



Chapter 16: Building Expert Systems: Process and Tools

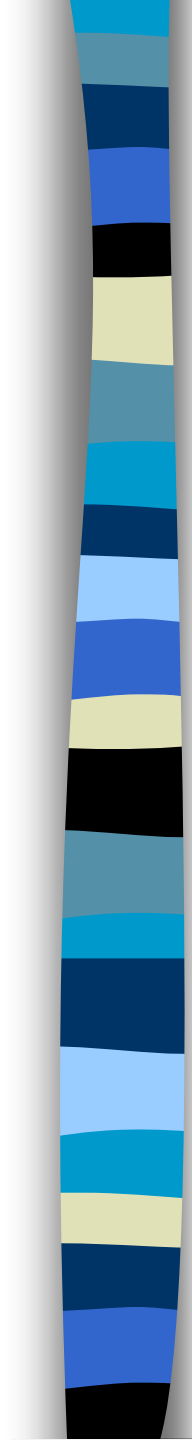
- **Overview** of the Expert System Building Process
- **Performed Differently Depending on the**
 - **Nature of the System Being Constructed**
 - **Development Strategy**
 - **Supporting Tools**

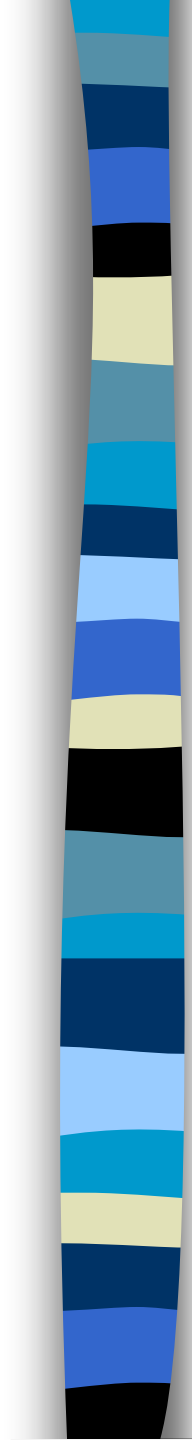


16.1 Opening Vignette: The Logistics Management System (LMS) at IBM

- **June 1985, IBM Burlington, VT Plant**
- **Industrial Engineering Group (IEG) Proposed The Logistics Management System (LMS)**
- **Expert System to**
 - **Improve line flow and utilization**
 - **Reduce cycle time in semiconductor manufacturing**

- 
- **Obtained Top Management's Full Support**
 - **Created a development team with**
 - IEG's manager
 - Several IEG building representatives
 - LMS building representatives
 - Two industrial engineers
 - Internal software development consultant

- 
- **GATEWAY and MAT provided many EIS and DSS features**
 - **Became quickly embedded into the business processes**
 - **ALERT Component increased throughput by detecting conditions that lead to future bottlenecks**

- 
- **June 1987 - LMS status elevated from exploratory to tactical**
 - **Saved Tens of Millions \$**
 - **System Enhanced and Expanded over Time**
 - **Ported to a PC**
-
- **1991: LMS became an IBM product**
 - **1994 LMS elevated to a strategic application**
 - **LMS - A long-term, successful ES**



Evolutionary Steps in LMS Development

- **Establishment of a funded project**
- **Establishment of a multidisciplinary development team**
- **Understanding and control of data**
- **Broad user involvement (later)**
- **Internal development of new expertise**
- **Expansion of the user base**
- **Personnel shifts and reassignment of tasks and responsibilities, and**
- **Emergence of new business opportunities (to a broader market)**



Critical to LMS Implementation Success

- **Top management support**
- **Business benefits**
- **Talented multidisciplinary team**



16.2 The Development Life Cycle

- For building expert systems - six phases (Figure 16.1)
- Process is nonlinear



Phases

- 1. Project Initialization**
- 2. Systems Analysis and Design**
- 3. Rapid Prototyping**
- 4. System Development**
- 5. Implementation**
- 6. Postimplementation**



16.3 Phase I: Project Initialization

(Table 16.1)

- **Problem Definition**
- **Need Assessment**
- **Evaluation of Alternative Solutions**
- **Verification of an Expert Systems Approach**
- **Consideration of Managerial Issues**



TABLE 16.1 Project Initialization Tasks

Problem definition

Needs assessment

Evaluation of alternative solutions (availability of experts, education and training, packaged knowledge, conventional software)

Verification of an expert systems approach (requirements, justification, appropriateness)

Consideration of managerial issues (project initiator, financing, resources, legal and other constraints, selling the project, identifying a champion, potential of gaining access to users and user support)

Ending milestone: approval of the project in principle



16.4 Problem Definition and Need Assessment

- **Write a clear statement and provide as much supporting information as possible**
- **Conduct a formal needs assessment to understand the problem**



16.5 Evaluation of Alternative Solutions

- **Using Experts**
- **Education and Training**
- **Packaged Knowledge**
- **Conventional Software**
- **Buying Knowledge on the Internet**



