EXPERT SYSTEM TESTING METHODOLOGY

BY

Dr.Khaled Shaalan

Dr.Abeer El_korany

August 2004

1 Introduction

The attempt to define life-cycle models of expert systems construction has led to the possibility of coupling different testing activities to the specific stages in the development. Thus, the suggested testing methodology is not standalone but parts of the development process of the expert system. This methodology evolves through a cycle of two main steps, namely knowledge base testing, and system testing. A complete testing cycle is performed in iterations through which, the expert system is updated and refined.

1.1 Motivations

Last version of the testing methodology suggests that testing is performed at the end of the development process, which make error correction very costly. From the central lab of agriculture expert system (CLAES) practical experience, it has been found that there is a strong need to harmonize the expert system development process with practical testing techniques. Thus, the proposed testing methodology suggests coupling different testing activities throughout the life cycle of KBSs development.

1.2 Component of testing methodology

This methodology evolves through a cycle of two main steps:

- Knowledge base testing
- System testing

As it is show in figure 1. , a complete testing cycle is an iterative process, through which the expert system is updated and refined. These steps are shared between two departments at CLAES: the Expert System Developing Department which is responsible for the development of the expert system and the Training, Evaluating and Updating Expert Systems Department which is responsible for training, evaluating and updating expert Systems. The following sections

describe the details of each of these steps as well as the duties of each departments.

