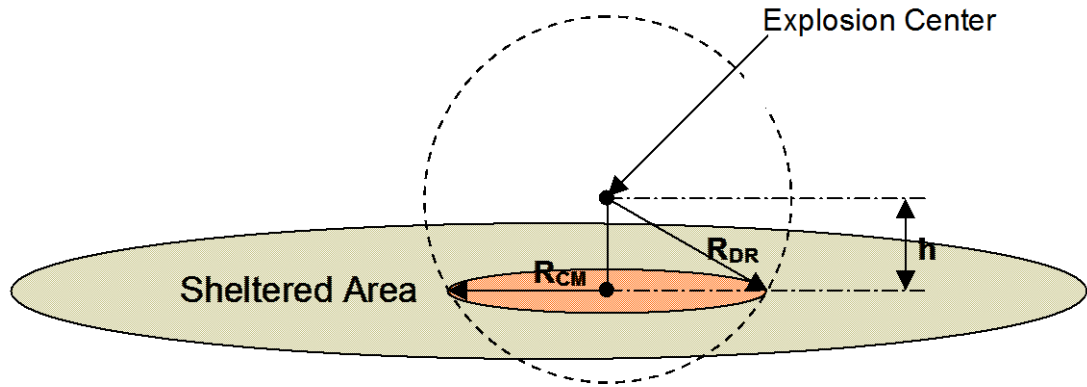


Figure 3.8 The effective destruction in the sheltered area (adopted from [78])

Where

R_{DR} : cruise missile destroy range

h : altitude of detonation

R_{CM} : effective radius of destruction

An example screenshot of INES for AD System evaluation result and the reason for “Destroyed Sheltered Area” evaluation objective is shown in Figure 3.9.

INES is capable of comparing the evaluation results graphically. For example, comparison of reaction times for different AD Systems is shown in Figure 3.10.

OBJ_ID	OBJ_TITLE	RESULT	RESULT_VAL
44	ADSystemReactionTime	VERY GOOD	4880
45	HitRatio	Sufficient	100
46	DestroyedShelteredArea	Damage on Defenced Area is Unacceptable	13.22
51	ADSystemOverallPerformance	ADSystem Overall Performance is Good	83.13

Reason

Used Rules:
Sheltered_Area_Ratio >=10 " Damage on Defenced Area is Unacceptable "

Used Methods:
 $Sheltered_Area_Ratio_Met = \arccos\left(\frac{R_{run}^2 + R^2 - R_{cm}^2}{2 \cdot R_{run} \cdot R}\right)$;
 $\beta = \arccos\left(\frac{R_{cm}^2 + R^2 - R_{run}^2}{2 \cdot R_{cm} \cdot R}\right)$;
 $Sheltered_Area_Ratio = \frac{100}{3.14} \cdot \left(\arccos\left(\frac{R_{run}^2 + R^2 - R_{cm}^2}{2 \cdot R_{run} \cdot R}\right) \cdot \sin^2\left(\frac{\alpha}{2}\right) + \arccos\left(\frac{R_{cm}^2 + R^2 - R_{run}^2}{2 \cdot R_{cm} \cdot R}\right) \cdot \sin^2\left(\frac{\beta}{2}\right) \right) = 13.219893914618$;

Used Parameters:
 $R=500$ " Range "
 $R_{run}=400$ " UN camp radius "
 $R_{cm}=300$ " Cruise Missile Destruction Radius "

Figure 3.9 Screenshot of INES ES for AD System evaluation

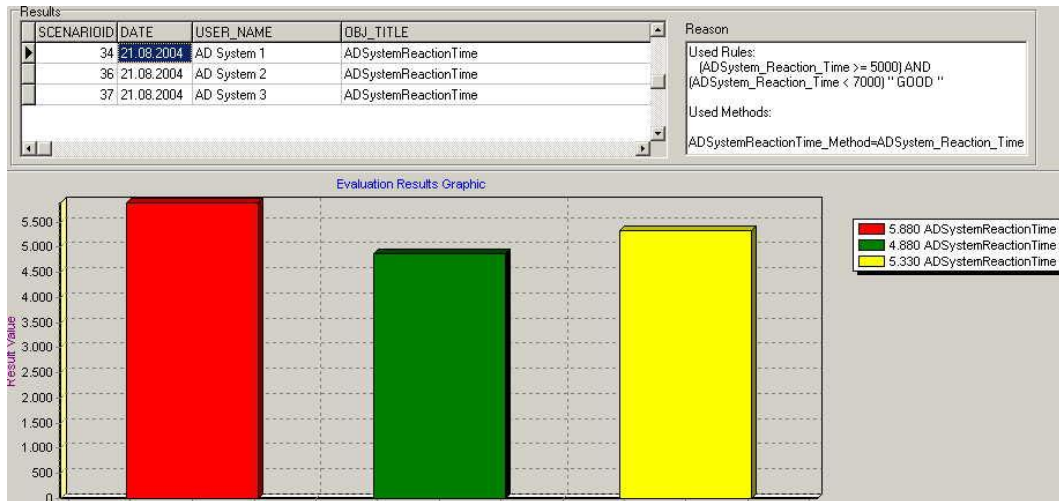


Figure 3.10 Screenshot of INES for AD Systems comparison

3.4.1.5 AD System overall evaluation

The overall evaluation of AD System was done in two ways:

1. Using INES Expert System
2. Using INES Fuzzy Logic

Using INES Expert System: The overall evaluation was calculated as a function of “reaction time”, “hit ratio” and “damaged ratio of the sheltered area”. Screenshot of INES ES for AD System evaluation result and the reason of “Overall Performance” is shown in Figure 3.11.

Using INES Fuzzy Logic: The results of ES are assessed by Fuzzy Logic (FL) Module by parameter passing from INES ES to INES FL.

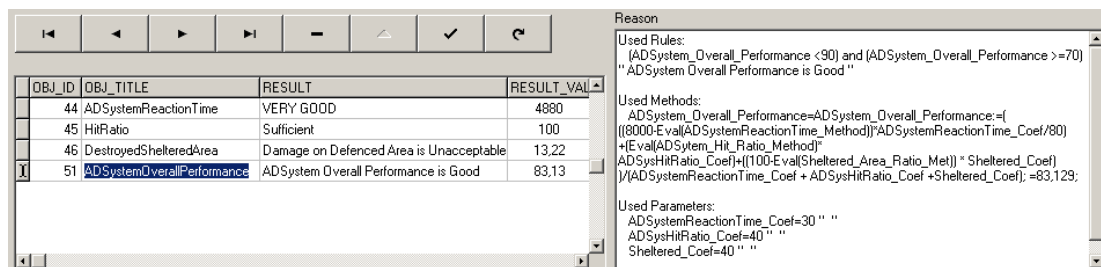


Figure 3.11 Screenshot of INES ES for AD System evaluation

Table 3.1 The rules of AD System Fuzzy Evaluation System

Reaction Time	VGood	Average	Bad
Destroyed Area			
Little	<u>vgood</u>	vgood	good
Average	lgood	bad	bad
Large	bad	vbad	vbad

If Hit Ratio is bad

Reaction Time	VGood	Average	Bad
Destroyed Area			
Little	excellent	vgood	vgood
Average	good	good	good
Large	lgood	bad	bad

If Hit Ratio is good

The rules of AD System Fuzzy Evaluation System are shown in Table 3.1 with “Hit Ratio”, “Reaction Time” and “Destroyed Area” as inputs and “Overall Performance” as output. For example, the meaning of the table for the underlined cell is

If (Hit Ratio is bad) AND (Destroyed Area is little) AND (Reaction Time is Very Good), then (Overall Performance is very good).

The meaning of table for other cells is defined in similar way.

For defining membership functions, several methods such as common sense, neural network, genetic algorithmics can be used [79]. In this thesis, membership functions are defined according to the nature of the problem, expertise knowledge about domain and using common sense. Several types (Gaussian, triangle and trapezoid) of membership functions were tried and compared with ES results. There is not much difference between the overall performance results generated by INES ES and Fuzzy Logic for Gaussian, triangle and trapezoid membership functions for different values of “Reaction Time”, “Hit Ratio” and “Destroyed Sheltered Area” as shown in Figure 3.12.

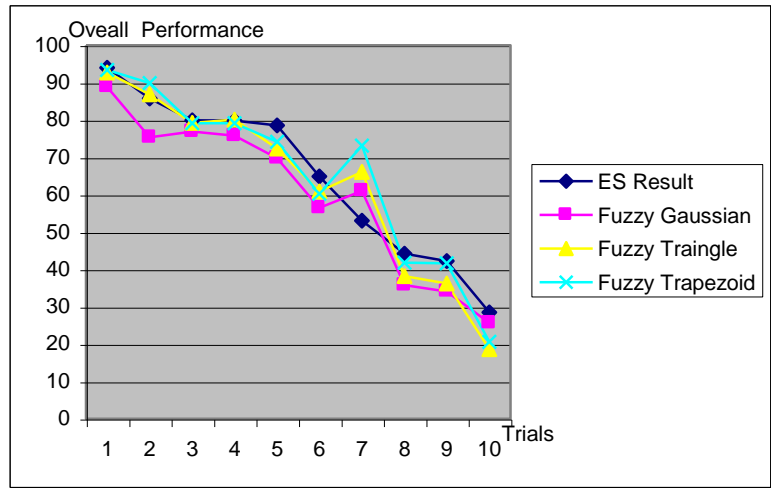


Figure 3.12 Overall results of INES ES and Fuzzy Logic Module for different membership functions

The “Gaussian” membership functions of “Hit Ratio”, “Reaction Time” and “Destroyed Area” as inputs and Overall Performance as output is shown in Figure 3.13, Figure 3.14, Figure 3.15 and Figure 3.16.

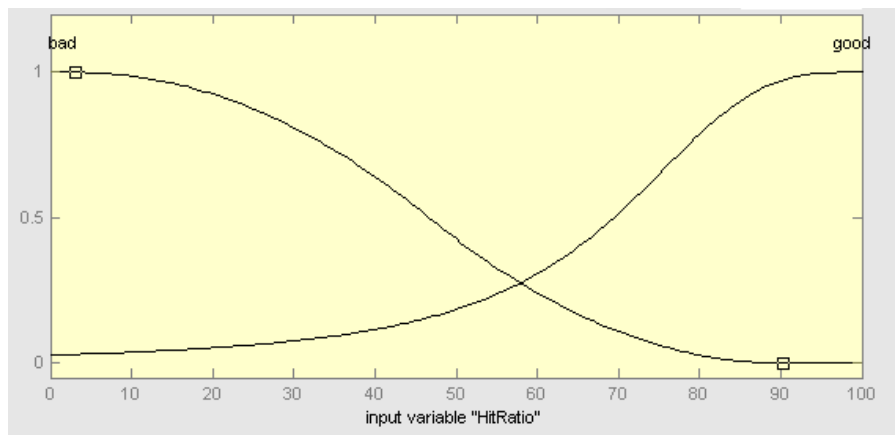


Figure 3.13 Membership function for input variable “HitRatio”

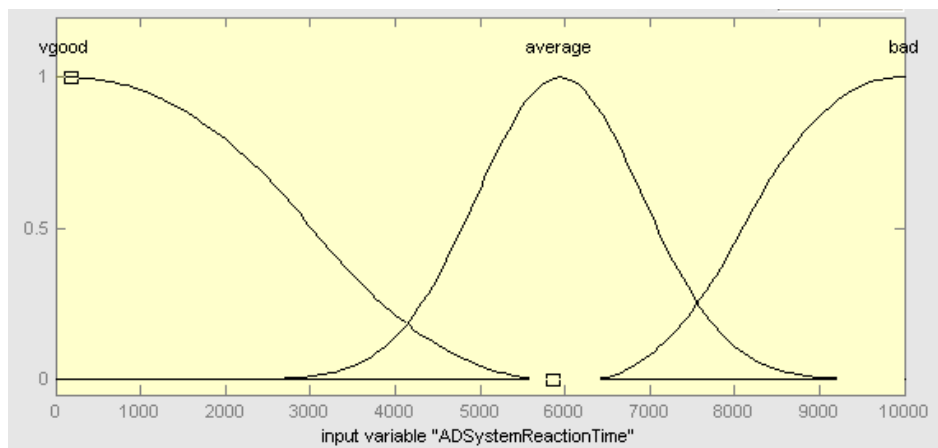


Figure 3.14 Membership function for input variable “Reaction Time”

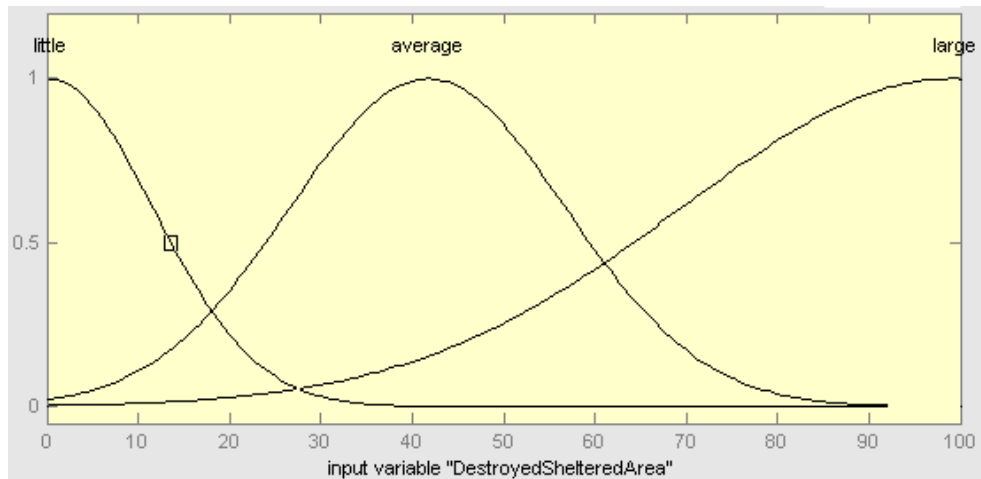


Figure 3.15 Membership function for input variable “Destroyed Area”

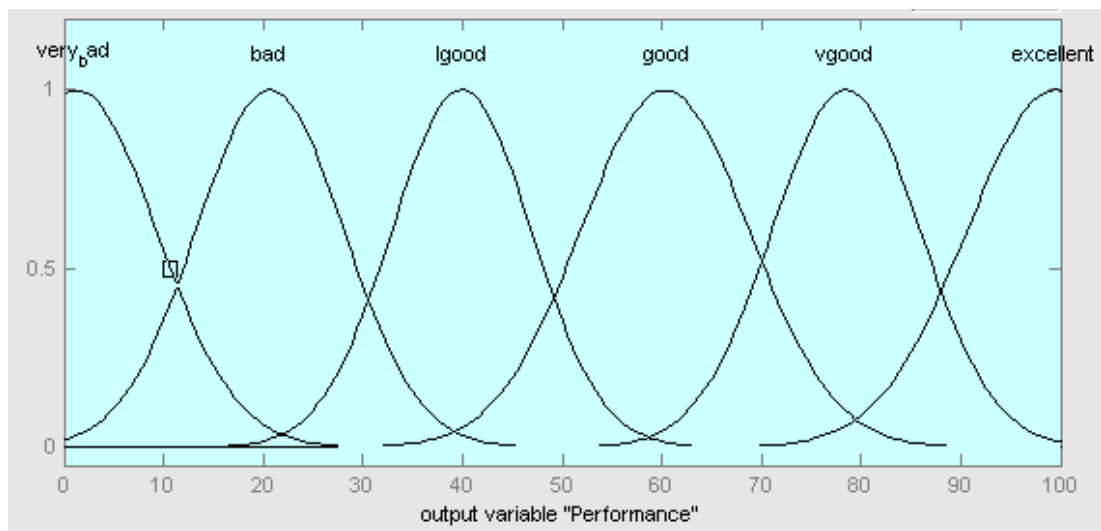


Figure 3.16 AD System Fuzzy Evaluation output variable “Performance”

The overall evaluation result of AD System using fuzzy logic is shown in Figure 3.17. All parts of the fuzzy inference process are simultaneously displayed. There are 18 rows and each row shows the fuzzy inference process of a rule. For example, Row 11 shows the following rule;

If (Reaction Time is average) AND (Hit Ratio is good) AND (Destroyed Area is little), then (Overall Performance is very good).