16.6 Verification of an Expert System Approach

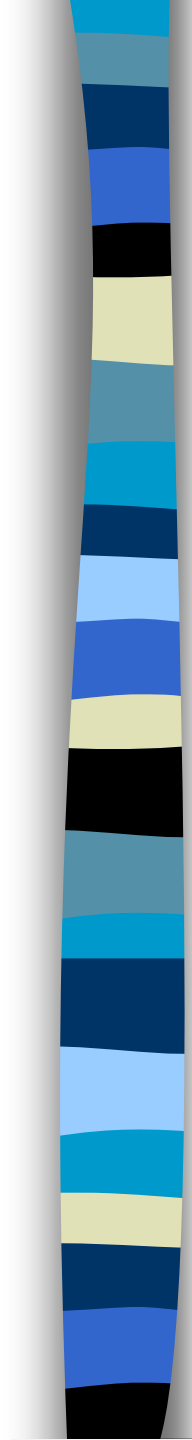
Framework to determine problem fit with an ES (Waterman [1985])

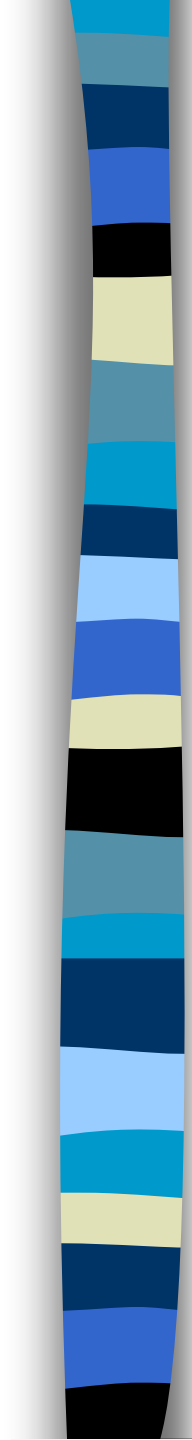
1. Requirements for ES Development
2. Justification for ES Development
3. Appropriateness of ES



1. Requirements for ES Development (all are necessary)

- 1. Task does not require common sense**
- 2. Task requires only cognitive, not physical, skills**
- 3. At least one genuine expert, willing to cooperate**
- 4. Experts involved can articulate their problem solving methods**
- 5. Experts involved can agree on the knowledge and the solution approach**

- 
- 6. Task is not too difficult**
 - 7. Task well understood and defined clearly**
 - 8. Task definition fairly stable**
 - 9. Conventional (algorithmic) computer solution techniques not satisfactory**
 - 10. Incorrect or nonoptimal results generated by the ES can be tolerated**
 - 11. Data and test cases are available**
 - 12. Task's vocabulary has no more than a couple of hundred concepts**



2. Justification for ES Development (Need at least one)

- 1. Solution to the problem has a high payoff**
- 2. ES can preserve scarce human expertise, so it will not be lost**
- 3. Expertise is needed in many locations**
- 4. Expertise is needed in hostile or hazardous environments**
- 5. The expertise improves performance and/or quality**
- 6. System can be used for training**
- 7. ES solution can be derived faster than a human**
- 8. ES is more consistent and/or accurate than a human**

Of Course: Benefits Must Exceed System Costs



3. Appropriateness of the ES (Consider 3 Factors)

- 1. Nature of the problem: Symbolic structure and heuristics**
- 2. Complexity of the task: Neither too easy nor too difficult for a human expert**
- 3. Scope of the problem: Manageable size and practical value**

Problem Selection is a Critical Factor



16.7 Consideration of Managerial Issues

- **Selling the project**
- **Identifying a champion**
- **Level of top management support**
- **End user involvement, support and training**
- **Availability of financing**
- **Availability of other resources**
- **Legal and other potential constraints**



16.8 Phase II: System Analysis and Design

(Table 16.2)

- **Conceptual Design and Plan**
- **Development Strategy**
- **Sources of Knowledge**
- **Computing Resources**
- **Feasibility Study**
- **Cost-Benefit Analysis**



TABLE 16.2 Project Conceptualization and System Analysis

Tasks

Conceptual design and plan

Development strategy

Sources of knowledge

Computing resources

Feasibility study

Cost-benefit analysis

Ending Milestone: Approved complete project plan



16.9 Conceptual Design

General Idea of the System

- General capabilities of the system
- Interfaces with other CBIS
- Areas of risk
- Required resources
- Anticipated cash flow
- Composition of the team
- Other information for detailed design later

Determine the development strategy



16.10 Development Strategy and Methodology

■ General Classes of AI Development Strategies

– Do It Yourself

1. AI development can be part of end-user computing
2. AI projects can be completely centralized
3. AI development can be decentralized, but control can be centralized
4. High technology islands
5. Utilize information centers