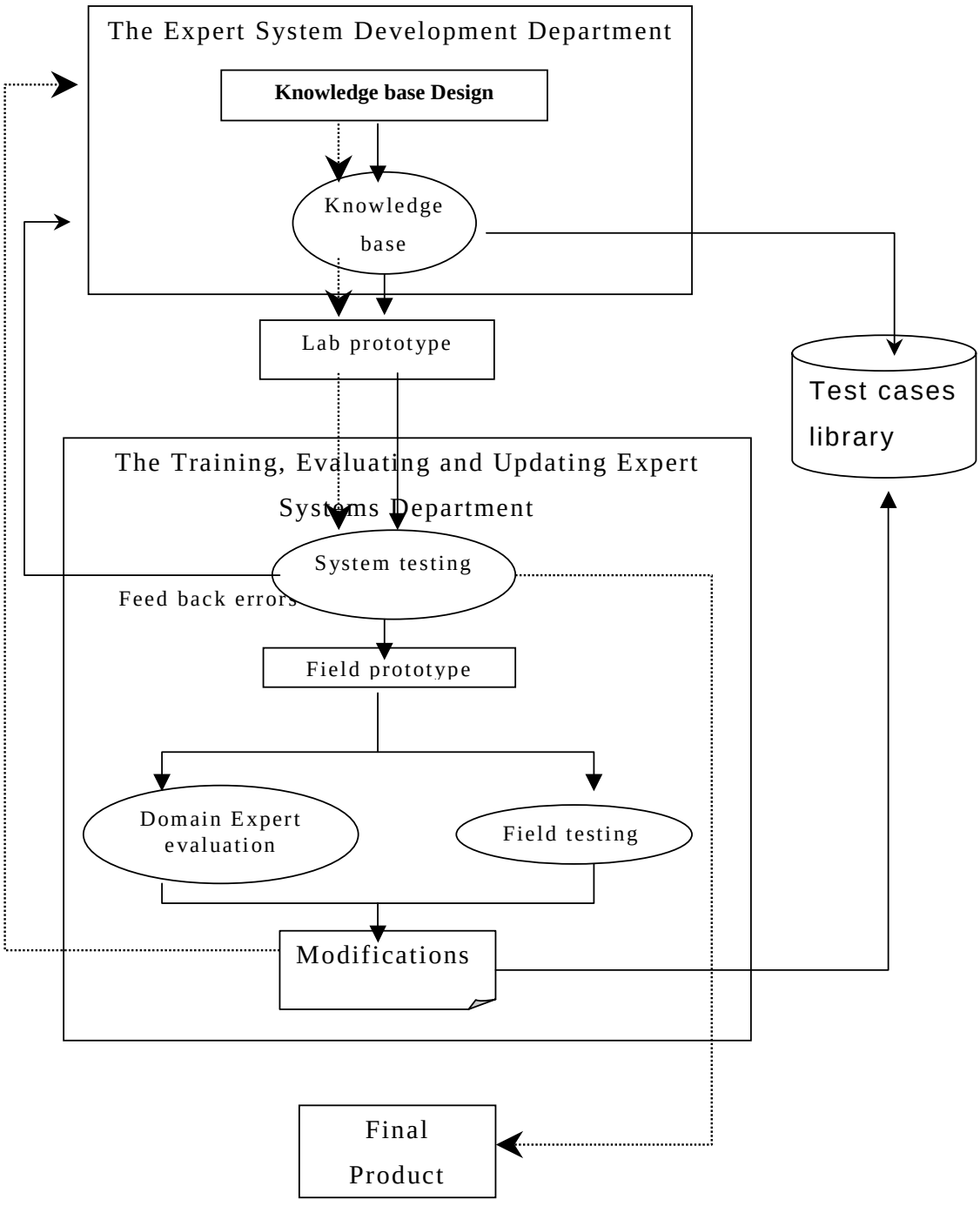

Figure1: The overall architecture of expert system

2 The Proposed testing methodology

2.1 Knowledge Base Testing

The aim of knowledge base testing is to find out whether the developed system is free from internal errors such as consistency and completeness as well as ensure that the knowledge meets the required specification. This can be accomplished through two complementary steps, namely: verification and validation.

2.2 System testing

The purpose of this test is to detect error that may arise during the implementation of the expert system. This test measures the effect of integration of knowledge base with the expert system shell. In CLAES, there are two expert system shells, MiniKROL and MiniKSR. The suggested testing activities are applicable for both of these shells. A major design goal of these tools is to support the CommonKADS (Breuker, J., 1994) knowledge engineering methodology that is used in expert system development.

3 Knowledge Base Testing

- **Responsibility**

This test is applied by the developer of the Expert System Developing Department. It is preferable to be applied by a knowledge engineer that was not involved in the design of the expert system.

3.1 Verification

The verification process of the KB ensures that the system is free from internal errors such as consistency and completeness. Verification does not involve the execution of KB, it examines internal correctness and consistency of the KB. This can be achieved through design walkthrough. The verification process involves elimination of errors that could be made in codifying the KB (such as insertion of duplicate or circular paths) i.e. the KB should be syntactically and semantically

correct. Distinction is made to verify consistency between different KB layers. The verification process can be distinguished into two main es

- Domain knowledge verification and,
- CommonKADS layer Verification.

3.1.1 Domain layer verification

Objective:

Detect the consistency and completeness errors of the domain knowledge.

Input:

The domain knowledge which contains all the knowledge about the application domain such as: concepts, properties, relation between concepts, rule clusters, tables, and mathematical functions.

Error type	Output	Method
Rule cluster errors	Detect undefined concept, undefined property, undefined property values.	Compare both the premise and action parts of each rule in a rule cluster with domain ontology
	Detect duplicate rule pairs, conflict rule pairs, and subsumed rule pairs.	Compare each rule with others in the same rule cluster
Table errors	Detect undefined concept, undefined property, undefined property values.	Compare each row of the table with domain ontology

	Detect duplicate table rows, and conflict table rows.	Compare each row with others in the same table
Function errors	• Detect undefined concept, undefined property. • Ensure that all input/output concept-property are of type integer or real	compared function body with domain ontology
Completeness Error	Detect unfirable domain relation	1. Get input parts of each domain relation 2. Get source of value of each concept/attribute pairs 3. If the source of value is derived then get the domain relation that produce this concept/attribute pair

	Detect unused consequence	1. Get the output part of each domain knowledge component 2. The output part should be either final goals, or used to fire other domain knowledge component