The right bottom rectangle shows the output for this instance, which is the combination of output of rules. The centroid calculation, which one of the most popular defuzzification method was used for generating the crisp output. This method returns the center of area under the output curve.

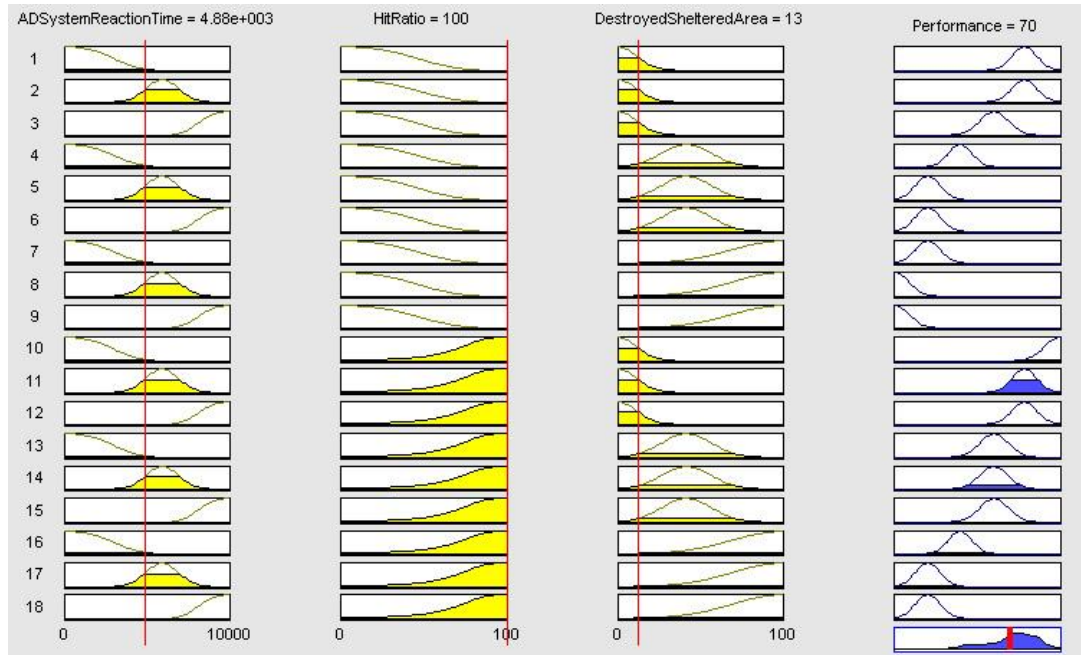


Figure 3.17 Overall result of AD System evaluation

3.4.2 Instructor evaluation

In this example, following evaluation sub-objectives were identified for university instructor (academic staff) performance:

- 1) The Education of Students: The success of instructor in teaching
- 2) Publications: The Achievement of instructor in Publications
- 3) Projects: The success of instructor in researches

Note that, different evaluation experts may define the above objectives and the details for evaluating these objectives in different formats. Evaluation objectives and evaluation definition are subject to change according to the intention of SMEs and as well as relevant expertise.

The performance of instructor was evaluated by applying several rules that use various measures. The defined measures are

- Publications_Score
- Student_Education_Score
- Projects_Score

The following rules were used as criteria to evaluate the publication performance:

- If Publications_Score > 400 then Publications Success is Very Good
- If (Publications_Score <= 400) AND (Publications_Score > 200) then Publications Success is Good
- If Publications_Score <= 200 then Publications Success is InSufficient

The method defined for calculating the Publication Success is as follows:

$$\begin{aligned} \text{Publications_Score} = & (\text{SCI_Papers} * 30) + (\text{SCI_Books} * 40) + (\text{SCI_Other} * 10) + \\ & (\text{Journals_Referee} * 15) + (\text{International_Conference_Papers} * 8) \\ & + (\text{National_Conference_Papers} * 4) + (\text{ReferredPapers} * 1) \end{aligned}$$

Where

- *SCI_Papers* are number of papers published in SCI (Science Citation Index), SSCI (Social Science Citation Index) or AHCI (Arts&Humanities Citation Index)
- *SCI_Books* are number of books published in SCI (Science Citation Index), SSCI (Social Science Citation Index) or AHCI (Arts&Humanities Citation Index)
- *SCI_Other* are number of other studies published in SCI (Science Citation Index), SSCI (Social Science Citation Index) or AHCI (Arts&Humanities Citation Index)
- *Journals_Referee* are number of papers published in Journals with Referee Review
- *International_Conference_Papers* are number of papers published in International Conferences
- *National_Conference_Papers* are number of papers published in National Conferences
- *ReferredPapers* are number of papers referred to the publications

The following rules was used as criteria to evaluate the success in teaching:

- If Student_Training_Score > 200 then Student Training Performance is Very Good
- If (Student_Training_Score <= 100) then Student Training Performance is insufficient
- If (Student_Training_Score <= 200) AND (Student_Training_Score > 100) then Student Training Performance is Good

The method used for calculating the Student Training Success is as follows:

$$\text{Student_Training_Score} = (\text{PhDAdviser} * 15) + (\text{MScAdviser} * 10) + (\text{BScAdviser} * 5) + (\text{CompletedPhD} * 30) + (\text{CompletedMSc} * 20) + (\text{GivenLessons} * 10)$$

Where

- PhDAdviser : Number of PhD Students
- MscAdvisor : Number of MSc Students
- BscAdvisor : Number of BSc Students
- CompletedPhD : Number of Completed PhD Thesis
- CompletedMSc : Number of Completed MSc Thesis
- GivenLessons : Number of Given Lessons

An example screenshot of INES for instructor evaluation result and the reason for “Publications” evaluation objective is shown in Figure 3.18.

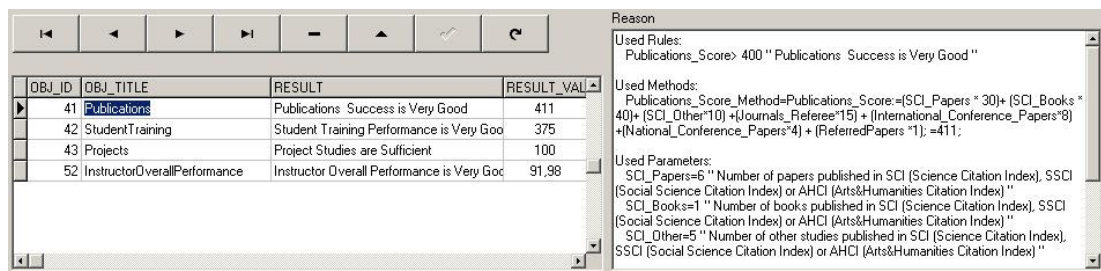


Figure 3.18 Screenshot of INES ES for instructor evaluation

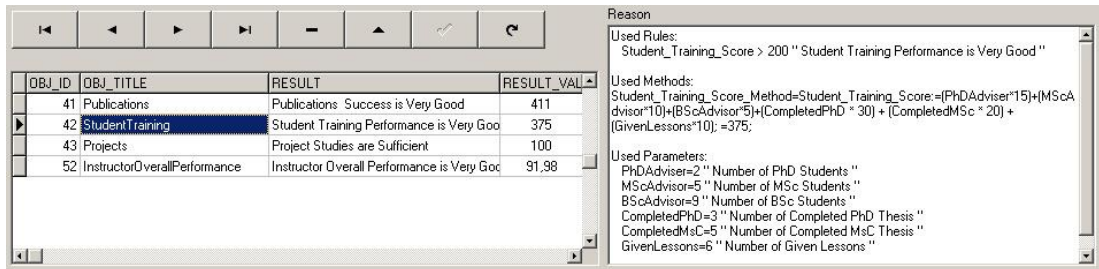


Figure 3.19 Screenshot of INES ES for instructor evaluation

Screenshots of INES for instructor evaluation result and the reasons for “Student Training” and “Project” evaluation objectives are shown in Figure 3.19 and Figure 3.20.

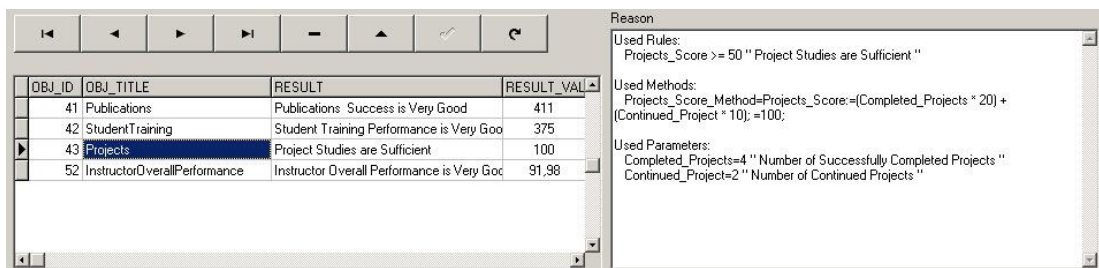


Figure 3.20 Screenshot of INES ES for instructor evaluation

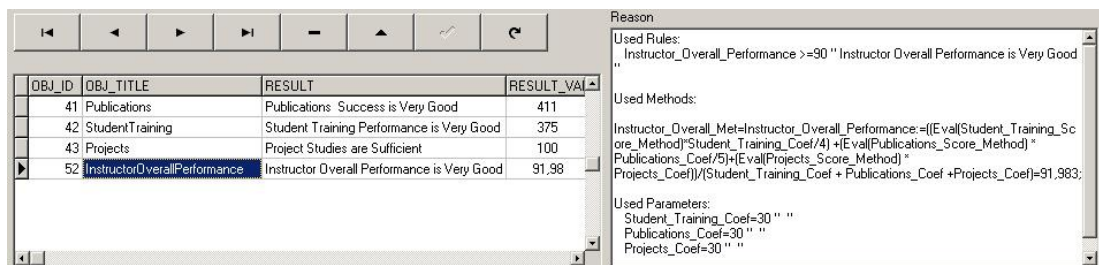


Figure 3.21 Screenshot of INES ES for instructor evaluation

3.4.2.1 Instructor overall evaluation

The overall evaluation of instructor performance was done in two ways:

1. Using INES Expert System
2. Using INES Fuzzy Logic

Using INES Expert System: The overall evaluation was calculated as a function of “Publication Success”, “Student Training” and “Project Success”

Screenshot of INES ES for instructor evaluation and the reason of “Overall Performance” result is shown in Figure 3.21.

Using INES Fuzzy Logic for instructor: The results of instructor evaluation in the previous section were assessed by fuzzy logic module using Matlab.

The rules of instructor Evaluation Fuzzy System are shown in Table 3.2 with “Publication Success”, “Teaching” and “Project Success” as inputs and “Overall Performance” as output. For example, The meaning of the table for the underlined cell is

If (Project Success is bad) AND (Project Success is low) AND (Teaching is Bad), then (Overall Performance is bad).

The meaning of table for other cells is defined in similar way.

Several types (Gaussian, triangle and trapezoid) of membership functions were tried and compared with ES results of instructor evaluation. The comparison of the overall performance results generated by INES ES and Fuzzy Logic for different values of “Publication Success”, “Teaching” and “Project Success” values is shown in Figure 3.22. The results generated from fuzzy logic using Gaussian membership functions seems the most approximate to the results of ES.

Table 3.2 The rules of Instructor Fuzzy Evaluation System

Teaching	Bad	Good	VGood
Publication Success			
Low	<u>Bad</u>	lgood	good
Average	lgood	good	good
Very Good	good	good	vgood

Project Success is bad

Teaching	Bad	Good	VGood
Publication Success			
Low	lgood	good	vgood
Average	good	vgood	excellent
Very Good	vgood	excellent	excellent

Project Success is good

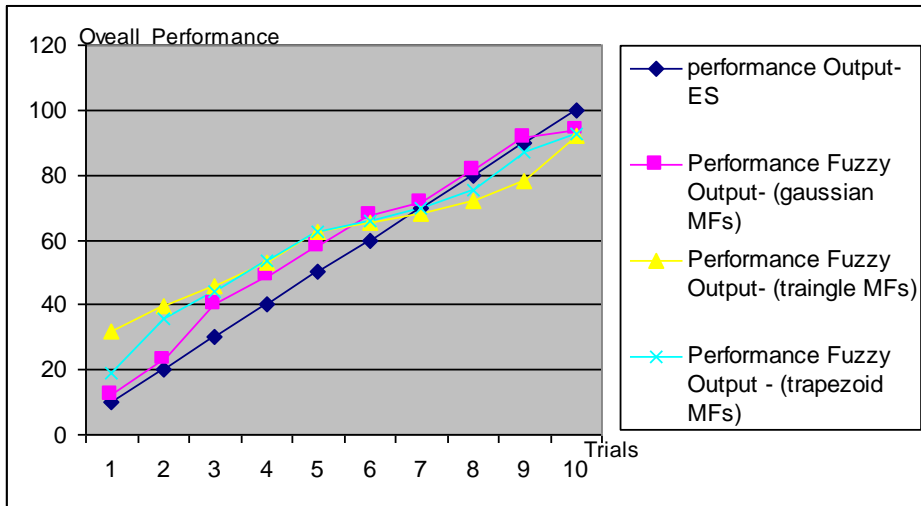


Figure 3.22 The results generated by INES ES and Fuzzy Logic for different membership functions

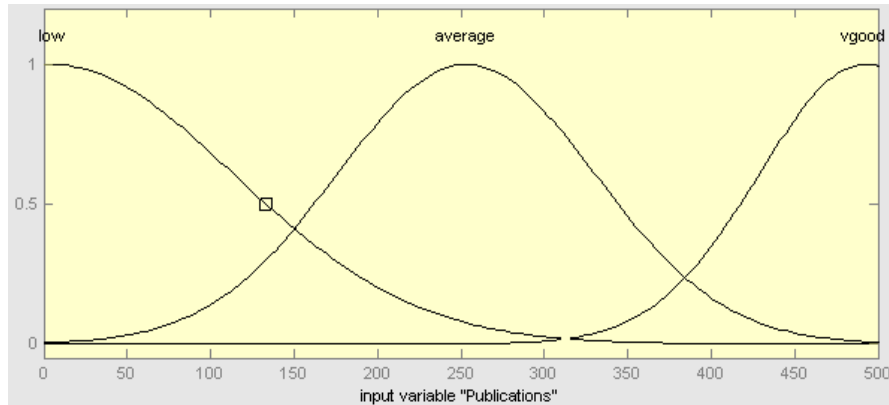


Figure 3.23 Membership Function for input variable “Publications”

The membership functions of “Publication Success”, “Teaching” and “Project Success” as inputs and Overall Performance as output is shown in Figure 3.23, Figure 3.24, Figure 3.25 and Figure 3.26.