– Hire an Outside Developer

– Enter into a Joint Venture

– Merge, Acquire or Become a Major

Stockholder in an AI Company



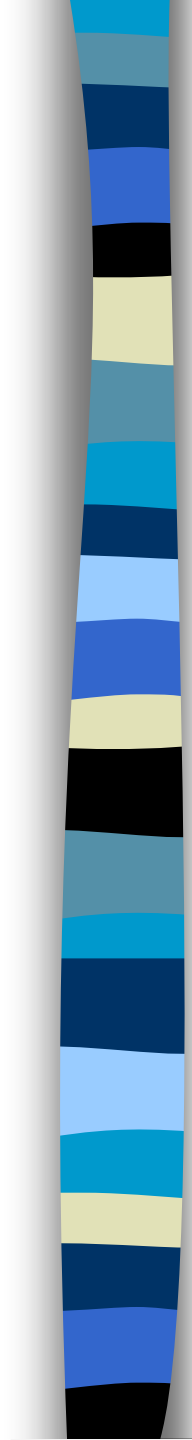
16.11 Selecting an Expert

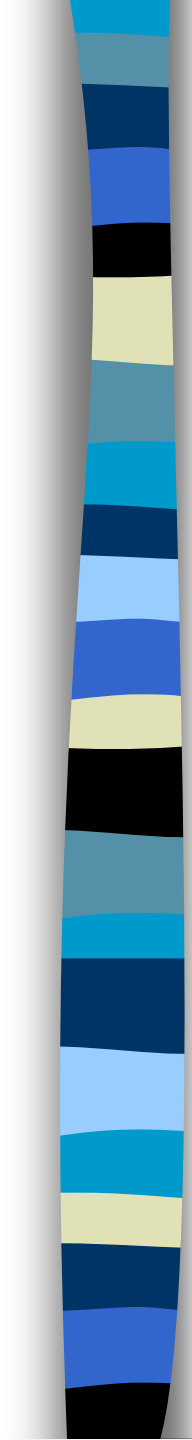
- **Experts**
- **Expertise is based on experience and can be expressed by heuristics (Table 16.3)**
- **Selection Issues**
 - Who selects the expert(s)?
 - How to identify an expert?
 - What to do if several experts are needed?
 - How to motivate the expert to cooperate?

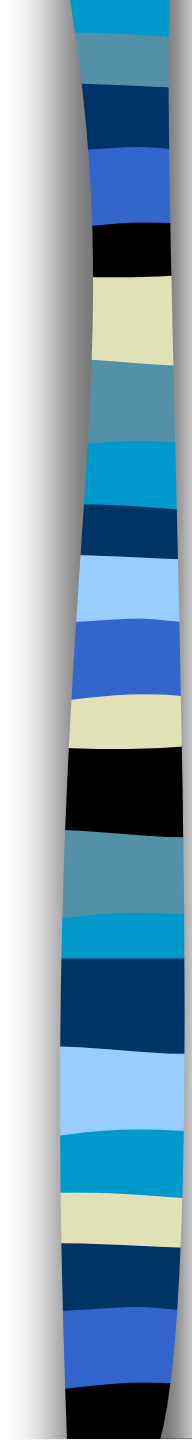


TABLE 16.3 Ideal Attributes of an Expert

- Highly developed *perceptual attention ability*—experts can "see" what others cannot.
- Awareness of the difference between *relevant* and *irrelevant* information—experts know how to concentrate on what is important.
- Ability to *simplify* complexities—experts can "make sense out of chaos."
- A strong set of *communication skills*—experts know how to demonstrate their expertise to others.
- Knowledge of when to make *exceptions*—experts know when and when not to follow decision rules.
- A strong sense of *responsibility* for their choices—experts are not afraid to stand behind their decisions.

- 
- *Selectivity* about which problems to solve—experts know which problems are significant and which are not.
 - Outward *confidence* in their decision—experts believe in themselves and their abilities.
 - Ability to *adapt* to changing task conditions—experts avoid rigidity in decision strategies.
 - Highly developed *content knowledge* about their area—experts know a lot and keep up with the latest developments.
 - Greater *automaticity* of cognitive processes—experts can do habitually what others have to work at.
 - Ability to *tolerate stress*—experts can work effectively under adverse conditions.

- 
- Ability to *tolerate stress*--experts can work effectively under adverse conditions.
 - Capability to be more *creative*--experts are better able to find novel solutions to problems.
 - Inability to *articulate* their decision processes--experts make decisions "on experience."
 - Thorough *familiarity* with the domain, including task expertise built up over a long period of task performance, knowledge of the organizations that will be developing and using the ES, knowledge of the user community, and knowledge of technical and technological alternatives.

- 
- Knowledge and *reputation* such that if the ES is able to capture a portion of the expert's expertise, the system's output will have credibility and authority.
 - Commitment of a substantial amount of *time* to the development of the system, including temporary relocation to the development site if necessary.
 - Capability of *communicating* his or her knowledge, judgment and experience.
 - Cooperative, *easy to work with* and *eager* to work on the project.

Interest in computer systems, even if he or she is not a computer specialist.

Source: Items 1-14 from S. K. Goyal et al., "COMPASS: An Expert System for Telephone Switch Maintenance," *Expert Systems*, July 1985. Reprinted from *Expert Systems* with permission of Learned Information, Inc., Medford, NJ; Items 15-20 adapted from J. Shanteau, "Psychological Characteristics of Expert Decision Makers," in *Proceedings, Symposium on Expert Systems and Audit Judgment*, University of Southern California, Los Angeles, February 16-18, 1986.