3.1.2 KADS Layers Verification

Objective:

Detect the consistency of inference knowledge

Input:

The domain, the inference and the task knowledge

Error type	Output	Method
Inference layer inconsistency error	Detect input/output role inconsistency error	1. Get the input role of each inference step. 2. The input role should be an output from a previous inference steps or user-database defined.
Inference/domain layer mismatching	Detect Inference/domain layer mismatching	1. Get defined domain relations of each inference step. 2. Find out whether these domain relations are defined in the domain layer 3. get input/output role of each inference step

		4. Compare the input role with the combination of the input property of the domain knowledge components which this inference uses
Task/Inference layer mismatching	Detect task/inference layer mismatching	1. Get defined inference layer of each sub-tasks 2. Find out whether these inference steps are defined in the inference layer.

*It is significant to mention that any modification required according to the verification process should be mapped to the design document.

3.2 Validation

The validation process finds out whether the KB is appropriate for the task domain for which it has been designed. Validation is concerned with the execution of the KB on a set of well-defined test cases to evaluate the functional, structural or computational aspects of the system. It checks whether the KB corresponds to the system it is supposed to represent. This can be considered a comparison of KBS behavior against the specification of intended behavior expressed as test cases. A general test case generation method can be described in three major steps:

- Test preparation (Test case generation)
- Test result analysis.

Test preparation (Test case generation)

Test cases are generated in the form of input/output concept-attribute pairs and their suggested values according to the design document of the expert system. A bottom-up test scenario that starts with testing all the fundamental components of the KB (domain knowledge), then inference knowledge and finally the complete system (task knowledge) is used. This prevents errors from propagating throughout the whole

KB and allows reusing test cases of one knowledge level in testing subsequent knowledge level.

Reviewing test cases

The domain experts review the generated set of test cases to approve its validity. Approved test cases are stored in a composite test case library to be used in the regression test. The knowledge engineer also use this list and compare it with the generic design report to check the consistency between the target system design versus the developed model. It is significant to mention that any modification required according to the validation process should be mapped to both the design and the implementation.

The following subsections present different methods used for test case generations.

3.2.1 Domain Knowledge Testing

3.2.1.1 Rule testing

Objectives

Generate test cases for each rule cluster

Input	Output	Method
Rule cluster	Test cases of rule cluster	1. Separate disjunctive rules. 2. Generate testing values. Test values for numeric attributes are generated according to table1. Nominal attributes are considered of type constant. 3. Combine testing values to generate test cases.

Operand	Operator	True
Const	=	Const
Const	! =	Const + step
Const	>	Const + step
Const	>=	Const
Const	<	Const – step
Const	<=	Const

Table1: The suggested testing values

3.2.1.2 Table testing

Objective:

Generate test cases for each table

Input	Output	Method
Table	Test cases of table	1- use the table header to identify the input concept-property pairs of table 2- Determine the used operator, if any for each table row, and accordingly generate the appropriate true value according to the table1

3.2.1.3 Function testing

Objective:

Generate test cases for each function

Input	Output	Method
Function	Test cases of	1. Generate testing values of the

	function	function condition according to table1. 2. Generate testing values of the each concept attribute pairs of function body. Three different test cases are generated for each of these concept-property pairs: one random, and two for boundary condition. 3. Combining step1 and step2

3.2.2 Inference Knowledge Testing

Objective:

Generate test cases for each inference step

Input	Output	Method
inference step	Test cases of inference step	1. Get approved test cases of the inference step components such that one case of each rule in each rule cluster, one random case for a function, and one case for a table 2. Combine these cases

3.2.3 Task Knowledge Testing

Objective:

Generate test cases for each sub-task

Input	Output	Method
sub-task	Test cases of sub-task	1. Obtain the task input concept/property pairs.

		2. Reuse the stored test cases of all the inference steps of this task 3. Extract test cases inputs of a task from the generated test suite.

