A Proposed Methodology For Expert System Engineering

Yasser Abdelhamid, Hesham Hassan, Ahmed Rafea
Central Laboratory for Agricultural Expert Systems
{Yasser, Hesham, Rafea} @esic.claes.sci.eg

Abstract

The methodology presented in this paper is the result of accumulated experience gained through many years of developing several expert systems in the agricultural domain by the Central Laboratory for Agricultural Expert Systems (CLAES). We have adopted a spiral model for the overall life cycle of expert systems development. As will be explained in this paper, the development methodology of an expert system has two aspects: Knowledge engineering, and Software engineering. From the knowledge engineering aspect, we adopted the CommonKADS methodology, and model driven approach has been applied. From the software engineering aspect, there are four activities for expert system development: requirements specification, design, implementation, and testing. This paper will include a detailed specification of each of these activities. The internal organization of CLAES is planed to cope with the technical requirements for expert systems development. The workflow among participating teams is explained.

I INTRODUCTION

Expert systems development is a complex and expensive process that needs to be applied in an organized manner. Many approaches have been introduced for this purpose e.g. CommonKADS[1], Components of expertise[2], and Generic tasks [3], but theoretical approaches must be supported by practical guidelines in order to apply these methodologies in real life applications. The adopted methodology presented in this paper is based on a spiral model that guides the overall life cycle of expert systems development (Fig.1). According to this model, the development methodology consists of two main components: *Knowledge Engineering, and Software Engineering.* These two components are interacting with each other. In other words, they are not sequential in nature. Some phases of the software engineering methodology may be applied before the completion of the knowledge engineering part and vice versa.

IIKNOWLEDGE ENGINEERING METHODOLOGY

The adapted methodology include three main activities, that are directed in iterations as illustrated in (Fig.1), to produce successive versions of the expert system, starting from research prototype and ending by the production version. These activities are:

- Knowledge acquisition,
- Knowledge analysis & modeling, and
- Knowledge verification.

The rest of this section is a detailed description of the applied methodology for each of these activities.

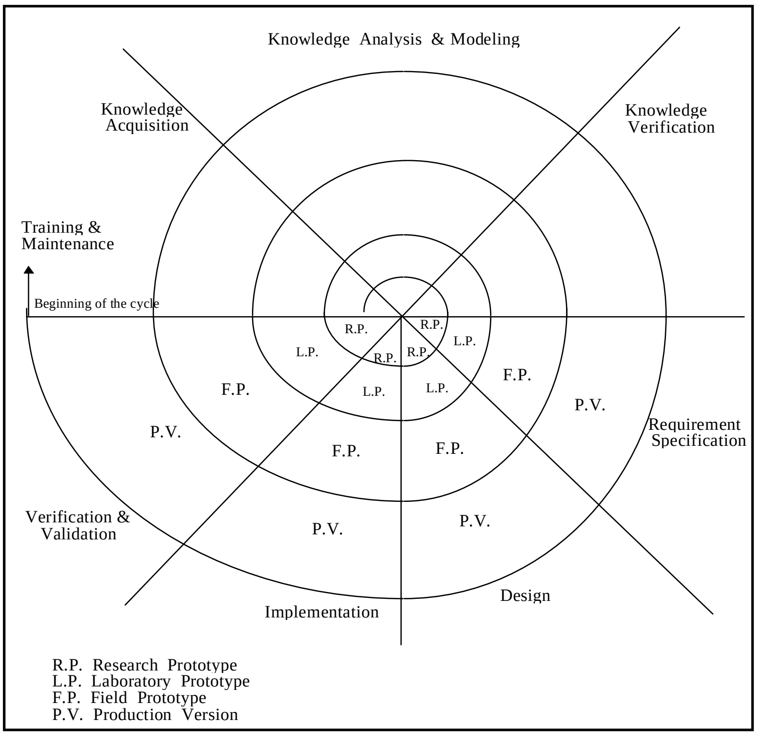


Fig.(1) Spiral Model for
Expert System Development

II-1 Knowledge Acquisition

Knowledge acquisition is considered the bottleneck of the expert system building process. One of the major difficulties at this stage is to explicitly identify and capture knowledge relevant to the intended application. The Central Lab for Agricultural Expert Systems (CLAES) has gained a considerable experience in building expert systems in the agricultural domain. Accordingly, CLAES has succeeded in building a set of interpretation models[1] (IM) for agricultural expert systems [6] that can be reused for building new applications. The developed models help in defining the set of domain models to be acquired from the domain expert, hence decrement unfruitful knowledge elicitation efforts, and direct the process in an organized manner. Accordingly, the task of a knowledge engineer is to select the appropriate model for the intended application, and determine the domain models required by each sub-task, and conduct knowledge elicitation sessions to acquire the required knowledge.