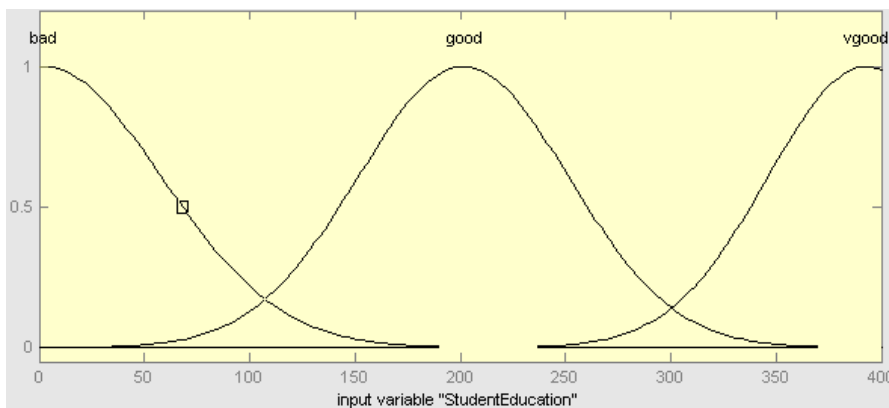


Figure 3.24 Membership Function for input variable “StudentTraining”

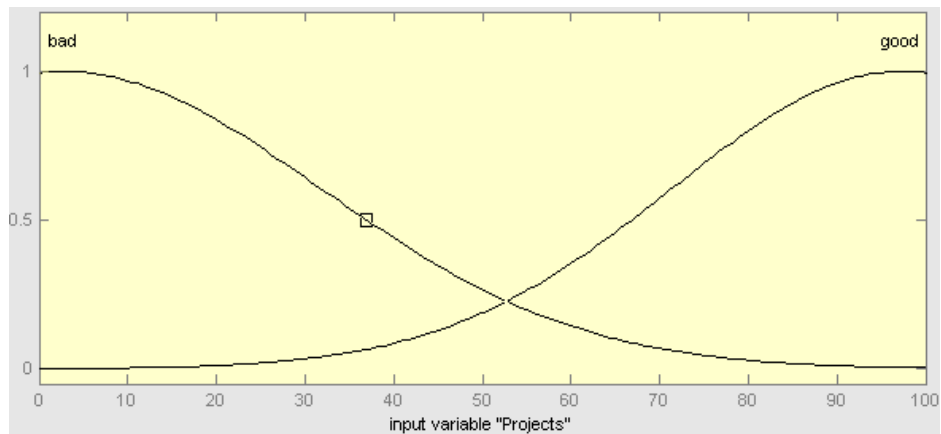


Figure 3.25 Membership Function for input variable “Projects”

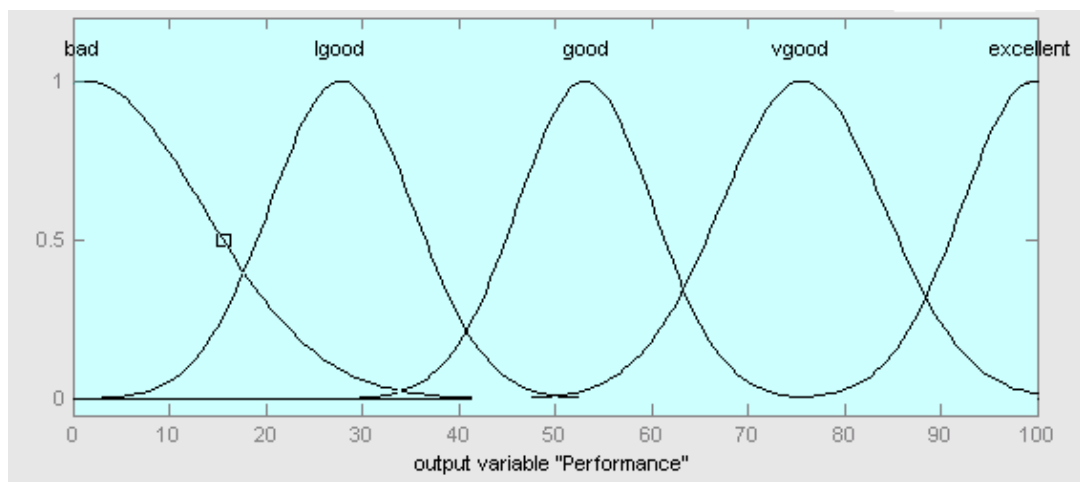


Figure 3.26 Membership Function for output variable “Performance”

The overall result of instructor evaluation using fuzzy logic evaluation system is shown in Figure 3.27. All parts of the fuzzy inference process are simultaneously displayed. There are 18 rows and each row shows the fuzzy inference process of a rule. For example, Row 18 shows the following rule;

If (Publications is very good) AND (StudentEducation is very good) AND (Projects is good), then (Overall Performance is excellent).

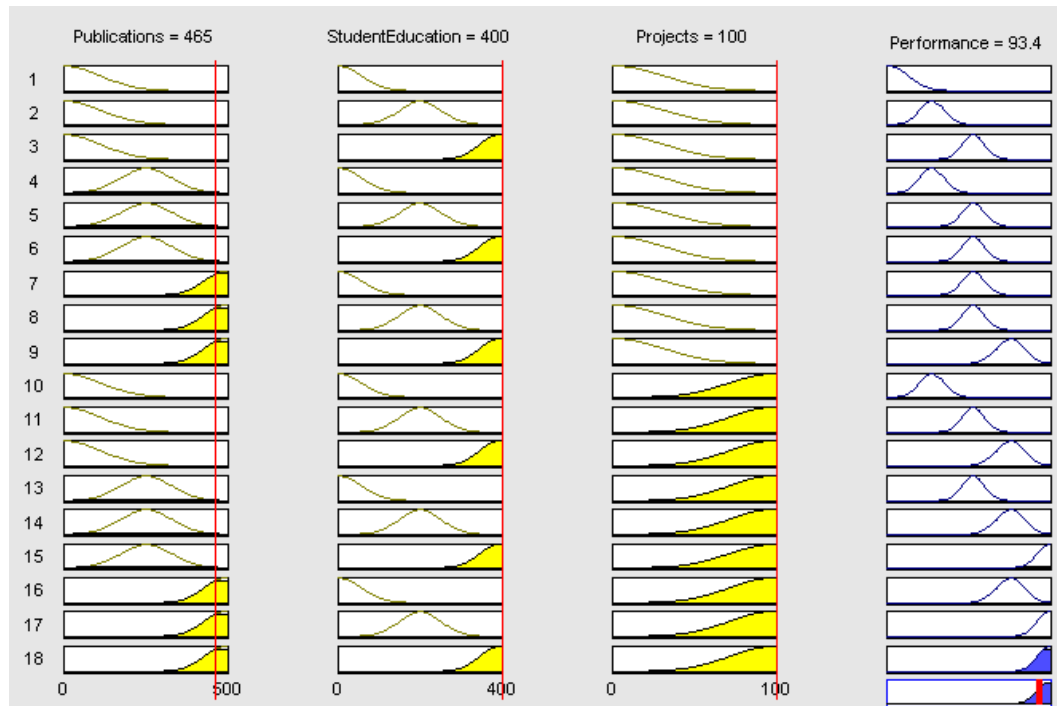


Figure 3.27 An example result of Instructor Evaluation Fuzzy System

The right bottom rectangle shows the output for this instance, which is the combination of output of rules. The centroid calculation was used for generating the crisp output. This method returns the center of area under the output curve.

3.4.3 Pilot evaluation

In this example, following sub-objectives were identified for pilot performance evaluation [63]:

1. Launch Success: The ratio between number of successful launches and total shots.
2. Destroy Success: The ratio between number of destroyed and total enemies.
3. Breaking the Lock: The ratio between number of broken launches and total shots. If the pilot can get rid of the threats, it will affect the performance of the pilot positively.
4. Defeated Missiles: Total number of missiles missed aircraft or defeated by aircraft.

5. Finalizing the Mission: If the mission is completed, pilot will be considered as successful for this sub-objective.
6. Fuel Level: If the mission is completed without finishing the fuel (much fuel level is better), the pilot will be considered as successful.
7. Validity of Shots: The ratio between valid shots and invalid shots. For example, if a weapon is fired out of range, the shot will be invalid.
8. Pilot Overall Performance: a function of all sub-objectives (launch success, etc)

The performance of pilot instructor was evaluated by applying several rules that use various measures. These measures were

- Launch_success
- Destroy_Ratio
- Breaking_the_Lock
- Ratio_ata_missiles (air to air)
- Finalizing_the_Mission
- Fuel_Level
- Validity of Shots
- Pilot_Performance_Overall

Following rules were used as criteria to evaluate the launch success:

- If (Launch_success \geq 85) and (Launch_success \leq 100) THEN launch capability is very good
- If (Launch_success \geq 70) and (Launch_success $<$ 85) THEN launch capability is good
- If (Launch_success \geq 40) and (Launch_success $<$ 70) THEN launch capability is sufficient
- If Launch_success $<$ 40 THEN launch capability is insufficient

The method for calculating the Launch Success is as follows:

$$\text{Launch_Success} = (\text{hits} / \text{shots}) * 100$$

Where

- Hits: hits done by aircraft
- Shots: total shots done by aircraft

Following rules were used to evaluate the destroy success:

- If (Destroy_Ratio < 40) THEN Destroy Success is insufficient
- If (Destroy_Ratio >= 40) and (Destroy_Ratio <= 100) THEN Destroy Success is sufficient

The method for calculating the Destroy success is as follows:

$$\text{Destroy_Ratio} = (\text{destroyed_threats} / \text{total_threats}) * 100$$

Where

- destroyed_threats: total number of destroyed threats
- total_threats: total number of threats

The following rules were used as criteria to evaluate the success in breaking the lock:

- (Breaking_the_Lock >= 80)
- (Breaking_the_Lock > 60) and (Breaking_the_Lock < 80)
- (Breaking_the_Lock <= 60)

The method for calculating the breaking the lock success is as follows:

$$\text{Breaking_the_Lock_Score} = (\text{broke_locks} / \text{total_locks}) * 100$$

Where

- broke_locks : Total number of broke locks
- total_locks : Total number of locks

OBJ_ID	OBJ_TITLE	RESULT	RESULT_VAL
1	Launch Success	launch capability is sufficient	50
5	Breaking the Lock	good	62.5
16	Destroy Success	Sufficient	50
20	Defeated Missiles	Very Good	75

Reason

Used Rules:
 (Breaking_the_Lock > 60) and (Breaking_the_Lock < 80) " good "

Used Methods:
 Breaking_the_Lock_Met1=Breaking_the_Lock:=(broken_locks / total_locks)*100;
 =62.5;

Used Parameters:
 Broken_locks=5 " Total number of broken locks "
 total_locks=6 " Total number of locks "

Figure 3.28 Screenshot of INES' ES for pilot performance evaluation

Following rules were used to evaluate the defeated missiles:

- If (Ratio_ata_missiles < 40) THEN Destroy Success is insufficient
- If (Ratio_ata_missiles >= 40) and (Destroy_Ratio <= 100) THEN Destroy Success is sufficient

The method for calculating the Ratio_ata_missiles is as follows:

$$\text{Ratio_ata_missiles} = (\text{airtoair_ag_ac} / \text{total_ata}) * 100$$

Where

- airtoair_ag_ac: umber of defeated air_to_air missiles fired against aircraft
- total_ata: total number of air_to_air missiles fired against aircraft

An example screenshot of INES for pilot evaluation result is shown in Figure 3.28.

Information for the methods and rules of “Finalizing the Misssion”, “Fuel Level”, “Validity of Shots” and “Pilot Overall Performance” can be found in INES Knowledge Base software.

3.4.4 Personnel selection

Recruitment and selection of candidates and employees' appraisal are important processes in human resources management of any organization. The job market becomes more dynamic than ever before [80]. Wrong personnel selections cause the loss of time and money. In the following example, evaluation of personnel for a research institute of information technologies developed with INES is given.

Following evaluation sub-objectives were identified for personnel selection:

1. Grade MSc and BSc: The Average Grade of MSc and BSc

2. English Level: English Skills
3. Experience: Years of experience
4. Publications: Achievement in publications
5. Computer Skills: Software experience and knowledge

Note that, the evaluation objectives and evaluation definition are subject to change according to the intention of SMEs and as well as relevant expertise.

The personnel were evaluated by applying several rules that use various measures.

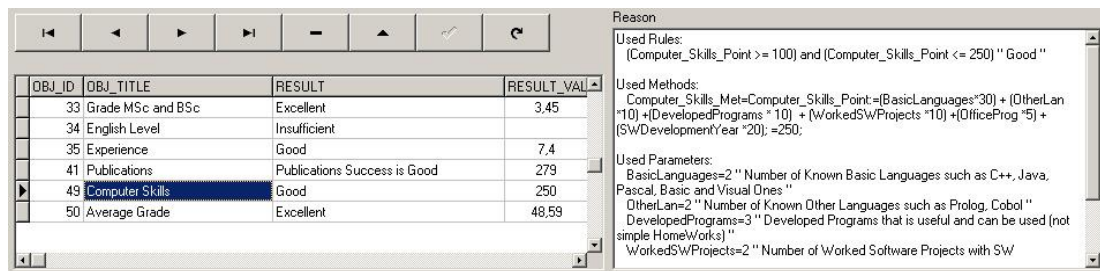


Figure 3.29 Screenshot of INES results for a person evaluation

An example screenshot of INES results for a person evaluation and the reason for “Computer Skills” evaluation objective is shown in Figure 3.29.

Information about the methods and rules of “Grade MSc and BSc”, “English Level”, “Experience” and “Publications” can be found in INES Knowledge Base software.

3.5 Distributed Evaluation

Distributed evaluation is a way to delegate and partition automatic evaluation between evaluated entities. Instead of sheer centralized evaluation, the analysis and evaluation of data will be disseminated using an evaluation master agent (in the following just called ‘master agent’) and evaluation agents, referring to different levels of evaluation [24, 81].

An agent is anything that can be viewed as perceiving the environment through sensors and acting upon environment through effectors [3]. Multi-agents systems are composed of multiple, interacting agents [59]. There are different application areas of multi-agents such as distributed simulation, decision support, computer games, learning, control of robots, telecommunications, etc [59, 82, 83].

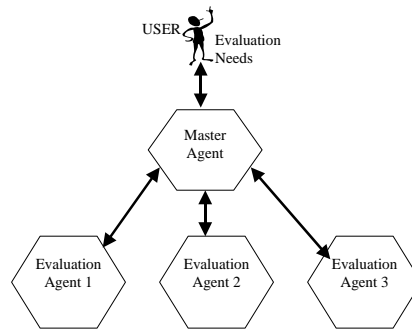


Figure 3.30 Distributed evaluation using master and evaluation agents

The study indicates that multi-agents can be used for distributed evaluation. In distributed environments, evaluation can split into two parts:

- Overall evaluation
- Evaluation at clients (or at sub systems)

Different types of multi-agent organizations can be developed and designed. In the first organization type, user controls master agent to handle other agents and master agent collects knowledge from evaluation agents in order to do overall evaluation as shown in Figure 3.30. Master agent generates evaluation definition information, where evaluation objectives, rules, measures, methods, parameters, questionnaires, and their relationships are stored. Then master agent sends generated evaluation definition messages (or files) to the evaluation agents. Each evaluation agent is capable of doing partial evaluation at client side. Evaluation agents collect information/data from the environment, evaluate the data and send results to the master agent. The master agent analyses evaluation results from the evaluation agents and provides user final evaluation results. Both, master and evaluation agents can be referred to as evaluation environment.

Different types of multi-agent organizations can be designed and developed. In the first type organization, evaluation agents can't communicate with each other. In the second type, evaluation agents can communicate with each other as shown in Figure 3.31. This organization can be used in the situation where the results generated by evaluation agents can influence the results of other agents.