After applying the verification and validation test, the Expert System Developing Department ***should*** supply the Training, Evaluating and Updating Expert Systems Department with a correct version of both design and implementation after applying all required modifications.

4 System testing

- **Responsibility**

This test is applied by the Training, Evaluating and Updating Expert Systems Department.

- **Generated reports**

A Testing report that documents all testing comments is generated. This report is provided to the Expert System Developing Department to apply the required modifications on the design, implementation and approved test cases.

Objective:

Ensure that the complete expert system is running properly, by applying different tests that serves to examine the performance of the system in different situations.

Input:

The complete expert system.

Error type	Output	Method
System installation	System installation errors	Install the expert system, find any installation error
Start Up & Quit	Start Up & Quit errors	Run the expert system and find out if it work when exit and rerun it again for several times.
Usability	Usability Errors	Navigate through the expert system and find out the following: 1- Is it obvious to the user which actions are available to him 2- Is the expert system consistent from page to page, including font sizes

		and colors, adjust of displays, meaningful titles, meaningful buttons in name and actions, use the same language 3- If a user forgets to fill in a required field, is there a friendly error message and a change of the color of the field label to some other conspicuous color
Unit Testing	Functionality errors	Apply random test cases that contains the following: 1. Valid data 2. Invalid data (+ ve ,0,- ve) 3. Field type (by trying value of wrong types for numeric field try nominal and visa versa). 4. Range checking. (Field boundary)
Load Testing	Load errors	Apply load testing (simulation) for web-expert system
Robustness	Robustness Errors	Measure how the system react to any input also how the output degrade for input degradation by applying the following cases: 1. For numeric fields, choose the minimum and maximum numbers. 2. For nominal, multi-value fields, choose most of the possible

		values. (Try all of them then decrease to 90 % and so forth) 3. For nominal, single-value fields, choose the most exceptional value (e.g. unknown). 4. For date fields, try very near date and very far date.

5 Domain expert assessments

The system is evaluated in a classroom setting by observing the system in use and administrating questionnaires. This setting consists of the domain expert that were involved in the knowledge acquisition process as well as other domain expert that are involved in the evaluation process. Observers videotape and take notes to assess how both users and domain experts interact to the system. The expert assesses the correctness of the KB, the quality of the explanation, and the quality of the answers. The user assesses his/her ability to interface with the system, the timelines of the response, the reasonableness of the output and explanation, and how the system fits in with the operating environment. This evaluation is applied in parallel with the filed testing (see next section) and all suggested modifications are collected to be applied to the design, the implementation and test cases of the expert system in order to produce the final product.

6 Field testing

In general, field testing is important because it allows the developer to monitor the system in its actual user environment. The expert system is tested in a similar environment for short period (1 year) and obtains feedback on system effectiveness and user interface. Accordingly, appropriate changes are made to the system. The system is then run in parallel at different sites to the existing process in the intended operational environment. During the parallel test, assessments are made as how well the system is meeting its goals.

7 Evaluation

The main goal of the evaluation step is to assess the quality, usability, and utility of the expert system from the point of view of human experts other than the domain expert, and from the point of view of the intended users.

The basic idea of the adopted technique is to evaluate the behavior of the expert system, against that of human experts, by generating a collection of carefully selected test cases, and let a number of human experts in the domain - as well as the expert system - handle these test cases. Another human expert will evaluate the generated solutions, and rank them according to their grades. Later on, an open discussion will be held to let human experts justify their solutions. According to this discussion, evaluation of solutions may change, and the final ranking of solutions will be reached. If the expert system is far from precedence, the knowledge base must be updated. The evaluation methodology presented here is based on the technical report "Verification & Validation" of the laboratory prototype Expert system [TR-88-024-17], and Dr. Robert M. O'Keefe [TR-88-024-08].
The following is a detailed, step-by-step, evaluation process:

7.1 Prepare case description forms

As a primary step in the evaluation process, forms must be designed for test cases. These forms may vary according to the kind of knowledge to be tested.