II-2 Knowledge Modeling Steps

We followed the *Select-and-Modify*[7] approach for expertise modeling, where a complete generic model is selected from a set of predefined models, and subsequently modified to suit the needs of the intended application, giving a complete customized expertise model, after additional domain knowledge acquisition. This modeling approach distinguished KADS-I [8] from other model driven approaches of the late 80's, like Generic tasks[3] and Role Limiting Methods [4], where the generic models were hardwired into tools, and could not be modified.

The Select-and-modify approach is divided into the following four activities:

1- **Select-IM**: Select an interpretation model according to a set of selection criteria (task features according to CommonKADS terminology). In our situation, we have developed a domain specific

---

[1] An interpretation model as defined by KADS methodology is a high level, skeletal structure that describes the reasoning method (inference and task structure).

library that contains models covering most of the agricultural problems, e.g., Irrigation, Fertilization, Diagnosis, Treatment, and Plant caring.

2- **Evaluate-IM**: Investigate, whether the selected IM is suitable for the application, or that it needs some modification. This activity is done by identifying the discrepancies between the required system behavior and that of the selected interpretation model . These discrepancies can be discovered either by walking through the IM or by trying hypothetical cases to evaluate the

3- **Modify-IM**: Modify the IM, to make it suitable for the intended application.

4- **Domain-KA:** Acquire the domain knowledge according to the selected, and probably modified interpretation model.

II-3 Knowledge Verification

Knowledge verification is the stage whereby we make quality assurance of the acquired knowledge. Actually there are two points of interest: review procedure, and multiple expert conflict resolving procedure.

II-3-1 Review Procedure

Establishing a review procedure at the knowledge acquisition stage reduces the efforts to be done later in the verification and validation of the developed system. Knowledge is reviewed at the end of different phases: knowledge elicitation, knowledge analysis and modeling, and implementation.

Reviewing at the *elicitation stage*, is conducted by letting the domain experts review the results of the knowledge elicitation sessions.

At the *analysis and modeling stage*, the domain experts review the filled forms describing the domain knowledge. Since task and inference knowledge are documented in KADS notation, which is hard to be understood by non specialists, the knowledge engineer performs this activity by walking through them with the presence of the domain experts.

Reviewing at the *implementation stage*, is conducted by letting the domain experts review any early prototype.

II-3-2 Multiple Experts Conflict Resolution

Multiple experts conflict resolution is considered as a way of verifying the acquired knowledge. Because when two experts give different knowledge for the same thing, then trying to resolve this conflict yields more reliable knowledge, hopefully, agreed upon by both of them. If no consensus is reached the expert who is recognized to be more specialized in the area of disagreement is considered.

III SOFTWARE ENGINEERING METHODOLOGY

As a software, building a knowledge based system entails doing software engineering activities that are accomplished in parallel with knowledge engineering activities that we have discussed in the previous section. These activities go through different stages, these stages are: requirements specification, design, implementation, and testing. As we mentioned before, these activities are done in successive iterations, each of which ends with the delivery of a new, more mature version of the expert system, until the production version is delivered.

The approach we are applying for software development is a combination of rapid prototyping, incremental, and traditional methods. Rapid prototyping is used first to reach an agreement on the initial set of the system requirements. A laboratory prototype then is incrementally developed and tested for maturity. The field prototype is then implemented by adding a better user interface, and more explanation facilities to the laboratory prototype, then the field prototype is tested in real environment. Once the system is successfully field tested, it can be considered as the final requirements specification of the production version. As we reach a complete requirements specification, the production version can be developed using traditional software engineering methods.

As we have described the general broad lines in our software methodology, the rest of this section is a more elaborate specification of each software engineering activity.

III-1 Requirements Specification

As shown in (Fig1), the outcome of the early knowledge elicitation activity is an initial set of requirements specifications. This initial set is the basis for further knowledge acquisition efforts and the basis for the research prototype preliminary design. The requirements specifications is revised once the research prototype is evolved. This version of the specification is the basis for developing the lab. prototype. At the end of the lab. prototype implementation and testing, the requirements specification document is revised again if necessary. This second revision is the basis for developing the field prototype. The final requirements specification document is then produced after the implementation and testing of the field prototype. This final document is the basis for the production version of the expert system to be developed.