The typical steps for distributed evaluation with using intelligent multi-agents are as follows [24]:

1. User (i.e. evaluator) determines what will be evaluated.
2. Master agent determines, what has to be evaluated by each evaluation agent. The master agent divides the tasks according to the evaluation agents' capabilities

and the environment where the evaluation agents perform evaluation. Each task is a subset of the overall evaluation.

3. The master agent generates evaluation definition messages (or files) for each evaluation agent.
4. The master agent sends generated evaluation definition messages (or files) to the evaluation agents.
5. After or during the execution of the exercise, the master agent evaluates high level aspects and the evaluation agents evaluate low level.
6. The evaluation agents send their evaluation results to the master agent.
7. The master agent evaluates all results and generates final evaluation results and presents to the user.

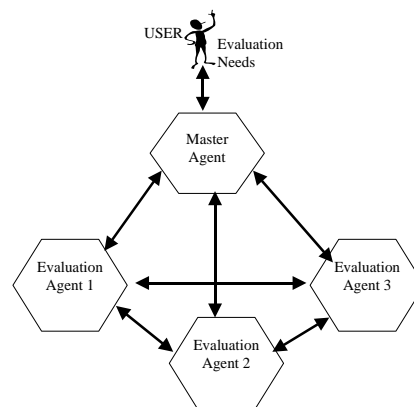


Figure 3.31 Distributed evaluation using master and evaluation agents

3.5.1 Advantages of distributed evaluation

The advantages of distributed evaluation are as follows:

- It provides solutions that efficiently use information sources that are spatially distributed [82]. Delegation and partitioning of evaluation, simplifies the evaluation process.
- It reduces network traffic during exercise execution [81].
- Certain sensitive data has not to be sent across the network. Only pre-processed data, i.e. evaluation results, are distributed across the network. This reduces security issues of evaluation [81].

- It can enhance performance along the dimensions of computational efficiency, reliability, extensibility, maintainability, flexibility and reuse [82].

3.5.2 Knowledge Representation

The master agent contains all the domain evaluation knowledge and evaluation agents contain partially evaluation knowledge, which is needed for local evaluation at client side. Integrated “Reference Model of Evaluation Objectives” and “Evaluation Definition Knowledge” can be used to represent evaluation knowledge of master agents as shown in Figure 2.2. Evaluation agents knowledge representation can be similar expect not including “Evaluation Objectives Hierarchical Tree” and “Evaluation Objectives”.

The main components of master agent are shown in Figure 3.32. Evaluation Agents structure is similar with master agent except not including a user interface.

The main components of master agent are as follows:

- Perception to perceive the events and information from the environment (e.g. receiving evaluation results from evaluation agents)
- Cognition to reason about perceived events. Rule-based systems such as expert system can be used for inferencing. As the evaluation includes uncertainty in some aspects, fuzzy logic can also be incorporated with expert system in the inference engines of master and evaluation agents for reasoning. Knowledge Base of agents contains the knowledge and expertise of Subject Matter Experts for performing evaluation in a structured format.
- Action to act according to the reasons produced by the cognition mechanism (e.g. sending evaluation definition information to evaluation agents).
- User interface to control the dialog between the user and the system and present the evaluation results to the user.

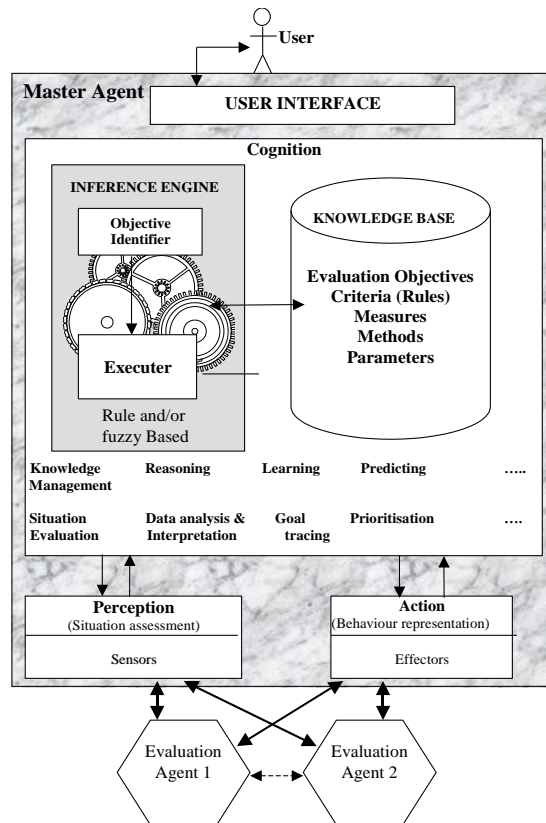


Figure 3.32 The main components of master agent

3.6 Overall Results of Implementation Studies

In the above implementation, INES was used for different domains such as AD System evaluation, instructor performance evaluation [84], pilot performance evaluation and personnel selection. The study indicated that it is possible to put evaluation knowledge into a structured format.

As mentioned before, evaluation process is ill defined by nature. It changes according to the SMEs. INES was developed to handle the heuristic knowledge of experts from different domains and information from different sources for evaluation purposes. For this reason, the results generated by INES were not compared with other tools results.

The execution time of different applications on different computers with using INES is listed in Table 3.3. INES can be used in real time applications, because the evaluation time of INES is below 1 sec (between 0,4-0,7 sec in fact and can be decreased with optimization) and is enough for many systems' real time requirement of evaluation.

Table 3.3 INES' execution time for different applications

	Number of Sub-Objectives	Number of Rules	Execution Time Laptop Pentium 4 1.7 GHz 512 Mbytes RAM	Execution Time PC Pentium Celeron 1.7 GHz 256 Mbytes RAM	Execution Time PC Pentium 4 2.8 GHz 1 Gbytes RAM
Pilot Performance	7	19	0,661	0,766	0,453
AD System	3	10	0,641	0,718	0,438
Job Applicants	3	5	0,621	0,719	0,437
Instructor Performance	3	8	0,631	0,719	0,438
Student Performance	4	8	0,631	0,734	0,438

4. COMPARISON, (DIS)ADVANTAGES OF INES WITH SIMILAR TOOLS

Resembling previous studies found in the literature are as follows and explained briefly in Section 1.4:

- SIMULTAAN PASS [62]
- RTP 11.12 Performance Evaluation System [63]
- RTP 11.13 EDST+EDT+EET [64, 65]
- ESTA Evaluation [66]
- ESSE (Expert System for Software Evaluation) [21]

The initial version of INES was developed by ESTA. It was realised that COTS tools such as ESTA have some disadvantages such as poor GUI and functionality. It was difficult to add new knowledge to the system. For these reason, INES was developed using High Level Developing Language (Borland Delphi 5) to satisfy the requirements of Intelligent Evaluation System and overcome the obstacles related with COTS tools.

4.1 Comparison of INES with Similar Tools

The comparison of INES with SIMULTAAN PASS, EDST+EDT+EET, Performance Evaluation System (PES) ESTA evaluation and ESSE is shown in Table 4.1.

As seen from Table 4.1, evaluation systems are generally domain dependent (SIMULTAAN PASS, PES, ESSE) and don't provide the explanation of reasoning (EDST+EDT+EET, SIMULTAAN PASS, PES, ESSE). It is impossible or difficult to update and add knowledge without source code change (PES, ESSE).

Table 4.1 Comparison of INES with Resembling Previous Studies

	INES	Resembling Previous Studies				
		EDST +EDT +EET	SIMULT- AAN PASS	PES	ESTA Evaluation	ESSE
Artificial Intelligence	Yes	Partially	No	No	Yes	Yes
Application area	General (Training, Rehearsal, SBA,..)	General (Training, Rehearsal, SBA,..)	Training	Pilot Evaluation	General (Training, Rehearsal, SBA,..)	Software Evaluation
Main purpose	Evaluation of Systems, Synthetic Environme nts, humans	Evaluation of Synthetic Environments	Automatic analysis and assessment of trainee and team performance	Evaluation of Pilot Performance	Developing expert systems	Software problem solving and software attribute assessment
Knowledge Details	Evaluation Objectives, criteria, methods and rules	Evaluation Objectives, criteria, methods and rules	Scenario- specific actions and action- related judgement rules	Pilot Evaluation Objectives & indexes, rules	Knowledge Bases	Multiple criteria, Software attributes
Knowledge Representation	Rule Based	Rule Based	Action- related judgement rules	Algorithmic	Rule Based	Rule Based
Updating Knowledge without source code change	Yes	Yes	Yes	No	Yes	Partially
Explanation of Results	Provided	Not Provided	Not Provided	Not Provided	Provided	Partially Provided
Maintenance and update	Easy	Easy		Difficult	Difficult	
Reasining Capability	Yes	Yes		No	Yes	
Structure	Inference Engine and Knowledge Bases are seperated	Manually Inferencing		Control integrated with information		
Usability	Easy	Difficult		Easy	Difficult	
Learning Itself	No	No	No	No	No	No

4.2 Advantages / Disadvantages of INES With Respect to Similar Tools

INES has the following advantages with respect to similar tools and manually evaluation:

- 1) The INES can be used for various domains such as pilot evaluation, instructor evaluation, trainee evaluation, evaluation of job applicants (nearly evaluation of all domains provided that required knowledge is provided)
- 2) Fuzzy Logic (FL) was integrated to the INES for doing overall assessment of results of executing ES in the highest level. FL was used to model the uncertainty about overall evaluation and provides reasoning on linguistic variables.
- 3) The INES can be used to handle the heuristic knowledge of experts from different domains and information from different sources about evaluation.
- 4) The INES explains the reason of evaluation results.
- 5) The INES allows the user to add new Knowledge and to modify existing ones without source code changing.
- 6) The INES can reduce the time required to accomplish evaluation tasks.
- 7) The INES is highly flexible with regard to the applied evaluation process and evaluation rules.
- 8) The INES can be used to access the captured knowledge of SMEs. By this way, each expert can benefit from others' knowledge
- 9) Generally, the instructors cannot monitor more than one trainee at a time. When the instructor changes from one trainee to another, he must transit from the original trainee's lesson and performance to that of a new student [4]. The INES can support instructors to evaluate more than one trainee (e.g. pilots) simultaneously.
- 10) Synthetic Environments and systems are getting more complex day by day. Most evaluations are done manually or using conventional computing and performing these evaluations are difficult, inefficient and inflexible. The INES can perform evaluation of SEs, systems and humans automatically according to the INES ES Knowledge Base easily.

The INES has the following disadvantages:

- 1) If the knowledge in the KB is not enough to perform a good evaluation, system produces poor evaluation results
- 2) It is hard to extract expertise from evaluation experts.
- 3) Evaluation knowledge is not always readily available. There are very few Subject Matter Experts (SMEs) being able to evaluate systems and SEs especially complex ones.
- 4) The approach of experts to define evaluation criteria can be different, yet correct.

The INES may have other advantages and disadvantages of ES and FL.

5. CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Information technology exhibits a remarkable progress and achievements especially in the field of Artificial Intelligence (AI). Enriching the military and non-military systems with intelligent capabilities makes them more powerful and more reliable [1]

In this thesis, it was shown that AI technology could be beneficial to decrease evaluation cost and evaluation time of systems, synthetic environments and human performance. AI can also provide some other functionalities for evaluation purposes such as understanding the reason of inferencing, updating required knowledge without source code changing and simplifying evaluation process.

Forming an evaluation definition *is a complicated and time-consuming task*. Finding out and formulating the required knowledge from the domain for which the evaluation is to be performed, is generally difficult due to lack of structured approach. It is not only important to formulate the knowledge, but also finding out the right source of knowledge is essential. Structured knowledge architecture is especially important in order to utilize evaluation knowledge automatically, especially in distributed environments. This study indicates that it is possible to put knowledge related with evaluation into a structured format. A process and methodology for forming an evaluation definition and performing evaluation according to this definition were developed and explained.

Scientific contributions of the thesis can be listed as follows:

- Common Evaluation Process (CEP), which can be used at evaluation of synthetic environments and human performance (instructors, pilots, job applicants, etc.) as well as real systems, was developed. CEP is a domain independent process for evaluation purposes. SEDEP, FEDEP, STEP, SAT processes, which includes evaluation steps were investigated and taken into account during the development of CEP. SEDEP and FEDEP evaluation related steps were mapped to CEP steps.

- A methodology was developed to handle the heuristic knowledge of experts from different domains and information from different sources for evaluation purposes. The knowledge was represented as reference model of evaluation objectives, production rules, measures, methods and parameters.
- The Common Evaluation Model, which is a knowledge representation of the CEP, was developed. CEM shows the relation between evaluation objectives, rules, measures, methods and parameters. Using “Reference Model of Evaluation Objectives” and “Common Evaluation Model” decreases the number of evaluation rules that is necessary to perform evaluation to the related application. CEM also simplifies the representation of evaluation knowledge.
- In this study the expression of “condition” in expert system rule definition is extended as a function of measures or/and parameters (See Section 3.1.2 for details).
- The high level requirements of Evaluation Tools were identified.
- An Expert System (ES), which is called INES (INtelligent Evaluation System) ES and can be used domain independently for evaluation purposes, was developed according to the determined requirements. Before development of INES, AI techniques including expert systems, fuzzy logic, neural networks, genetic algorithms, intelligent agents and conventional programming were investigated and compared with respect to achieving high level requirements of Evaluation Systems. INES ES was developed using Borland DELPHI5 to provide a user-friendly interface.
- INES Expert System was implemented for the first time in the following areas:
 - Air Defence (AD) System Evaluation,
 - Instructor performance evaluation (University Staff)
 - Pilot Performance Evaluation
 - Personnel Selection
 - Student Evaluation

- Fuzzy Logic was utilized for evaluation purposes and integrated to the INES expert system. Matlab was used for the development of fuzzy systems and providing a user-friendly interface.
- INES Fuzzy Logic was implemented for the first time in the following areas:
 - Air Defence (AD) System Overall Evaluation,
 - Instructor overall performance evaluation (University Staff)

INtelligent Evaluation System (INES), which was developed on the basis of the CEP, shows the applicability of the CEP. INES was compared with resembling previous studies and advantages / disadvantages of INES with respect to similar tools were identified and given in previous sections. Some benefits of INES, which most of them AI related, are as follows:

- Speeding up evaluation process and decreasing evaluation cost
- Explaining how the system reaches evaluation results.
- Modelling the uncertainty about overall evaluation and providing reasoning on linguistic variables.
- Providing flexible structure
- Allow user to update/add evaluation knowledge base without changing source code.
- Reducing the complexity associated with the evaluation
- Increasing confidence on the knowledge utilised in the evaluation process
- Providing objective and reliable evaluation

INES was successful and tested in the following conditions:

- Knowledge of experts from the related domain and knowledge (or information) from the related sources for evaluation purposes is existed
- Identifying evaluation criteria from the expert knowledge and information from different sources is possible

INES's Knowledge Base (KB) and KB Editor were developed for forming, editing and updating evaluation knowledge. INES's Inference Engine was developed for executing the evaluation definition, which includes evaluation objectives, production rules, measures, methods and parameters.

Separating the knowledge base from inference engine is important in order to make the knowledge base easily editable and modifiable. Besides it worth to note that the architecture of the knowledge base may need to be updated upon receiving new knowledge, which are not encountered.

For developing an evaluation system, it is very advised to develop an abstract model and improve it in time, if the amount of knowledge is huge.

The evaluation knowledge generally cannot be acquired from only a single expert especially for military applications. Therefore constituting a working team including evaluation experts from different domains is necessary.

INES is a powerful tool for evaluation purposes, but there are still some shortcomings of the system. These shortcomes are mainly the nature of expert systems.