16.12 Software Classification: Technology Levels

(Figure 16.2)

- **Specific Expert Systems**
- **Shells**
- **Support Tools**
- **Hybrid Systems (Environments)**
- **Programming Languages**
- **NEW**
 - **Object-oriented Programming (OOP)**
 - **Internet/Web/Intranet-based Tools**



16.13 Building Expert Systems with Tools

- 1. The builder employs the tool's development engine to load the knowledge base**
- 2. The knowledge base is tested on sample problems using the inference engine**
- 3. The process is repeated until the system is operational**
- 4. The development engine is removed and the specific expert system is ready for its users (using a separate runtime (executable) component of the tool)**



16.14 Shells and Environments

- **Expert Systems Components**
 1. Knowledge acquisition subsystems
 2. Inference engine
 3. Explanation facility
 4. Interface subsystem
 5. Knowledge base management facility
 6. Knowledge base

- **Shell: Components 1-5 (Figure 16.3)**



Rule-Based Shells

- **EXSYS**
- **Guru**
- **NEXPERT OBJECT**
- **KEE**
- **1stCLASS**



Domain-Specific Tools

Designed to be used only in the development of a specific area

- **Diagnostic systems**
- **Shells for configuration**
- **Shells for financial applications**
- **Shells for scheduling**



Development Environments

- Support several different knowledge representations and inference methods (Table 16.4)
- Examples
 - KEE
 - ART-IM
 - Level5 Object
 - KAPPA PC



TABLE 16.4 Features of Hybrid Systems

Backward, forward and bidirectional chaining

Object-oriented programming, frames

Metarules

Semantic networks

Other graphical representations like inference trees and decision trees

Hypothetical reasoning

Case-Based Reasoning

Complete pattern matching or variable rules

Automatic rule identification

Nonmonotonic reasoning or truth maintenance

Dynamic graphics, icons, visual interactive simulations

High-quality browsing utilities



CASE library facilities

Debugger with the ability to set breakpoints or interrupt a consultation

Interfaces to databases, spreadsheets and hypermedia, neural networks, the Web and other packages

Ability to import and export knowledge, data and results

Real-time capabilities

Graphical user interface

Knowledge editor

Rule verifier

Command language

blackboard

Ability to generate computer code (e.g., C)

Explanation subsystem

Additional modeling and solution routines such as optimization, neural networks, fuzzy logic and genetic algorithms

Source: Modified from Expert Systems Strategies 4, No. 2, 1988. Published by Harmon Associates.



16.15 Software Selection

- **Complex Problem**
 - Frequent Technology Changes
 - Many criteria
- **First Check Out**
 - “PC AI Buyer’s Guide”
 - “Expert Systems Resource Guide” in *AI Expert*
 - FAQs of newsgroups on ES

**Major Issues in Selecting ES Development Software
(Table 16.5)**



**TABLE 16.5 Representative Issues in Software Selection for
Expert System Development**

Can the tool be easily obtained and installed? (includes cost factors, legal arrangements and compatibility with existing hardware)

How well is the tool supported by the vendor? Is the current version of the system fairly stable?

How responsive is the vendor to the market (in terms of upgrade features and bug fixes)?

What are the vendor's plans for improvement? Will backward compatibility be maintained (or a conversion subsystem made available)?

Can the vendor provide training and consulting if needed? How well is this supported?

How easy is the tool to learn to use as a developer? As a user?

What training programs and materials are there?

What training is necessary for the builder and for the users? Are there online tutorials?



How easy is the interface to use for the developer and for the user?

How difficult will it be to expand, modify or add a front-end or a back-end to the tool? Is the source code available or is the system sold only as a black box?

Is it simple to incorporate Prolog (or other language) functions to compensate for necessary features that are not built in?

What are the existing programming languages, databases, other knowledge bases and other systems that are likely to interface with the proposed application? Which ones will it support?

What kind of knowledge representation schemes does the tool provide? (Rules? Networks? Frames? Others?) How well do these match the intended application?

Can knowledge be easily imported and exported? If so, what formats does it support?

Can the tool handle the expected form of the application knowledge (continuous, error filled, inconsistent, uncertain, time varying, etc.)?



Do the inference mechanisms provided match the problem?

How does the inference mechanism handle uncertainty? Is it appropriate for the problem?

Does the allowable granularity of knowledge match what is required by the problem?

Does the expected speed of the developed system match the problem if real-time use is required?

Is there a delivery (consultation) vehicle available if many copies of the application will be needed (e.g., a run-time executable module)?

What is the track record of success of the package?

Are there any failures? Why?

What are the in-house software capabilities? (Are programmers available and qualified?)



What are the future plans and strategy regarding AI dissemination and the use of languages and tools? Can it generate HTML? Is there a Web server version?

What hardware and networks are present in the organization?

Is this the organization's first ES application? Or have systems been developed before? What software was used in the past?

What is the anticipated maintenance plan? Who is going to do it?

Where is the product going to be used and by whom?

How easy is it to port applications to different hardware environments?

Does the software have a good knowledge verifier and logic debugger?

What platforms will the software run on? Are the versions on the multiple platforms compatible?



Can multiple knowledge representations be used, if needed?

What knowledge acquisition aids does it provide, and what methods does it support?

Is the software capable of automatic learning from documents and databases?

What kind of explanation facilities does it support?

Do we have software in house already? Will it work?

Source: Modified from S. K. Goyal et al., "COMPASS: An Expert System for Telephone Switch Maintenance," *Expert Systems*, July 1985. Reprinted from *Expert Systems* with permission of Learned Information, Inc., Medford, NJ.