III-2 Design

A preliminary design is done just after the set of initial requirements specification is determined, and a preliminary model of knowledge layers is specified. This design is the basis for the research prototype which is used to produce the requirements specification for the lab. prototype. Another cycle of the design is done after the lab. prototype specifications are determined, and more elaboration on the knowledge model is conducted. This cycle is repeated after the implementation and testing of the lab. prototype to produce the design of the field prototype. Once the field prototype is implemented and tested, the final design document of the production system is issued.

The following is a description of the main subjects to be considered in the design document.

III-2-1 Knowledge Representation

The model of knowledge produced as an output of the knowledge engineering activities is used as a generic representation of knowledge in both the preliminary design and the lab. prototype design documents. In the field prototype design document, a section is included describing how to map this generic design into the knowledge representation schemes supported by the tool to be used.

III-2-2 Interfaces

The need for a robust, and mature interfaces increase from one prototype to the next.
In the research prototype, very simple interfaces are used (the interfaces provided by the shell). In the lab. prototype, more enhanced user interfaces are developed. Interfaces to external packages and databases are included in the field prototype.
The same interfaces are used in the production version after adding any required enhancements.

III-2-3 Explanation Module

In the research prototype, explanation facilities depend on the capabilities provided by the shell, and are designed primarily for the developer to trace the behavior of the system. In the lab. prototype, special explanation facilities concerning the why the system is asking a certain question and how the system has reached a certain conclusion are provided. Term explanation, and detailed information concerning the conclusions are included in the field prototype. The production version uses the same explanation capabilities of the field prototype after making the required enhancements.

III-2-4 Implementation

The first decision to be taken after the approval of the design, is the selection of the implementation tool to be used. The criteria of selection depend on the facilities required by the system version being implemented. Expert system shells are suitable for research, and lab. prototypes. Although expert system shells speed up the implementation process, customized tools provide more flexibility in implementation. For this reason, a general purpose knowledge representation object language

(KROL), has been developed, and guidelines for KADS implementation using this language were defined[5]

The research prototype implementation starts with the purpose of acquiring the user requirements specifications. The primary purpose of the lab. Prototype is to test whether sufficient and appropriate expert knowledge has been obtained and represented properly for solving the class of problems associated with the given application. The secondary purpose of the prototype is to provide a realistic test of the application's man-machine interface. The main purpose of developing the field prototype is to test the acceptance of the system by other professionals different from the experts participated in providing the domain knowledge. The prototyping process can be summarized as follows:

An initial stage, in which a research prototype is created from initial knowledge captured from experts. This prototype is a throw away prototype in which the overall skeletal frame of the system is built with one or more complete sections.

An interim stage, in which a more capable prototype is derived from the initial stage prototype by testing and reviewing the knowledge with experts and prospective users. A laboratory prototype is the output of this stage.

A final stage, in which a field prototype is produced from the laboratory prototype. The field prototype development starts once the laboratory prototype is matured. The field prototype is tested by the actual users for approval.

Once the prototyping stages are successfully terminated, the implementation of the production system starts using the field prototype as a valid specification.

IV EXPERT SYSTEM TESTING

Expert system testing, is the procedure by which we can be confident that the developed expert system is consistent, complete, correct, and satisfies the original requirements and needs of the user. This procedure evolves through a cycle of three main steps, namely *Verification*, *Validation*, and *Evaluation*.

Verification ,as defined by Adrien et al. [9], is the demonstration of consistency, completeness, and correctness of software. et al.[10,11,12 *Building the system right* ed system is functionally matching the proposed design, and free of semantic and syntactic errors.

Validation is the process whereby the system is tested to show that its performance matches the original requirements of the proposed system. It is defined by Adrian et al.[9 *Validation is the determination of the correctness of the final program or software produced from a development project with respect to the user needs and requirements* et al. [10], *he right system*

Evaluation is the process whereby we ensure the usability, quality, and utility of the expert system [13]. A complete testing cycle is performed in iterations through which, the expert system is updated and refined. The following is a detailed description of the different phases of testing procedure.

IV-1 Verification

Verification process evolves through the following two main stages, during the development of the expert system:

1. Development stage

At this stage, the developers practice different functions of the implemented systems, looking for potential errors. This can be accomplished using different techniques. Generally, these techniques fall into two broad categories. Non Case-based techniques which include tracing, spying and other traditional debugging techniques, and Case-based verification techniques which are applied by spelled out in the requirements specification document.