As the evaluation includes uncertainty in some aspects, Fuzzy Logic was used for reasoning. But it was realised that Fuzzy Logic could be used to perform overall performance or assessment instead of evaluation itself for complex tasks. In other words, fuzzy logic can be more beneficial and more easily used for overall evaluation of main objective instead of all aspects of evaluation. A lot of parameters for evaluation are required and writing a lot of rules for these parameters in fuzzy logic is not an efficient method. As more rules are needed for complex systems, it becomes increasingly difficult to relate these rules to the system. *The capability to relate the rules typically diminishes when the number of rules exceeds approximately 15* [55]. Therefore, fuzzy system was used at an abstract level.

It is possible that INES can be used in real time applications, as the evaluation time of INES is under 1 sec and can be decreased with optimization and is enough for many systems' real time requirement of evaluation.

5.2 Future Work

In this study, generating the output of the evaluation objectives, definition and results in XML format was considered and developed for providing evaluation knowledge to other tools upon user request. Knowledge in the INES KB was represented in

database files and hierarchical tree. The Knowledge in the KB can be represented using XML (eXtensible Markup Language) (See Section 3.2.2.2 for XML). An XML language can be developed for representing evaluation knowledge.

Knowledge Base Editor can be improved. For example, the formulations can be shown graphically.

The Knowledge Base of INES can be updated with new knowledge from other evaluation areas. Note that the architecture of the INES Knowledge Base may need to be updated upon receiving new knowledge, which are not encountered.

During this study, INES was tested by several users. The INES should be tested by more users to identify the difficulties, which users may encounter.

Other hybrid artificial intelligence technologies such as neuro-fuzzy, expert-genetic algorithms, etc can be used to improve success of evaluation systems.

The study highlights that multi-agents can be used for distributed evaluation. The proposed methodology is applicable to different types of multi-agent organizations. Multi-agents can simplify the evaluation process and reduce network traffic in distributed environments and decrease security problems among the network.

NLP can be used for questionnaire analysis and automatic speech analysis of trainees such as pilots especially during team training.

Self-Learning Evaluation Systems can be developed.

Common Evaluation Process defined in this study can be improved and detailed.

REFERENCES

- [1] **Öztemel, E. and Öztürk, V.**, 2003, Intelligent Evaluation Definition of Training Systems in Synthetic Environments, *ITEC2003 Conference*, UK
- [2] **Hornung, B.**, 1995, Bernd Hornung's Glossary definitions <http://pespmc1.vub.ac.be/ASC/hornung.html>
- [3] **Russell, S. and Norvig, P.**, 1995, Artificial Intelligence: A Modern Approach, Prentice Hall
- [4] **Drewes, P. and Gonzalez, A.**, 1994, Automatic performance monitoring and evaluation, *IEEE Proceedings of the Fifth Annual Conference on AI, Simulation, and Planning in High Autonomy Systems, Distributed Interactive Simulation Environments*, 274-280
- [5] **Pace, K. D.**, 1999, Use of Subject Matter Experts (SMEs) in Simulation Evaluation, 1999 Fall SIW (Simulation Interoperability Workshop) http://www.sisostds.org/doclib/doclib.cfm?SISO_FID_3321
- [6] **Dumetz, J. and Fabbri, M.**, 2002, Are there Obstacles that prevent Synthetic Environment Exploitation in Europe? Findings from EUCLID RTP 11.13 "Realising the potential of Networked Simulation in Europe", *3rd SMI Conference Virtual Battlefields: Synthetic Environment Training*, London, UK http://www.euclid1113.com/Published_Papers/Conferences/2002_SMI/SMI%20virtual%20battlefield.doc
- [7] **Sandeep, S. M., Karen A. H. and Greg L. Z.**, 2000, SAMPLE: Situation Awareness Model For Pilot-In-The-Loop Evaluation, *9th Conference on Computer Generated Forces (CGF)*, SISO
- [8] **Rigg, G., Morley R. and Hepplewhite R.**, 2000, Themis: The Objective Assessment of CGF Performance, *9th Conference on Computer Generated Forces & Behavioral Representation (CGF-BR)*, SISO
- [9] **Volant, M., Guillemin, L., Fabbri, M., Raivio, J., Tuukkanen, J., Lemmers, A. Keuning, M., Diaz de Rio, V. and Warleta, J.**, 2001, Deliverable for WE 1.4 User and System Requirements Technical Report, The Euclid RTP 11.13 Realising the Potential of Networked Simulation in Europe, http://www.euclid1113.com/deliverables/reports/WE1.4/RTP1113_WE1.4_report_2.0.doc
- [10] **Bass, E. J.**, 1998, Architecture for an intelligent instructor pilot decision support system, *IEEE International Conference on Systems, Man, and Cybernetics*, **1**, 891 –896

- [11] **Gregory, W. H.**, 1998, Evaluating Simulation Based Training Scenarios, 1998 Spring SIW (Simulation Interoperability Workshop) http://www.sisostds.org/doclib/doclib.cfm?SISO_FID_897
- [12] **Hemel, P. and Kisg, W. J.**, 1981, Simulation Techniques In Operator And Maintenance Training, *Performance Assessment, And Personnel Selection, Comput. & Indus. Eng.*, **2.**, 105-112. 1981, UK
- [13] **Sanders, J.**, 2001, A Vision for Evaluation, *American Journal of Evaluation*, **22(3)**, 363–366
- [14] **LinguaLinks Library**, 1999a, Version 4.0, published on CD-ROM, SIL International, <http://www.sil.org/lingualinks/literacy/ReferenceMaterials/GlossaryOfLiteracyTerms/WhatIsEvaluation.htm>
- [15] **Trochin, W. M.K.**, 2004, Introduction to Evaluation, <http://www.socialresearchmethods.net/kb/intreval.htm>
- [16] **Farmer, E. van Rooji, J., Riemersma, J., Jorna, P. and Jan, M.**, 1999, Handbook of Simulator-Based Training, Ashgate Publishing Ltd, Great Britain
- [17] **Hartley, R. and Varley, G.**, 2001, The Design And Evaluation For Development Of Complex Decision Making Skills, Proceedings. *IEEE International Conference on Advanced Learning Technologies*, 2001
- [18] **Shub, Y., Kushnir, A. and Frenkel, J.**, 1994, Pilot Evaluation System, *Aerospace and Electronics Conference, NAECON, Proceedings of the IEEE*, vol.2, 734 -741
- [19] **Oliveira, J.C., Shirmohammad, S. and Georganas, N.D.**, 1999, Collaborative Virtual Environment standards: a performance evaluation, 3rd *IEEE International Workshop on Distributed Interactive Simulation and Real-Time Applications*, 22-23 Oct., 14 – 21
- [20] **Macannuco, D., Hung, J. and Civinskas, W.**, 1998, A Test Suite to Evaluate Run-Time Infrastructure (RTI) Implementations for High Performance, Human-In-The-Loop (HITL) Simulators, 1998 Spring SIW (Simulation Interoperability Workshop), http://www.sisostds.org/doclib/doclib.cfm?SISO_FID_1260
- [21] **Vlahavas, I., Stamelos, I., Refanidis, I. and Tsoukias, A.**, 1998, ESSE: an expert system for software evaluation, *Knowledge-Based Systems*, 183–197
- [22] **Miskell, S.G., Happell, N. and Carlisle, C.**, 1989, Operational Evaluation of an Expert System: The FIESTA Approach, Heuristics, *The Journal of Knowledge Engineering*, 2(2), Systemware Corporation, Rockville, Maryland
- [23] **DODD 5000.59-M, Online M&S Glossary**
<https://www.dmsomil/public/resources/glossary>
- [24] **Rollesbroich, B., zu Drewer, P. M., Greiwe, K. Jokipii, M. and Hartikainen, T.**, 2003a, Common Evaluation Framework &

- [25] **Wilcox, P.A., Burger A.G. and Hoare P.**, 2000, Advanced distributed simulation: a review of developments and their implication for data collection and analysis, *Simulation Practice and Theory*, 8, 201-231, Elsevier Science B.V.
- [26] **TRADOC Regulation 350-70**, 1999, Part III, Evaluation and Quality Assurance, United States Army Training and Doctrine Command, www.tradoc.army.mil/tpubs/regs/r350-70/
- [27] **Bhola, H.**, 1990, Evaluating "Literacy for development" projects, programs and campaigns: Evaluation planning, design and implementation, and utilization of evaluation results, UNESCO Institute for Education, DSE, Hamburg, Germany
- [28] **LinguaLinks Library**, 1999b, Version 4.0, published on CD-ROM, SIL International, <http://www.sil.org/lingualinks/literacy/ReferenceMaterials/GlossaryOfLiteracyTerms/WhatIsPreTrainingEvaluation.htm>
- [29] **UK MoD web site**, <http://www.mod.uk/issues/simulation/glossary.htm>
- [30] **Dumetz, J.**, 2002, The Euclid RTP 11.13 Realising the Potential of Networked Simulation in Europe, *ITEC 2002*, England
- [31] **DoD 5000.59-P**, 1995, Modeling and Simulation Master Plan, October 1995
<https://www.dmsomil/public/resources/glossary/> and
<https://www.dmsomil>
- [32] **Maamar, Z.**, 2003, Design of a simulation environment based on software agents and the high level architecture, *Information and Software Technology*, 45, 137–148, Elsevier
- [33] **Campbell, L., Lotwin A., DeRico, M.G. and Ray, C.**, 1997, *IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation*, 3
- [34] **Mephram, S.**, 1998, Synthetic Environments - Delivering Real Benefits for UK Defence, *Simtect 98 Proceedings*
- [35] **Ford, K. and Jokipii, M.**, 2003, The Euclid RTP11.13 SE Management Tool, *EuroSIW (Simulation Interoperability Workshop)*,
<http://www.sisostds.org/index.php?tg=fileman&idx=get&id=2&gr=Y&path=Simulation+Interoperability+Workshops%2F2003+EURO+SIW%2F2003+EURO+SIW+Papers+and+Presentations&file=03E-SIW-093.doc>
- [36] **Dahman, J., Fujimoto, R. and Weatherly, R.**, 1998, The DoD High Level Architecture: An Update, *Proceedings of 1998 Winter Simulation Conference*
- [37] **DMSO**, 1998, Defense Modelling and Simulation Office High-Level Architecture Rules Version 1.3
- [38] **Kuhl, F., Weatherly R. and Dahmann J.**, 2000, Creating Computer Simulation Systems: An Introduction to the High Level Architecture, pp.1, USA

- [39] **Scrudder, R. and Lutz R.**, 1998, Automation of the HLA Federation Development and Execution Process, Simulation Interoperability Workshop
http://www.sisostds.org/doclib/doclib.cfm?SISO_FID_1748
- [40] **IEEE Std 1516.3-2003**, 2003, IEEE Recommended Practice High Level Architecture (HLA) Federation Development and Execution Process (FEDEP), The Institute of Electrical and Electronics Engineers, Inc. New York, USA
- [41] **Ford, K. and Peyronnet, P.**, 2001, The Euclid RTP 11.13 Synthetic Environment Development & Exploitation Process, *Simulation Interoperability Workshop*,
http://www.sisostds.org/doclib/download.cfm?r_id=237443&Filename=01F-SIW-124_KeithFord_03.doc
- [42] **Ford, K.**, 2001, The Euclid RTP 11.13 Realising the Potential of Networked Simulation in Europe, *EuroSIW (Simulation Interoperability Workshop)*,
http://www.sisostds.org/doclib/doclib.cfm?SISO_FID_7237
- [43] **Peyronnet, P., Ancker, G., De Angelis, P., Batini, C., Christensen, A., Eidesen, D., Ford, K., Janssen, H., Hartmann, A., Holla, K., Marçal, J., zu Drewer, P. M., Öztemel, E., Öztürk, V., Öztürk, S and Adlem, S.**, 2001, SEDEP version 1.0, **RTP11.13 – TT&S SA – WE1.5 – volume 2, RTP 11.13 Technical Report**
- [44] **Hines, D. and Thomen, D.**, 1997, STEP 101: A Discussion of DoD's Simulation, Test and Evaluation Process (STEP), *1997 Spring SIW (Simulation Interoperability Workshop)*
- [45] **Öztemel, E., Öztürk, V., Soyer, B. S., Öztürk, S., Detsis, G., Gonidakis, V., zu Drewer, P. M. and Greiwe, K.**, 2001, AI Support for SEDEP and Requirements for intelligent tool support, *RTP 11.13 Project Technical Report*
- [46] **Abel, W., Brady, M., Dubowsky, S., Dunne, M. J., Eastwood, A., Gerhardt, L., Grossman, D., Hukkala, R., Kanal, V., Lehnert, W., Rosen, C. and Schweizer, F.**, 1983, Applications Of Robotics And Artificial Intelligence To Reduce Risk And Improve Effectiveness, National Academy Press, Washington D.C., USA
- [47] **Turban, E.**, 1992, Expert Systems and Applied Artificial Intelligence, Macmillan Publishing Company, USA
- [48] **Schnupp, P. H.**, 2000, Expert Systems in Prolog, Amzi Inc.
- [49] **Kandel, A.**, 1991, Fuzzy Expert Systems, pp. 34, CRC Press, CRC Press LLC
- [50] **Turban, E.**, 1995, Decision Support and Expert Systems, Prentice Hall, USA
- [51] **Krishnamoorthy, C.S. and Rajeev, S.**, 1996, Artificial Intelligence and Expert Systems for Engineers, pp. 34, CRC Press, CRC Press LLC
- [52] **Zadeh, L.A.**, 1999, Computers and mathematics with applications, fuzzy logic and the calculi of fuzzy rules, Fuzzy Graphs, and Fuzzy Probabilities, Vol. 37, Elsevier Science Ltd.