16.16 Hardware Support

**Software Choice Usually Depends
on the Hardware**

- **AI Workstations**
- **Mainframes**
- **PCs**
- **Unix Workstations**



16.17 Feasibility Study

Outline in Table 16.6

TABLE 16.6 Elements of a Feasibility Study

Economic (financial) feasibility	Cost of system development (itemized) Cost of maintenance Anticipated payoff Cash flow analysis Risk analysis
Technical feasibility	Interface requirements Networking issues Availability of knowledge and data Security of confidential knowledge Knowledge representation scheme Hardware/software availability/compatibility
Operational feasibility and impacts	Availability of human and other resources Priority as compared to other projects Needs assessment Organizational and implementation issues Management and user support Availability of expert(s) and knowledge engineer(s) Legal and other constraints Corporate culture User environment



16.18 Cost-Benefit Analysis

- **To Determine Project Viability**
- **Often Very Complicated**
- **Difficult to Predict Costs and Benefits**
- **Expert Systems Evolve Constantly**



Cost-Benefit Analysis - Complicating Factors

- **Getting a Handle on Development Costs**
 - Consider (and Revise) the **Scope** of the System
 - Estimate Time Requirements
- **Evaluating the Benefits**
 - Some **Intangible**
 - Hard to relate specifically to the ES
 - Benefits result over time
 - Not easy to assess quantity and quality
 - Multiplicity of Consequences hard to evaluate:
 - (goodwill, inconvenience, waiting time and pain)
- **Key: Identify the Appropriate Benefits**

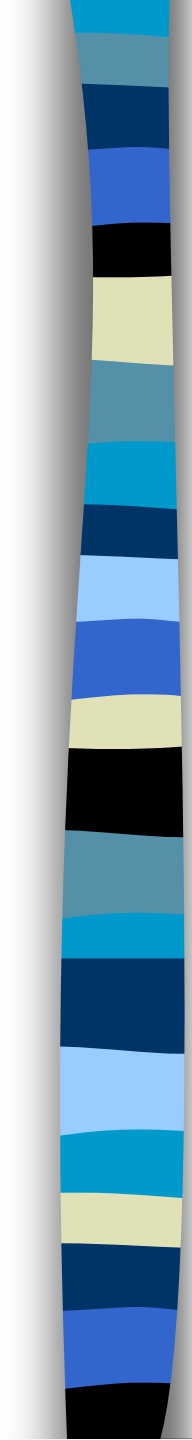
- 
- **When to Justify (Very Often!)**
 - At the end of Phase I
 - At the end of Phase II
 - After the initial prototype is completed
 - Once the full prototype is in operation
 - Once field testing is completed (prior to deployment)
 - Periodically after the system is in operation (e.g., every six or twelve months)
 - **Reality checks**
 - How to Justify?

TABLE 16.7 Commonly Used Methods of Evaluating ES Proposals

Indicator	Advantages	Disadvantages
Internal rate of return (IRR)	Brings all projects to common footing Conceptually familiar No assumed discount rate	Assumes reinvestment at same rate Can have multiple roots
Net present value or net worth (NPV NW)	Very common Maximizes value for unconstrained project or selection	Difficult to compare projects of unequal lives or sizes
Equivalent annuity (EA)	Brings all project NPVs to common footing Convenient annual figure	Assumes projects repeat to least multiple of lives, or imputes salvage value
Payback period	May be discounted or nondiscounted Measure of exposure	Ignores flows after payback is reached Assumes standard project cash-flow profile
Benefit-to-cost ratio	Conceptually familiar Brings all projects to common footing	May be difficult to classify outlays between expense and investment

Source: A. Smith and C. Dagli, "An Analysis of Worth: Justifying Funding for Development and Implementation," in E. Turban and J. Liebowitz (eds.), *Managing Expert Systems*, Harrisburg, PA: Idea Group Publishers, 1992.



16.19 Phase III: Rapid Prototyping and a Demonstration Prototype

- **Build a Small Prototype**
- **Test, Improve, Expand**
- **Demonstrate and Analyze Feasibility**
- **Complete Design**



Rapid Prototyping

- **Crucial to ES Development**
- **Small-scale System**
- **Includes Knowledge Representation**
- **Small Number of Rules**
- **For Proof of Concept**

- **Rapid Prototyping Process (Figure 16.4)**

- **Possible Tasks and Participants in the Rapid Prototyping Process (Figure 16.5)**

TABLE 16.8 Advantages of the Rapid Prototype

-
- It allows project developers to get a sense of whether it is feasible to attempt to tackle the full application using expert system technology.
 - It provides a vehicle through which to study the effectiveness of the knowledge acquisition and representation.
 - It may identify important gaps or important problems in the proposed final system.
 - It yields a tangible product of the project at an early stage.
 - It gives an opportunity to impress system funders with the capabilities of the system, helping to retain or increase support of the project.
 - It allows the possibility of an early midcourse correction of the project direction based on feedback from management, consulting experts and potential users.
 - In early development stages, it provides a system that can be field tested—yielding experience in using and testing the system and, if the tests are successful, credibility that the final system will perform its desired function well.
 - It might provide a system with enough utility that, although not a final product, may be put in the field on an extended basis. This early deployment yields benefits, such as giving experience to system deployers, system operators and system maintainers, and might identify potential problems early.
 - It accelerates the process of knowledge acquisition.
 - It makes it easier for experts to criticize existing programs or provide exceptions to the rules.
 - It makes selling the system to skeptics easier.
 - It helps sustain the expert's interest.
 - It can help to build user support
 - It provides an idea of the value of the software and the hardware, and of the degree of the expert's cooperation.
 - It provides information about the initial definition of the problem domain, the need for the ES and the like.
-

Source: Based in part on David S. Prerau, *Developing and Managing Expert Systems*, Reading, MA: Addison-Wesley, 1990, p. 39.