7.2 Prepare comparison criteria:

Evaluation criteria, or a formula, must be designed, to enable a formal judgment on solutions generated by human experts, and the expert system. The selected criteria should provide both quantitative and qualitative evaluation basis for judgment. The following is an example for qualitative and quantitative evaluation criteria:

Grade	Approve.	Points
Excellent	E	3
Good	G	2
Acceptable	A	1
Unacceptable	U	0

$$P_i = 3*NE_i+2*NG_i+1*NA_{i+}0*NU_i$$

Where:

P_i the performance score for expert # i

NE_i number of cases evaluated as excellent

NG_i number of cases evaluated as good

NA_i number of cases evaluated as acceptable

NU_i number of cases evaluated as unacceptable

7.3 Generate test cases:

Test cases are to be prepared manually by knowledge engineers. The most import criteria of these test cases is that it must cover both normal cases, as well as the most difficult, rare cases.

7.4 Solving test cases:

A copy of the selected test cases will be given to three or four domain experts. The same cases will be introduced to the expert system. Each of the domain experts as well as the expert system will work out the test cases independently

7.5 Evaluation of test cases:

Solutions of test cases must be evaluated in a blind manner, so that distinguishing between solutions of the expert system and solutions of domain experts is not possible. One or two domain experts, other than those who gave the knowledge acquired by the Expert system - and of course other than those who solved test cases - will be given generated test cases for evaluation.Solutions will be evaluated according to the previously prepared formula.

7.6 The result

A score will be given to each solution, and solutions will be ranked according to these scores.

7.7 Observations and remarks:

A meeting will be held to discuss solutions. The domain expert who gave the knowledge acquired by the expert system, domain experts who solved the test cases, evaluators, and knowledge engineer should attend this meeting to analyze solutions and reach a final conclusion about the behavior of the expert system.

7.8 Updating knowledge and implementation:

According to the conclusions reached in the previous step, the knowledge base and implementation of the Expert system must be updated, so the system will become more robust and valid.

7.9 Documentation

A detailed evaluation report should be prepared at the end of evaluation process.

8 Conclusion and future work

This document describes the methodology required for testing expert systems at the central lab of agriculture expert system (CLAES). The methodology consists of two main steps: knowledge base testing, and system testing. These steps are shared between two departments (CLAES): the Expert System Developing Department which is responsible for the development of the expert systems and the Training, Evaluating and Updating Expert Systems Department which is responsible for training, evaluating and updating expert systems. During the knowledge base system step, the expert system is checked for its consistency, completeness and correctness. System testing aims to ensuring the reliability of the expert system before distribution. To ensure the correctness of the final system with respect to users' needs and requirements, a parallel test is done by both the domain expert and

field testing in different sites. Feed backs are obtained, analyzed and then applied by the Expert System Developing Department in the design document, implementation and stored test cases.

The testing methodology can enriched by a supporting tool set for each of the recommended testing criteria. In order to automate the testing process as indicated by the testing methodology, a set of tools is suggested. This will facilitate time and efforts spend during the test process. These tools are:

- An automatic verification tool (El-Korany, Shaalan, Baraka & Rafea, 1998).
- An automatic test case generation tools (El-Korany, Rafea, Baraka & Eid, 2000).

The maintenance process aims to keep the KB with the same quality level even after the development and testing processes. Thus, it is recommended to introduce a maintenance methodology to complement the testing methodology.

Refrences

Breuker, J. & Van de Velde, W. (1994). The CommonKADS Library for Expertise Modeling, IOS Press, Amsterdam, The Netherlands.

El-Korany, A., Rafea, A., Baraka, H., Eid, S. (2000). A Structured Testing Methodology for Knowledge-Based Systems, *In proceeding of DEXA international conference*, London.

El-Korany A., Shaalan K., Baraka H., & Rafea A.(1998). An Approach for Automating the verification of KADS-based Expert Systems, *New Review of Applied Expert Systems*, Vol. 4, pp. 107-124, Taylor Graham, UK.