- [53] **Rao, V. B.**, 1995, C++ Neural Networks and Fuzzy Logic, M&T Books, IDG Books Worldwide, Inc
- [54] **Konar, A.**, 2000, Artificial Intelligence and Soft Computing Behavioral and Cognitive Modeling of the Human Brain, CRC Press, USA
- [55] **Lakhmi, C. J. and Martin, N. M.**, 1998, Fusion of Neural Networks, Fuzzy Systems and Genetic Algorithms: Industrial Applications, CRC Press LLC
- [56] **Liebowitz, J.**, 1997, The Handbook of APPLIED EXPERT SYSTEMS, CRC Press LLC
- [57] **Haykin, S.**, 1998, Neural Networks A Comprehensive Foundation Second Edition, Prentice Hall
- [58] **Principe, J.C.**, 2000, Artificial Neural Networks, The Electrical Engineering Handbook, CRC Press LLC
- [59] **Weiss, G.**, 1999, Multiagent Systems A Modern Approach to Distributed Modern Approach to Artificial Intelligence, The MIT Press, London, England
- [60] **Green, S., Hurst, L., Brenda, N., Cunningham, P., Somers, F. and Evans, R.**, 1997, Software Agents: A review, http://www.cs.tcd.ie/research_groups/aig/iag/toplevel2.html
- [61] **Caglayan, A. and Harrison, C.**, 1997, Agent Sourcebook, Wiley Computer Publishing, John Wiley & Sons, Inc.
- [62] **Arend, J. and Jansen, R.**, 2000, Individual and team performance assessment in networked simulation training, <http://www.nlr.nl/public/hosted-sites/simultaan/simultaan.pdf>, Netherlands
- [63] **Öztemel, E., Gürbüz, A. and Görçin, A.**, 2003a, In-Flight Demonstration of Embedded Simulation for Training Purposes On-Board Fighter Aircraft, Technical Report, *TÜBİTAK-MAM Library*, Turkey
- [64] **Öztemel, E., Öztürk, V., Soyer, B. S., Öztürk, S. and Gonidakis, V.**, 2003b, Inference Engines Specifications, *RTP 11.13 Project Technical Report*
- [65] **Rollesbroich, B., Öztürk, V., Öztemel, E., Soyer, B. S., Öztürk, S. and Gonidakis, V.**, 2003b, Knowledge Base Description. Technical Document, *RTP11.13-CAE-WE7.3-TR-2.1, RTP 11.13 Project Technical Report*
- [66] **ESTA Website**, Expert System Shell for Text Animation, <http://www.visual-prolog.com/esta/pdcindex.htm>
- [67] **Öztürk, V., Sönmez, C. and Öztemel, E.**, 2004, An AI based Synthetic Environment Evaluation Process, *The 3rd Asia Pacific International Symposium on Information Technology*, İstanbul, Türkiye
- [68] **Dominick, W. D.**, 1987, A Performance Measurement And Evaluation Environment For Information Systems, *Information Processing & Management*, **23(1)**, 7-15, Great Britain
- [69] **Haines, S., Longshaw, T., Sleep, R. and Kennaway, R.**, 1998, An open architecture approach to AAR, *1998 Spring SIW (Simulation*

- [70] **Rambhia, A. M.**, 2002, XML Distributed Systems Design, Sams Publishing
- [71] **Calzarossa, M. and Marie, R.**, 1998, Tools for performance evaluation, *Performance Evaluation An International Journal*, **33**, 1-3, Elsevier Publications
- [72] **Oser, R.L., Gualtieri, J.W., Cannon-Bowers, J.A. and Salas, E.**, 1999, Training team problem solving skills: an event-based approach, *Computers in Human Behavior*, **15**, 441-462, Pergamon, Elsevier Science Ltd
- [73] **Tidhar, G., Heinze, C., Goss, S., Murray, G., Appla, D. and Lloyd, I.**, 1999, American Association for Artificial Intelligence, www.aaai.org
- [74] **Lucas, P. and Van Der Gaag, L.**, 1991, Principles of Expert Systems, Addison-Wesley Publishing Company, Great Britain
- [75] **Jackson, P.**, 1990, Introduction to Expert Systems, Second Edition, pp. 356, Addison-Wesley Publishing Company, England
- [76] **Luger, G. F. and Stubblefield, W.A.**, 1989, Artificial Intelligence and the Design of Expert Systems, Benjamin Cummings Publishing Com. Inc.
- [77] **Stytz, M. R. and Banks, S. B.**, 1999, Enabling Reasoning System Migration for Computer Generated Actors using XML, *Simtect 99 Proceedings*
- [78] **Rollesbroich, B.**, 2003, Evaluation Objectives & Definitions for Nordic Fox, EUCLID 11.13 Project Memo
- [79] **Şen, Z.**, 2004, Mühendislikte Bulanık Mantık ile Modelleme Prensipleri, pp. 33, Su Vakfi, 2nd Edition, İstanbul (in Turkish)
- [80] **Ruskova, N.A.**, 2002, Decision support system for human resources appraisal and selection, *First International IEEE Symposium on Intelligent Systems, Proceedings*, **1**, 10-12 Sept., 354 – 357
- [81] **Drewer, P. M.**, 2001, Working Document to Technical report WE 6.1, RTP 11.13 Project Technical report, **RTP11.13 - CAE - WE6.1–WD1_MzD - 2.0b**
- [82] **Gokturk, E. and Polat, F.**, 2003, Implementing Agent Communication for a Multiagent Simulation Infrastructure on HLA, *Proc. of the International Symposium on Computer and Information Science (ISCIS 2003)*, LNCS, Springer-Verlag
- [83] **Sycara, K. P.**, 1998, Multiagent Systems, *AI Magazine*, American Association for Artificial Intelligence
- [84] **Öztürk, V. and Sönmez, C.**, 2004, An Expert-Fuzzy Approach for Evaluation, *IMS'2004: 4th International Symposium On Intelligent Manufacturing Systems*, Sakarya, Türkiye

APPENDIX A USE OF INES

A.1 INSTALLATION INSTRUCTIONS

Perform the following actions in order to install the INES:

- Run setup.exe file in the INES_SW folder and follow the instructions until the installation is completed. It is highly recommended to use the default values. Restart the computer.
- Click on the program folders, then click INES folder.
- Click INES for running Inference Engine and evatkb for running KB Editor. (or run evat.exe for inference engine and evatkb.exe for KB Editor from the folder that the program was installed))

Note that, the INES Inference Engine and Knowledge Base Editor can be best viewed using 1024*768 pixels screen resolution.

A.2 INFERENCE ENGINE

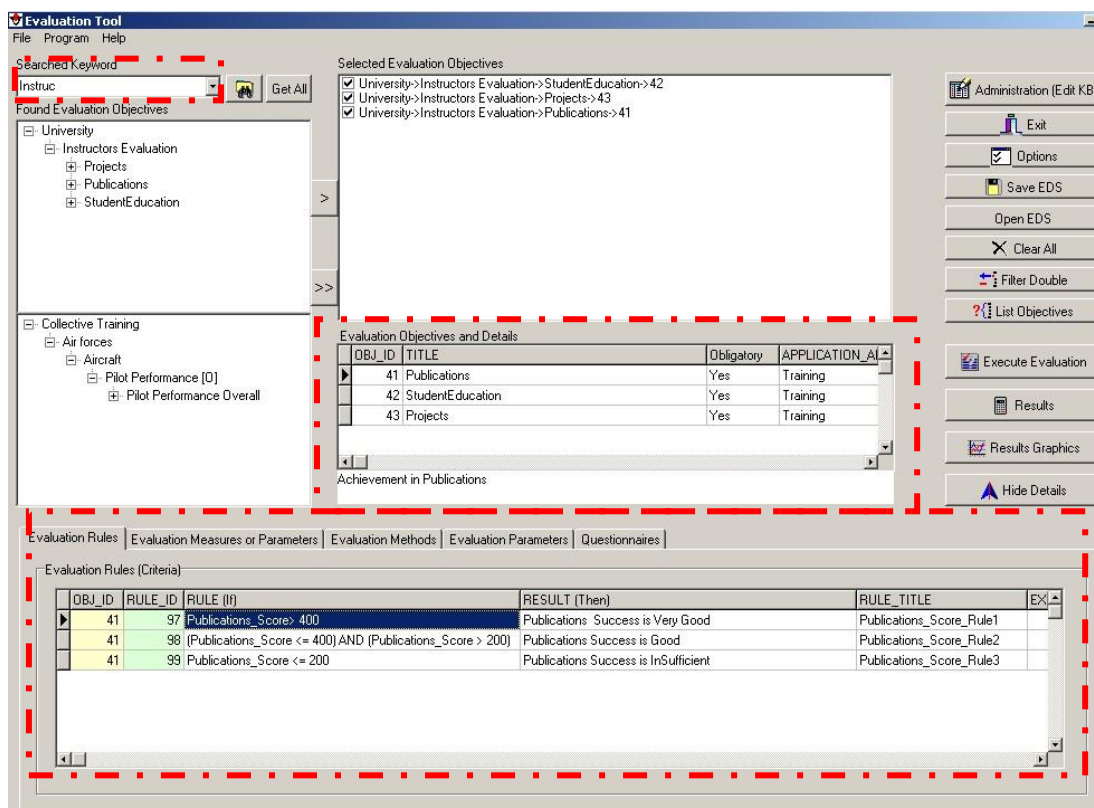


Figure A.1 INES Inference Engine screenshot

High-level evaluation objectives (Collective training, etc.) contain/consist of sub-Objectives (pilot performance,...).

The user enters keyword(s) in the “Searched Keyword” combobox (e.g. pilot, performance, etc) and presses “**Search KB**” or “**Enter**” button in order to find the related objectives from Evaluation Knowledge Base (KB). INES generates a hierarchy of objectives including the searched keyword(s) from Evaluation Objective tree (the above treeview) and evaluation database (the bottom treeview) as shown in Figure A.1. “Searched Keyword” combobox also stores the searched keywords for the current session and completes searched words according to the user text entrance.

The user can get information about the sub-objective by clicking sub-objectives in the treeview in order to be sure about his selections.

The user drags and drops or presses one of the “ > “ (only for selected and sub-objectives if any) or “ >> “ (for all) buttons to transfer the evaluation objectives from the generated objective list to the “Selected Evaluation Objectives” check list.

The user can then select/de-select sub-Objectives. When an objective from the Objective treeview box is selected and transferred to the Selected Evaluation Objectives editbox, all the dependent sub-Objectives, is listed in the Selected Evaluation Objectives edit box. It is possible to select or deselect from editbox. Also it is possible to delete items by pressing “Del” key on the keyboard from selected evaluation objectives. The explanations of buttons are given below.

“**Administration (Edit KB)**”: This button activates “INES KB Editor”. If an instance of “INES KB Editor is active”, this button won’t activate INES KB Editor and warn the user with a message ‘ "INES KB Editor" is open’.

“**Get All**”: This button loads all the evaluation objectives in the KB to the “Found Evaluation Objectives” treeview.

“**Save EDS**”: This button lets user to save the current work.

“**Open EDS**”: This button lets user to load the saved previous work.

“**Clear All**”: This button clears all Selected Evaluation Objectives from “Selected Evaluation Objectives” list box.

Clicking “Del”: This deletes Selected Evaluation Objectives from “Selected Evaluation Objectives” list box.

“**Filter double**”: This button filters the double Selected Evaluation Objectives from “Selected Evaluation Objectives” list box.

“**List Objectives**”: This button list detailed information of “Selected Evaluation Objectives” list box.

“**Show Details**”: This button shows the detailed information (criteria (rules), methods, measures) about selected objectives.

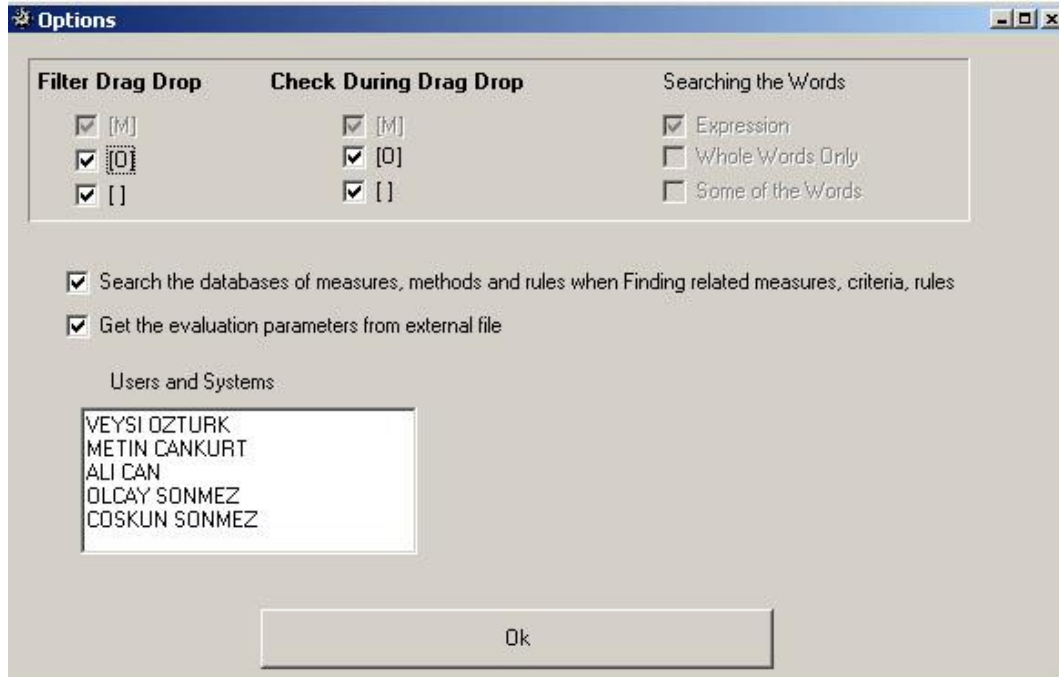


Figure A.2 Options form

“Hide Details”: This button closes the detailed information (criteria (rules), methods, measures) about selected objectives.

“Refresh DB”: Click Programs Menu and then “Refresh DB” sub-menu to refresh database fields (this may be necessary when you do changes in INES KB).

“Options”: This button activates options form.

The user may change the options of drag drop and other filtering capabilities from options form. Figure A.2 illustrates the options form. The following notation is used in the options form.

[M] : mandatory

[O] : optional

[] : Not specified

Note that the user must write next to evaluation objectives about the state of objective such as “Launch success [M]” for Mandatory sub-objectives, “Fuel [O]” for optional objectives and “Finalizing the Mission” for not specified objectives.

By being able to define his options, the user is more flexible in selecting evaluation objectives.

“Generate Results File (XML)”: This button generates EvaluationResults.xml which includes evaluation Results for the active scenario.

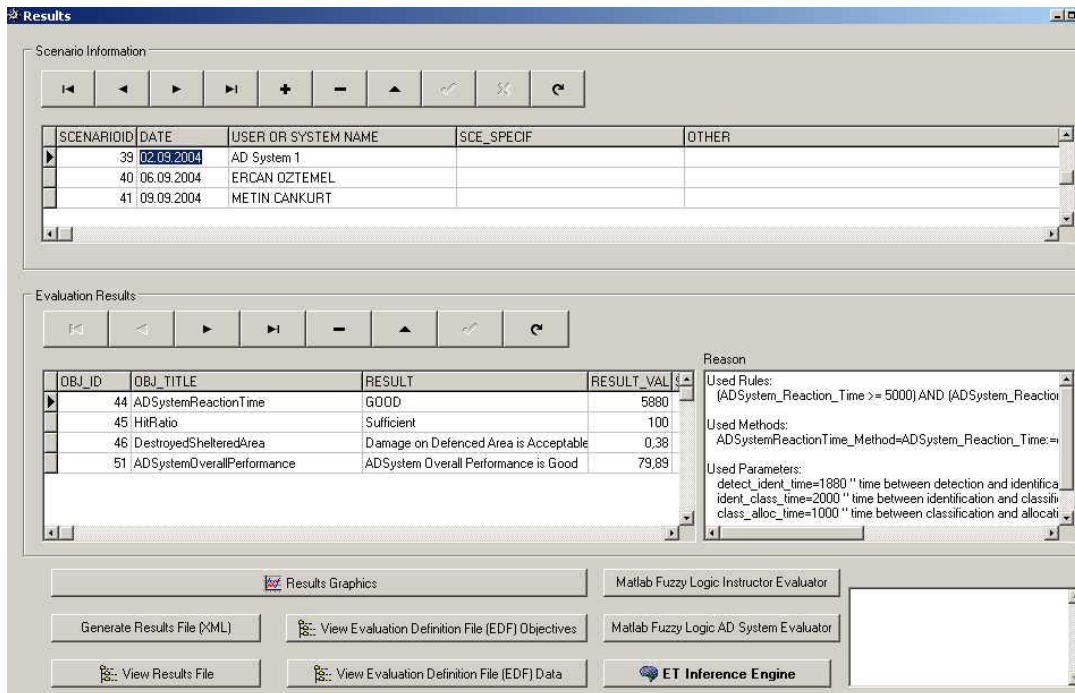


Figure A.3 Results form

“View Evaluation Definition File (EDF) Objectives”: This button views EvaluationObjectives.xml file.

“View Evaluation Definition File (EDF) Data”: This button views EvaluationDefinition.xml file.

“Matlab Fuzzy Logic Instructor Evaluation”: This button runs Matlab Fuzzy Logic Toolbox and sends required parameters for instructor evaluation and shows the result of overall evaluation as shown in Figure 3.27.

“Matlab Fuzzy Logic AD System Evaluation”: This button runs Matlab Fuzzy Logic Toolbox and sends required parameters for AD System Evaluation and shows the result of overall evaluation as shown in Figure 3.17.