16.20 Phase IV: System Development

- **Complete the Knowledge Base**
- **Test, Evaluate, and Improve Knowledge Base**
- **Plan for Integration**
 - **Following the Initial Prototype's Acceptance**
 - **System Development Begins**



Use a System Development Approach

- Continue with prototyping
- Use the structured life cycle approach
- Do both
 - Tasks and Participants (Figure 16.6)



16.21 Building the Knowledge Base

Acquire and Represent the Knowledge in an Appropriate Form

- **Define the Potential Solutions**
- **Define the Input Facts**
- **Develop an Outline**
- **Draw a Decision Tree**
- **Map a Matrix**
- **Create the Knowledge Base**



16.22 Testing, Validating and Verifying, and Improving

- **Test and Evaluate the Prototype and Improved Versions of the System Both**
 - In the Lab
 - In the Field

 - Initially - Evaluate in a Simulated Environment

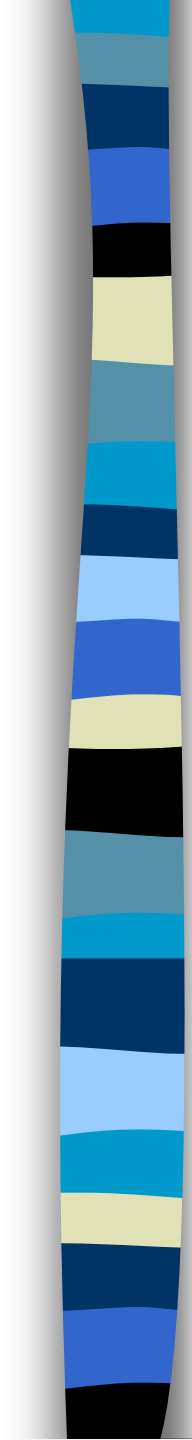


AIS In Focus 16.3: Difficulties in Evaluating an Expert System

Questions that indicate the difficulties that block evaluation studies:

- 1. What characteristics should be evaluated?**
- 2. How should performance be evaluated?**
- 3. How should the test problems be selected?**
- 4. How should one evaluate the program's mistakes?**

(Source: A. A. Assad and B. L. Golden, " Expert Systems, Microcomputers, and Operations Research," *Computers and Operations Research*, Vol. 13, No. 2/3, 1986. Reprinted with permission.)

- 
- **Modified Turing Test: Compare ES Performance to an Accepted Criterion (Human Expert's Decisions)**
 - **Experimentation**
 - **Iterative Process of Evaluation:**
 - **Refine the ES in the Field**
 - **Use New Cases to Expand the Knowledge Base**
 - **Validation - Determination of Whether the Right System Was Built**
 - Whether the system does what it was meant to do and at an acceptable level of accuracy
 - **Verification confirms that the ES has been built correctly according to specifications**



AIS In Focus 16.4: Some Requirements of a Good Expert System

- 1. The ES should fulfill a recognized and important need**
- 2. The processing speed of the system should be very fast**
- 3. The ES should be able to increase the expertise level of the user**
- 4. Error correction should be easy to perform**
- 5. The program should be able to respond to simple questions by users**
- 6. The system should be capable of asking questions to gain additional information**
- 7. Program knowledge should be easily modified**
- 8. The user should feel that he or she is in control**
- 9. The degree of effort by the novice should be reasonable**
- 10. Input requirements should be clear and simple to obtain**

(Source: Based, in part, on D. C. Berry and A. E. Hart, "Evaluating Expert Systems,"
Expert Systems, November 1990)



16.23 Phase V: Implementation

- **Acceptance by Users**
- **Installation, Demonstration, Deployment**
- **Orientation, Training**
- **Security**
- **Documentation**
- **Integration, Field Testing**



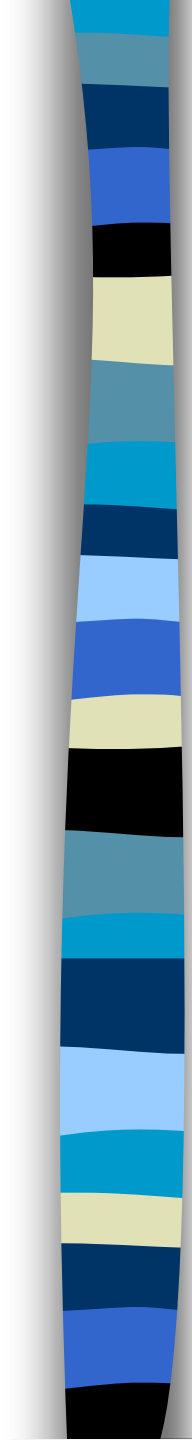
ES Implementation Issues

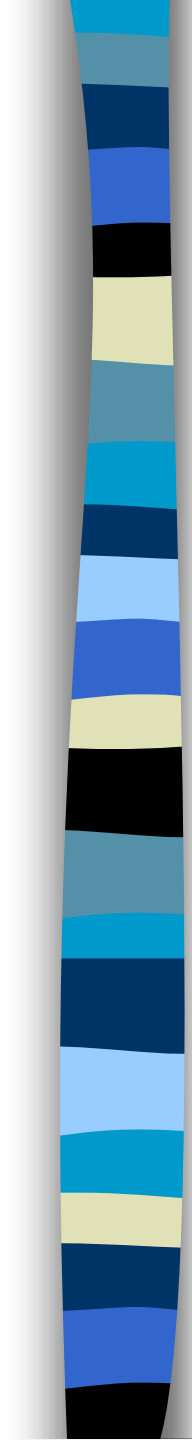
- **Acceptance by the User**
- **Installation Approaches**
- **Demonstration**
- **Mode of Deployment**
- **Orientation and Training**
- **Security**
- **Documentation**
- **Integration and Field Testing**