2. Examination stage:
Before delivering the system to the user, the expert system is tested to make sure that it is running properly, by testing all the functions of the system trying to examine the performance of the system in different situations. The output of this stage is the ***verification report*** that documents the differences between system design and. This report serves two aspects. First, undocumented modifications applied to the system during development can be addressed. Second, further errors in implementation can be discovered.

As a direct consequence of this report, the design document is updated to reflect undocumented modifications in the implementation, and implementation can be revised to match design specifications.

IV-2 Validation

The validation process is performed by generating a set of test cases to be solved by both the expert system and the domain expert. Test cases are generated using the ***White-box*** testing method [10], where they are selected according to different situations of the expert system.

The output of this stage is the ***validation report***, which identifies the differences between domain experts solutions, and the expert system results.

IV-3 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 those domain experts who have participated in knowledge acquisition phase, and from the point of view of the target 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 **-** solve these test cases. Another human expert evaluates the generated solutions, and rank them according to their grades. Later on, an open discussion is held to let human experts justify their solutions. According to this discussion, evaluation of solutions may change, and the final ranking of solutions is reached. If the expert system is far from precedence, the knowledge-base must be updated. The following is a detailed, step by step, evaluation process:

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

Prepare comparison criteria
An evaluation criteria, or a formula, is designed, to enable a formal judgment on solutions generated by human experts, and the expert system. The selected criteria provides both quantitative and qualitative evaluation basis for judgment. The following is an example for qualitative and quantitative evaluation criteria:

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

$$\mathbf{P_i = 3*NE_i+2*NG_i+1*NA_i+0*NU_i}$$

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

Generate test cases
Test cases are prepared manually by knowledge engineers. The most import criteria of these test cases is that it covers both normal cases, as well as the most difficult, and rare cases.

Solving test cases
A copy of the selected test cases are given to three or four domain experts. The same cases are introduced to the expert system. Each of the domain experts as well as the Expert system works out the test cases independently

Evaluation of test cases
Solutions of test cases are evaluated in a blind manner, so that distinguishing between solutions of the expert system and solutions of domain experts becomes impossible. 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 the test cases - are given test cases solutions for evaluation according to the previously prepared formula.

The result
A score is given to each solution, and solutions are ranked according to these scores.

Observations and remarks
A meeting is 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 the knowledge engineer attend this meeting to analyze solutions and reach the final conclusion about the behavior of the expert system.

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 becomes more robust and valid.

Documentation
A detailed evaluation report is prepared at the end of evaluation process.

V EXPERT SYSTEM MAINTENANCE

System maintenance is one of the most important activities to be considered in the methodology. There are mainly two objectives of system maintenance. The first is to discover bugs, and problems that may arise during the actual, at site running of the system. The second is to make sure that the system is up to date, and possessing the most accurate and the most recent knowledge concerning the domain of application.

To fulfill the first objective, forms are designed and distributed to all sites, where users of the system can denote their remarks, complaints, and problems. These forms are collected periodically, and discussed with domain experts if it is concerning the knowledge. Modifications to the knowledge-base are documented and attached to the design document, accordingly, these modifications may reflect some changes to the implementation. Otherwise, if the problem is concerning implementation, the required modifications are made to the development version of the system, and all modifications are documented and attached to the user manual of the next version of the system.

The second objective is achieved by the arrangement for periodical meetings with domain experts, where domain knowledge is reviewed, and the latest updates in the field are discussed, and the required knowledge are acquired and augmented into the knowledge-base.

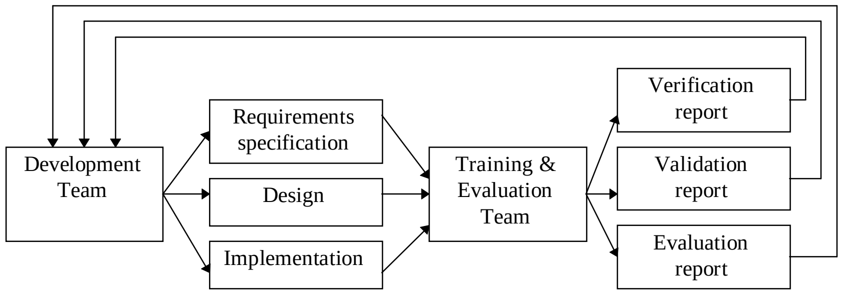
VI EXPERT SYSTEM DEVELOPMENT LIFE CYCLE

To place the previously described methodology in action, the internal organization of CLAES is designed according to the activities done through the whole expert system development life cycle. Three teams in CLAES are sharing the responsibilities of expert systems development activities: Methodology & Tools team, Development team, and Training & Evaluation team.