“Result Graphics”: This button views Evaluation Results form as shown in Figure A.4. Choose or leave empty “User/System”, “Objective” combo-boxes and Date interval in order to run results query (press Results Graphics).

“Result Graphics (Run Query)”: This button runs finds the related data and updates graphics according the selections of USER/SYSTEM, Objective and date interval.

“Exit”: This button closes INES. Before exit, INES asks the user if he wants to save the work done so far.

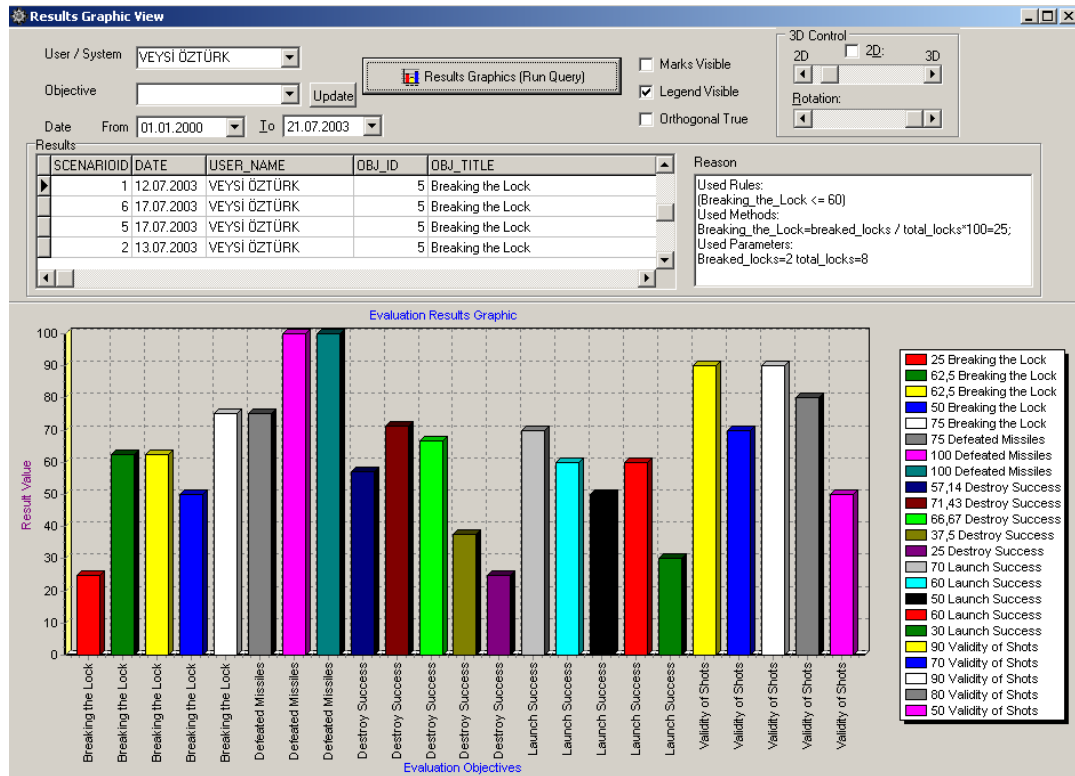


Figure A.4 Evaluation results graphic form

A.3 KNOWLEDGE BASE (KB) EDITOR

The INES Knowledge Base includes the evaluation objectives and database tables including “objectives”, “rules”, “methods”, “parameters”, and “questionnaires” tables. Each evaluation objective has related measures, methods, rules and parameters.

The main window of the INES Knowledge Editor is shown in Figure A.5. As shown in the Figure, the INES Knowledge Editor has two main parts:

- Evaluation Objectives tree
- Evaluation Database

These parts are explained in detail in the following section. The explanation of table fields is given in Appendix B. The related table fields are in the same color as shown in Figure 3.2. The relations of the fields are shown with black arrows in Figure 3.2.

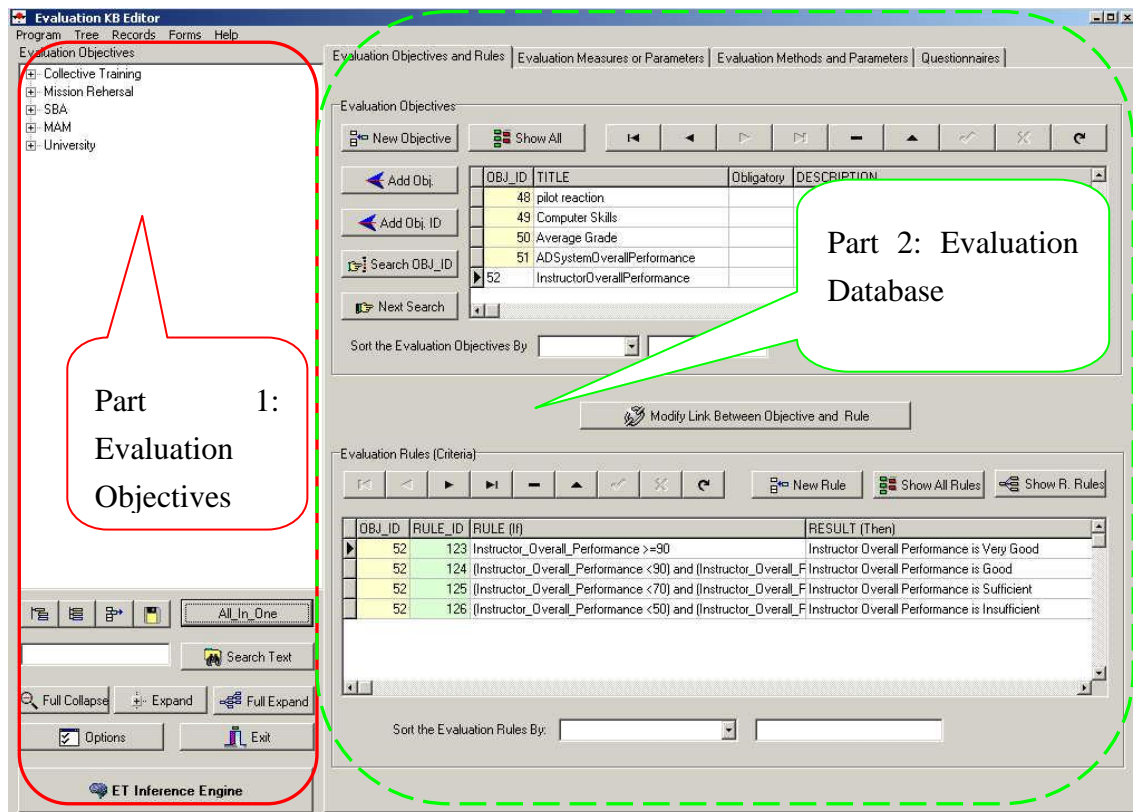


Figure A.5 KB Editor main window

Objective Tree Part

This part shows the hierarchy of objectives and the relation with the evaluation database. This tree can be updated, edited and the branches of the tree can be deleted.

There are mainly 8 buttons to be used in manipulating the evaluation objectives as shown in Figure A.6.

These are:

Add sub Item: Adds a new sub item into selected branch.

Add Item: Adds a new item in hierarchical evaluation objectives.

Delete Item: Deletes an item or a sub item from the selected branch.

Save KB: Saves the changes in hierarchical evaluation objectives.

Full Collapse: Collapses the entire tree.

Full Expand: Expands the entire tree.

Expand: Expands the selected branch of the objective tree.

Figure A.6 Buttons for updating the hierarchical evaluation objectives

The menu items to be used in manipulating the evaluation objectives are as follows:

MoveToTop: Moves the active branch to the top of the tree.

Save Model: Saves the evaluation objectives tree to a user defined file. With this facility, the user can define a new evaluation objectives model or modify an exist one easily.

Open Model: Opens the evaluation objectives tree from a user defined file.

Also note that, for renaming the branches, respective branch should be selected and right mouse button should be clicked once to allow the user to edit a new name or make required changes.

The place of the branches can be easily changed by drag and drop facilities provided. If the user presses “**Ctrl**” key during drag and drop, the selected branch will be **copied** to the destination branch.

The integer numbers at the bottom of tree branches show the ID of the related evaluation objective.

If the user double clicks any item in the evaluation objective tree, then only related evaluation objectives, measures, rules, methods, parameters and questionnaires will be listed on the right part.

Manipulating the objectives

For updating the objectives part of KB Editor, use the objective panel as shown in. The buttons are explained below.

New Objective: Inserts a new objective after the last record. Each objective has a unique identifier called as OBJ_ID. Once this button is clicked, a new OBJ_ID is generated automatically (It is important to use generated OBJ_ID for the data integrity) and then the user is allowed to enter his objective details such as title, state, application area, military application and keywords.

Add Obj.: Inserts the active objective title and ID to the selected part in the evaluation objective tree. In order to do that first the required branch of the objective tree must be selected and then the objectives must be typed in and following this “Add Obj.” button must be clicked. Note that the objectives, which were already typed in but linked to the objective tree, could be added later on.

Add Obj. ID: Inserts the active objective ID to the selected part in the evaluation objective tree. This button works similarly to the “Add Obj.” button as explained above. This button adds only the active objective ID to the tree. This prevents the user to type or enter the same objective, which is applicable to more than one branch of the tree. By using the ID, the inference engine can handle the same objective for different purposes.

Search OBJ_ID: Searches the active record OBJ_ID in the hierarchical evaluation

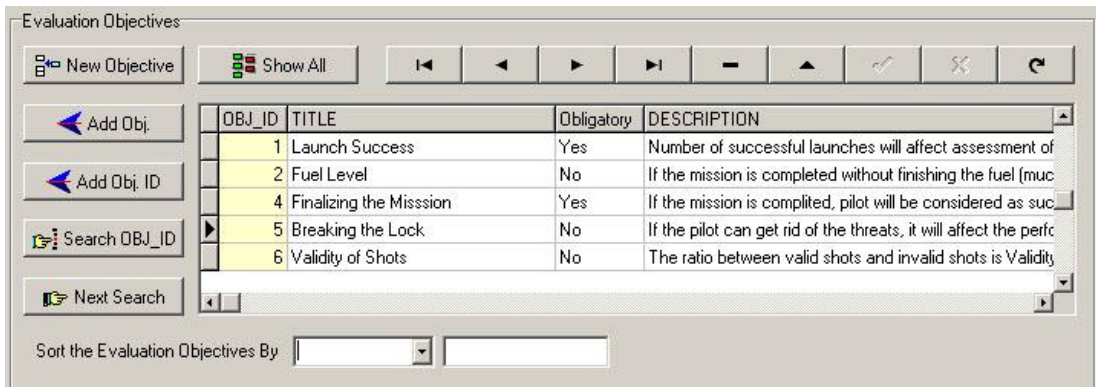


Figure A.7 Evaluation objectives

objectives tree. This allows the user to locate the objective inside the tree.











Next Search: Searches for the next repetition of the current record OBJ_ID in the hierarchical evaluation objectives tree.



Figure A.8 Database navigator bar

The Database Navigator Bars in the program has the same functionalities. Only the number of buttons in the Database Navigator Bars is different and each of them is related to different database tables.

The individual components of Database Navigator Bar () are explained below:

First 	Sets the current record to the first record in the dataset, disables the First and Prior buttons, and enables the Next and last buttons if they are disabled.
Prior 	Sets the current record to the previous record and enables the Last and Next buttons if they are disabled.
Next 	Sets the current record to the next record and enables the First and Prior buttons if they are disabled.
Last 	Sets the current record to the last record in the dataset, disables the Last and Next buttons, and enables the First and Prior buttons if they are disabled.
Insert 	Inserts a new record before the current record, and sets the dataset into Insert and Edit states.
Delete 	Deletes the current record and makes the next record the current record.
Edit 	Puts the dataset into Edit state so that the current record can be modified.
Post 	Writes changes in the current record to the database.
Cancel 	Cancel edits to the current record, restores the record display to its condition prior to editing, and turns off Insert and Edit states if they are active.
Refresh 	Refreshes the buffered data in the associated dataset.

A.4 Evaluation Database Part

This part is used to store the evaluation definition data (“objectives”, “rules”, “measures”, “methods” and “questionnaires” tables).

Sorting and searching the objectives

This section of the Knowledge Base Editor is designed to allow the user to find out the objectives easily. There are several options including;

“Sort the Evaluation Objectives By” identify the sort type (OBJ_ID or Recorded) (see). Modifying search edit box finds the nearest OBJ_ID from the objectives.

Pressing “Show All” button cancels the filtering and shows all the recorded data in the evaluation “objectives” table. The user may find the related OBJ_ID using the “Search OBJ_ID” button in the hierarchical evaluation objectives tree.



Figure A.9 Sorting, searching and displaying all records of evaluation objectives

Manipulating the evaluation rules

indicates the information related to the evaluation rules. The user can handle these information and make necessary changes whenever needed using the navigation bar as explained above. The following can be performed using the Knowledge Editor.