16.24 Phase VI: Postimplementation

- **Operations**
- **Maintenance and Upgrades
(Expansion)**
- **Periodic Evaluation**

- 
- **Expansion (Upgrading)**
 - **The Environment Changes**
 - **More Complex Situations Arise**
 - **Additional Subsystems can be Added (e.g., LMS)**

- 
- **Evaluation (Periodically)**
 - **Maintenance Costs Versus Benefits?**
 - **Is the Knowledge Up to Date?**
 - **Is the System Accessible to All Users?**
 - **Is User Acceptance Increasing? (Feedback)**



16.25 Organizing the Development Team

- **Team Varies with the Phases**
 - **Typical Development Team**
 - **Expert**
 - **Knowledge Engineer**
 - **IS Person**



Team May Also Include

- **Vendor(s)**
- **User(s)**
- **System Integrator(s)**

- **Cooperation and Communication Required!!!**
- **Possible Functions and Roles in an ES Team (Table 16.9)**

TABLE 16.9 Potential Functions and Roles in an Expert System

Team

Documentation writer	Legal advisor	System analyst
End-users	Network expert	System integrator
Expert	Programmer	System operator
Hardware (software) specialist	Project manager	System tester (evaluator)
Internet/Intranet expert	Project champion	Trainer
Knowledge engineer	Security specialist	Vendor or consultant
	Special tool developer	

Source: Based on David S. Prerau, *Developing and Managing Expert Systems*, Reading, MA: Addison-Wesley, 1990, p. 57.



Important Players

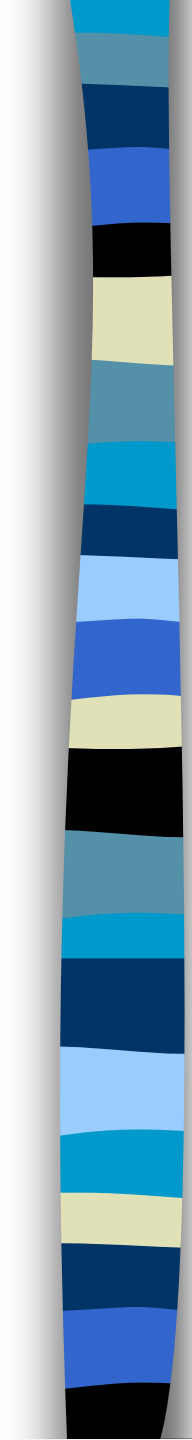
- **Project Champion**
- **Project Leader**



16.26 The Future of Expert Systems Development Processes

Expect Advances In

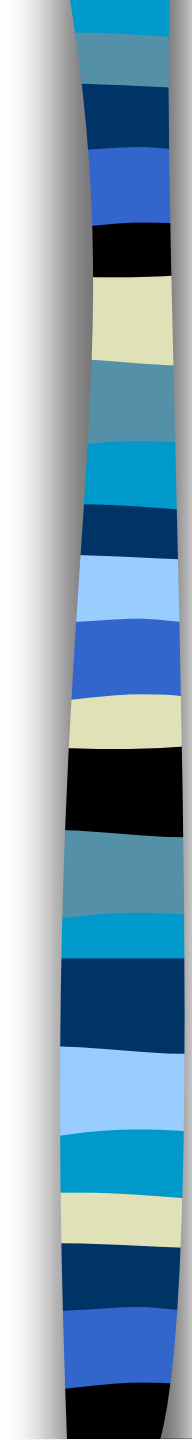
- **Flexible toolkit capabilities, including inferencing hybrids**
- **Improved languages and development systems**
- **Better front ends to help the expert provide knowledge**
- **Improved GUIs via Windows-based environments**
- **Further use of intelligent agents in toolkits**
- **Better ways to handle multiple knowledge representations**

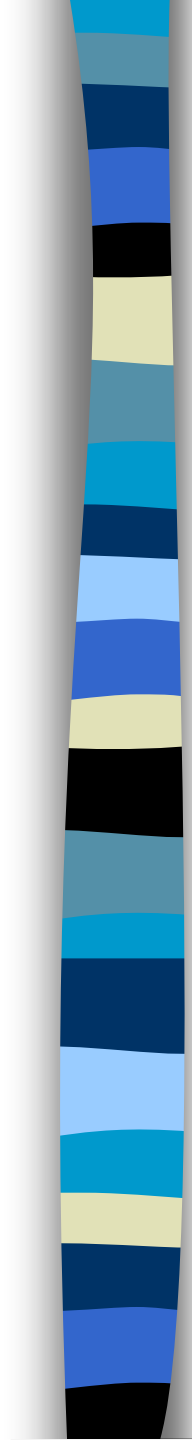
- 
- **Use of intelligent agents to assist developers**
 - **Use of blackboard architectures and intelligent agents in ES**
 - **Advances in the object-oriented approach, for representing knowledge and ES programming**
 - **Improved and customized CASE tools to manage ES development**
 - **Increased hypermedia use and development (Web)**
 - **Automated machine learning of databases and text**