The Methodology and Tools team has two main responsibilities: First, setting up the standards for both, knowledge engineering and software engineering methodologies. Second, developing the necessary tools, that are required for the implementation of the target knowledge based systems.

The Development team, acquires, analyzes, models, and implements the proposed expert systems according to the standards specified by the methodology team, using the developed tools.

The Training and Evaluation team, takes the responsibility of expert systems verification, validation, and evaluation, in addition to organizing training courses for end users on the developed expert systems.



(Fig. 2) Expert system development work flow

The actual expert system life cycle runs between the development team, and the training & evaluation team. As illustrated in Fig.2, the development team starts the first cycle whose output is the requirements specification report, the design report, and the first implemented version of the expert system. The training & evaluation team takes this output and generates three reports: verification report, validation report, and evaluation report. Verification report includes discrepancies between the requirements specification report and the design report, and between the design report and the implemented system. Validation report includes differences between the implemented system behavior and the behavior expected by domain experts. Evaluation report includes comments given by domain experts other than those who participated in knowledge acquisition, to certify that the system is accepted from their point of view.

The produced reports by the testing & evaluation team are forwarded to the development team who analyzes these reports, and starts new cycle based on the comments documented in these reports.

VII Conclusion

The methodology presented in this paper ties three main aspects: theoretical basis, practical implementation, and workflow organization. The theoretical basis are derived from CommonKADS, the second generation expert system approaches state of the art. The practical experience was gained through the development of several expert systems in the domain of crop production management, this directly affected both the methodology application, and the workflow organizational structure of the lab.

A spiral model for expert systems development is introduced, through this model, knowledge engineering activities and software engineering activities are described, also we described how the expert system evolves starting from research prototype, till the development of the production version.

The internal organization of CLAES, and the workflow is introduced, showing how the responsibilities are assigned to each department, and how they interact through different stages of building the expert system according to the defined standards.

References

[1] B.J. Wielinga, W. Van de Velde, A. Th. Schreiber, and J.M.Akkermans Krivine, and Reid Simmons, editors, Second Generation Expert Systems, Pages 299-335, Springer-Verlag, Berlin - Heidelberg, Germany 1993.

[2] L. Steels Components of Expertise. AI Magazine, 11(2):29-49.

[3] B. Chandrasekaran, Generic tasks in knowledge based reasoning. High level building blocks for expert systems design. IEEE Expert. 1(3):23-30.

[4] J.McDermott. Preliminary steps towards a taxonomy of problem solving methods. S.Marcus, editor, Automating knowlegde acquisition for expert systems. Pages 225-255. Kluwer, Boston, 1988.

[5] Technical report No. TR-88-024-31, Guide to KADS implementation using the knowledge representation object language KROL. December 1993.

[6] Technical Report No. TR-88-024-41, Methodology for the Engineering of Expert Systems - Version 6.0, December, 1995.

[7] K. Orsvarn, O. Olsson, H. Hassan, Guidelines for adapting an interpretation model in an application, Ute Gappa, Hans Voss (Eds.), Proceedings of Knowledge engineering forum 1995.

[8] B. Wielinga & J. Breuker., KADS: A Model Approach to Knowledge Engineering, Esprit project 5 248 KADS II. May, 1991.

[9] Adrion, W., Branstad, M., Cherniovsky, J. Validation, Verification and Testing of Computer Software, ACM Computing Surveys , Vol. 14, No. 2,1982.

[10] O' Keefe, R.M., Balci,O., Smith E.P., Validating Expert System Performance, IEEE Expert, Winter 1987.

[11] O' Leary, D., O'Keefe, R., Verifying and Validating Expert Systems , Tutorial: MP4, IJCAI,1989.

[12] O' Keefe," Consultancy Report" Report No-CR-88-024-08 the Expert Systems for Improved crop management project.Project No EGY/88/024, August 1990.

[13] I. Benbasat and J. S. Dhaliwal, A Framework for Validation of Knowledge Acquisition , knowledge Acquisition, Vol.1 No.2, 1989, PP 215-233.