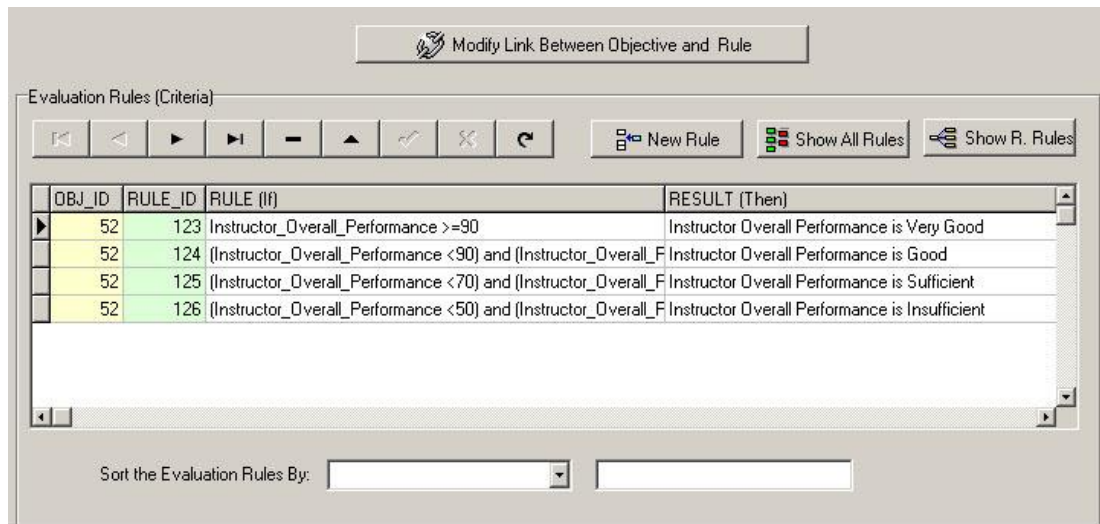


Figure A.10 Evaluation rules

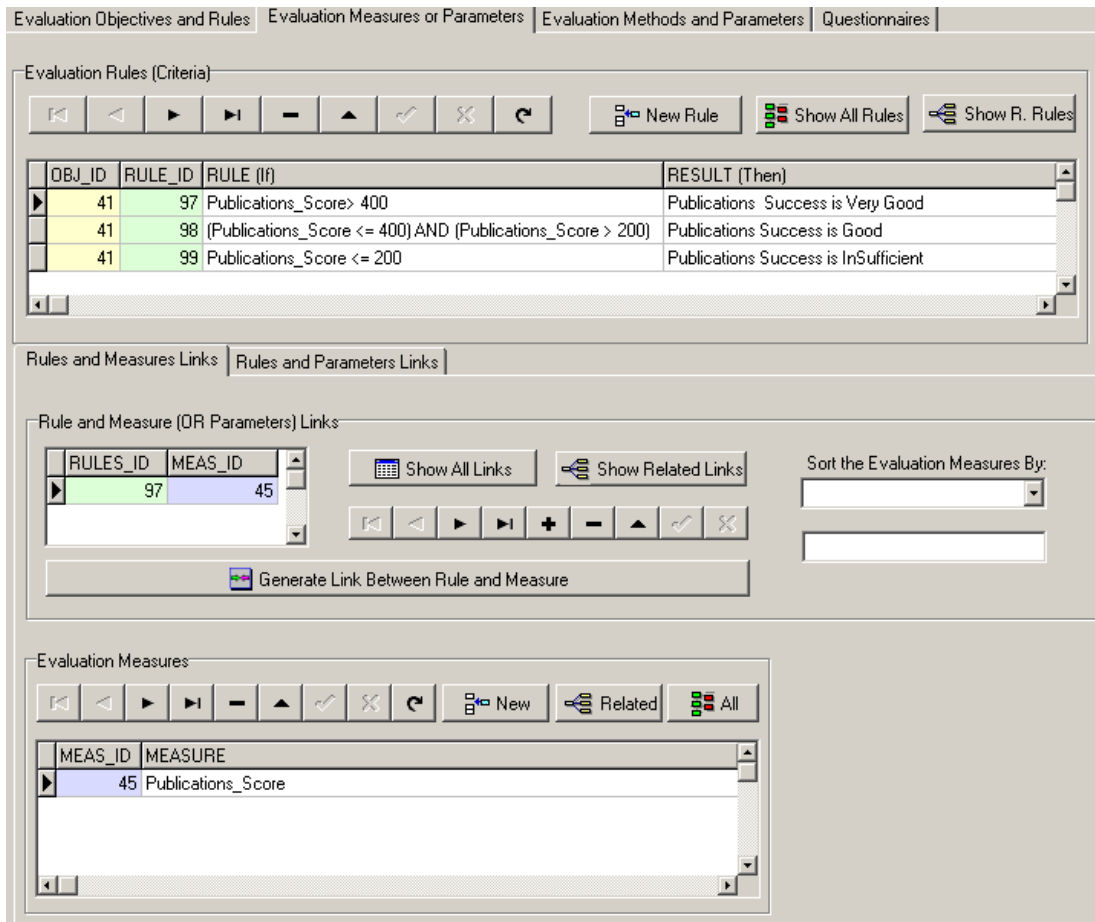


Figure A.11 Evaluation measures

New Rule: Inserts a new rule after the last record. Each rule has a unique identifier called as RULE_ID. Once this button is clicked, a new RULE_ID is generated automatically (It is important to use generated RULE_ID for the data integrity) and the user is allowed to enter his objective details such as title, rule, result and explanation.

Show All Rules: Shows all records in the “rules” table.

Show Related Rules: Shows related rules to the current evaluation objective in the “objectives” table.

Manipulating the evaluation measures

indicates the information related to the evaluation measures. Note that there must be at least one measure for each rule specified. The user can handle these information and make necessary changes whenever needed using the navigation bar as explained above. The following can be performed using the Knowledge Editor.

New Measure: Inserts a new measure after the last record. Each measure has a unique identifier called as MEAS_ID. Once this button is clicked, a new MEAS_ID is generated automatically (It is important to use generated MEAS_ID for the data

integrity) and the user is allowed to enter his measure details such as title, unit, precision, etc.

Show: Shows all records in the “measures” table.

Related: Shows related measures to the current record in the “rules” table.

Generate Link Between Rule and Measure: Generates link between the current rule and current measure. The measures used in rules are stored in the table “measures” and linked to the rules in table “MeasureLinks”. RULES_ID and MEAS_ID in the “MeasureLinks” table are generated automatically. The advantage of using a separate table to store linking information is that a measure is only need to be defined once, but can be used in several rules. So the user does not need to define the same measure over and over again.

Manipulating the evaluation methods.

indicates the information related to the evaluation methods. Note that there must be at least one method for each measure specified. The user can handle these information and make necessary changes whenever needed using the navigation bar as explained above. The following can be performed using the Knowledge Editor.

New Method: Inserts a new method after the last record. MEAS_ID is generated automatically (It is important to use generated MEAS_ID for data integrity).

Each measure has a unique identifier called as MEAS_ID. Once this button is clicked, a new MEAS_ID is generated automatically (It is important to use generated MEAS_ID for the data integrity) and the user is allowed to enter his measure details such as title, unit, precision, etc.

All Methods: Shows all records in the “methods” table.

Related Methods: Shows related methods to the current record in the “measures” table.

New Parameter: Inserts a new parameter after the last record. PARA_ID is generated automatically (It is important to use generated PARA_ID for data integrity).

All Prms: Shows all records in the “parameters” table.

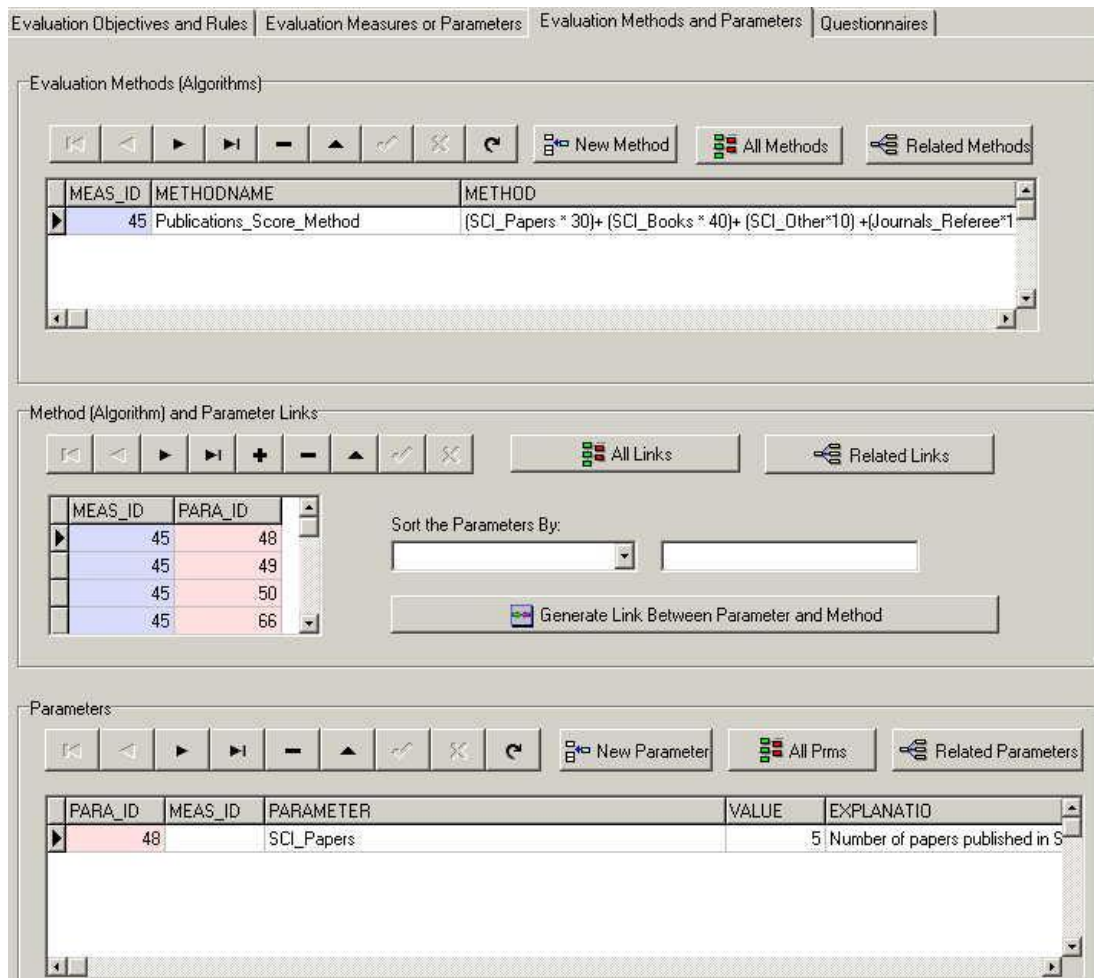


Figure A.12 Evaluation methods and parameters

Related Parameters: Shows related parameters to the current record in the “methods” table.

Generate Link Between Parameter and Method: Generates link between the current parameter and current method. In order to reuse parameters in different methods the linking information is stored in an additional table called “ParameterLinks”. MEAS_ID and PARA_ID in the “ParameterLinks” table are generated automatically. The parameters used in the measure have to be defined and linked to the specific method. The parameter only needs to be defined once, but can be used in several methods. The user does not need to define the same parameter over and over again.

Questionnaires

Clicking “Questionnaires” Button shows “Questionnaires and CheckLists” form as shown in This form can be manipulated using the navigation bars as explained above.

“Measures” table also represents questionnaires. The questions and predefined answers are stored in the tables “questions” and “answers”. A single question is identified by the MEAS_ID and the Q_ITEM_ID. Often a specific order of questions

is needed. Therefore the numbering of questions should always start with 1. MEAS_ID is an additional identifier in table “questions”. Also the number of predefined answers always starts with 1, so the MEAS_ID and Q_ITEM_ID are required in table “answers” to identify an answer unambiguously.

Evaluation Measures

MEAS_ID	MEAS_NAME	MEASURE	RESPONDENT
21		SARatingScale	Pilot

Questionnaires and CheckList Questions

MEAS_ID	Q_ITEM_ID	Q_TEXT
21	1	Was it possible to perform the task with respect to SA?
21	2	Was your SA on an acceptable level?
21	3	Was your SA on a satisfactory level?

Questionnaires and CheckList Answers

MEAS_ID	Q_ITEM_ID	ANSWER_ID	ANSWER
21	1	1	Yes
21	1	2	No - No SA at all. All important aspects were out of my control.
21	2	1	Yes
21	2	2	No - My SA was extremely low. Almost all important aspects were out of my control.
21	2	3	No - My SA was very, very low. Most important aspects were out of my control.

Main Form

Figure A.13 Questionnaires and checklists form

A.5 Options

Clicking “Options” button shows “Options” form, which will perform the following (see).

Only Double Click to see related Objectives: Check this, if you edit hierarchical evaluation objectives tree. If this option is unchecked, clicking any item in hierarchical evaluation objectives tree will list related database data on the right part.

Auto Collapse On: When clicking “Search OBJ_ID’ button, this option collapses the searched braches. If this option is unchecked, the searched braches will be expanded.

Figure A.14 Options form

“**Evaluation Objectives Fields Strings**” part lets user to modify or add default values of drop down menus in Evaluation Objectives table for the fields of Obligatory, Application Area, Military Application. This part also lets user to modify or add default values of User & System Names which is used in Results Form of IE.

“**Default File:** ET’s directory:” defines the default value of evaluation objectives file (e.g. data\treeref.egm) which saves the evaluation objectives in hierarchical manner.

“**Exit**”: button closes the INES KB. If any unsaved changes occur, INESKB will warn the user with a message “Evaluation Objectives KB has not saved. Do you want to save changes?”

A.6 The use of Knowledge Base Editor

There are various alternatives to store knowledge into the knowledge base using the Knowledge Base Editor. The following is sequence (order) of actions, which is recommended.

1. In Evaluation Objectives Part, click New Objective. Edit the record. Use the automatically generated OBJ_ID in hierarchical evaluation objectives tree to generate the relation between evaluation objectives tree and evaluation objectives table.
2. In Evaluation Rules Part, click New Rule. Edit the record. Rule_ID will be generated automatically.

3. In Evaluation Measures Part, click New Measure. Edit the record. Measure_ID will be generated automatically.
4. For generating the relationship between measures and Rules (editing “MeasureLinks” Table), click “Evaluation Measures or Parameters” part. Activate the related Measure. Click “Generate Link Between Rule and Measure” button. Rules_ID and Measure_ID will be generated automatically in “MeasureLinks” table.
5. For adding methods, click “New Method” button in “Evaluation Methods and Parameters” part. Edit the record. Measure_ID will be generated automatically in “Methods” table.
6. For generating the relationship between methods and Parameters, activate the related Parameter. Click “Generate Link Between Parameter and Method” button. Measure_ID and Parameter_ID (in “ParameterLinks” table) will be generated automatically.

“INES Inference Engine”: This button activates “INES Inference Engine”. If an instance of “INES Inference Engine is active, this button won’t activate INES Inference Engine and warn the user with a message "INES Inference Engine" is open.

APPENDIX B GLOSSARY

Artificial Intelligence	A branch of computer science that is concerned with the automation of intelligent behaviour
Evaluation	General term referring to the collection and processing of information and data in order to compare events, which have taken place (e.g. effects caused by a new technology) to a set of normative criteria or goals.
Evaluation methods	The algorithms used for analyzing the collected parameters.
Evaluation parameters	Indicate the type of data, their precision (if applicable), units (if applicable).
Evaluation rules	Criteria used to assess the collected parameters or calculated evaluation measures.
Evaluator	Evaluator is the person who is performing the evaluation
Expert Systems	Computerized advisory programs that attempt to imitate the reasoning processes and knowledge of experts in solving specific types of problems
Federate	Each simulation that is combined to form a federation
Federation	The combined simulation system developed from the constituent simulations
Fuzzy Logic	<p>Fuzzy Logic is a logical system, which aims at a formalization of approximate reasoning.</p> <p>In a wide sense, it is coextensive with fuzzy set theory. Fuzzy sets are sets in which members are presented as ordered pairs that include information on degree of membership.</p>
High Level Architecture (HLA)	Architecture for reuse and interoperation of simulations
Inference Engine	A computer program or code that handles the knowledge stored in the knowledge base and generates conclusions.
Knowledge Base	A data structure, which contains knowledge about the problem domain.
Subject Matter Expert (SME)	Individual who, by virtue of position, education, training, or experience, is expected to have greater than normal expertise or insight relative to a particular technical or operational

	discipline, system, or process.
Synthetic Environments	Internetted simulations that represent activities at a high level of realism from simulations of theaters of war to factories and manufacturing processes. These environments may be created within a single computer or a vast distributed network connected by local and wide area networks and augmented by super-realistic special effects and accurate behavioral models. They allow visualization of and immersion into the environment being simulated.
User	Any person performing evaluation of the synthetic environment, systems, humans
XML	A general-purpose, meta-markup language for documents containing structured information that supports the definition of customized markup components

CIRRICULUM VITAE

Veysi Öztürk is a senior researcher at Information Technologies Institute of TÜBİTAK (Science and Technical Researches Council of Turkey).

He graduated from Diyarbakır Anatolian High School. He received his B.Sc. degree in Electric and Electronic Engineering of Hacettepe, in 1996 and his M.Sc. degree in electronic engineering of Gazi University.

He is a co-author of six international commercial project reports, and six conference papers, which three of them is related with his PhD studies.

His research interests include artificial intelligence, expert systems, fuzzy logic, neural networks, software engineering, distributed simulation, and web based programming.

He has the following awards and achievements:

- Government scholarship examination, 3.th of Turkey (in ~75 000 students) of Turkey (Bursary Competition- ~75000 students entered)
- TÜBİTAK Mathematics Competition (1.st of Southeast Anatolian)
- TUBITAK (The Scientific and Technical Research Council of Turkey) Mathematics Olympiad (3.rd of Turkey)
- Student Selection Exam for University, 1.st of Diyarbakır (286.th of Turkey)
He was ranked in the top 1%, among 1 million examinees at the University Selection and Placement Test in TURKIYE (1991)
- His bibliography was published “Who’s Who in Science and Engineering” 2002 and 2003 editions.