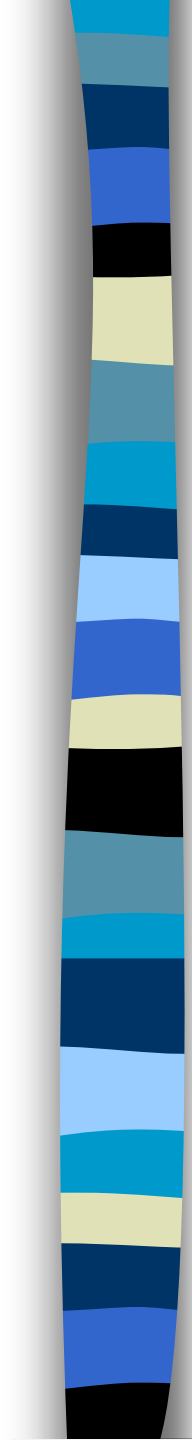


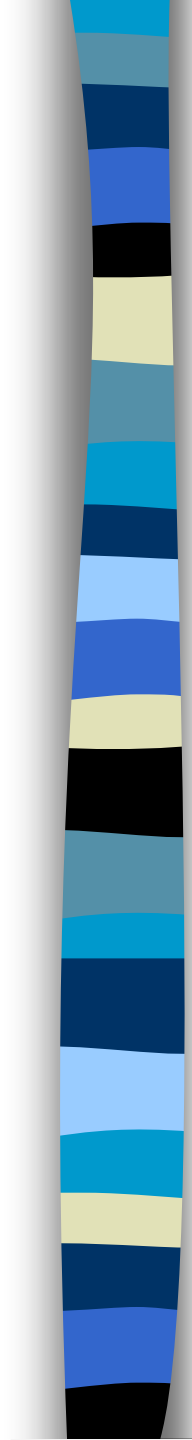
Summary

- **Building an expert system is a complex process with six major phases: system initialization, system analysis and design, rapid prototyping, system development, implementation and postimplementation**
- **Defining the problem properly simplifies the remaining development tasks**
- **Sometimes conventional technologies outperform potential expert systems**
- **Like any other project, an ES needs to be justified**
- **Without the proper level of resource commitment, the ES will fail**

- 
- **Top management support is essential**
 - **A large ES needs a champion as a sponsor**
 - **Expert systems can be developed in-house (internally) or subcontracted (several variations)**
 - **Expert systems developed by end-users can be successful**
 - **The Internet is changing the way we provide expertise in organizations adopting ES technology**

- 
- **Although ES can be developed with several tools, the trend is to develop the initial prototype with a simple (and inexpensive) integrated tool (either shell or hybrid environment)**
 - **Currently, most ES are being developed and run on standard computers (PCs, workstations, mainframes)**
 - **A feasibility study is essential for ES success**
 - **Expert systems are difficult to justify because of many intangibles**
 - **Do justification several times during the development process, and so is the go/no-go decision**

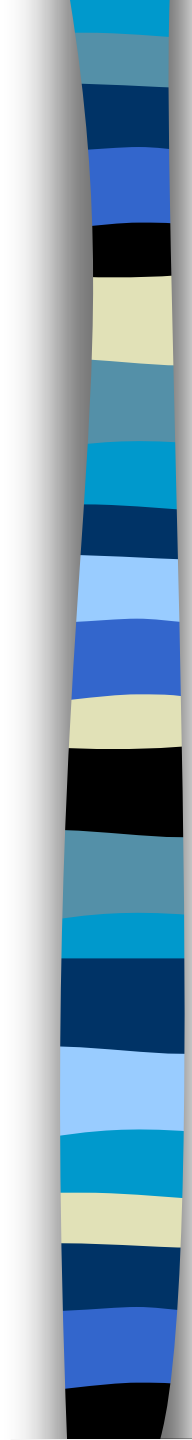
- 
- **Many ES are built by prototyping, testing, and improving and expanding. Process advantageous**
 - **Choosing the correct knowledge representation of the problem domain and the appropriate ES shell or tool are important**
 - **Major system development aspects: building the knowledge base and evaluating and improving the system**
 - **Evaluation of expert systems is difficult because of the many attributes that need to be considered and the difficulties in measuring some of them**

- 
- **Implementing an ES is similar to implementing any other CBIS. Integration may be difficult**
 - **Once the system is distributed to users, must perform: operation, maintenance, upgrade and post-implementation evaluation**
 - **Developing a proper team for ES development can be challenging; Some Important Factors: size, composition and leadership**
 - **New, powerful ES development methods include the object-oriented approach toward ES design and development, CASE tools and ES Web-server engines**



Questions for the Opening Vignette

- 1. How did the LMS project get started? Is it common in organizations for initial funding of a project like this to be made in the manner described?**
- 2. The formal status of LMS kept being elevated within IBM. Comment on these actions. Is this important? Why or why not?**
- 3. What was different about developing and maintaining LMS as a commercial product versus as an internally used system?**

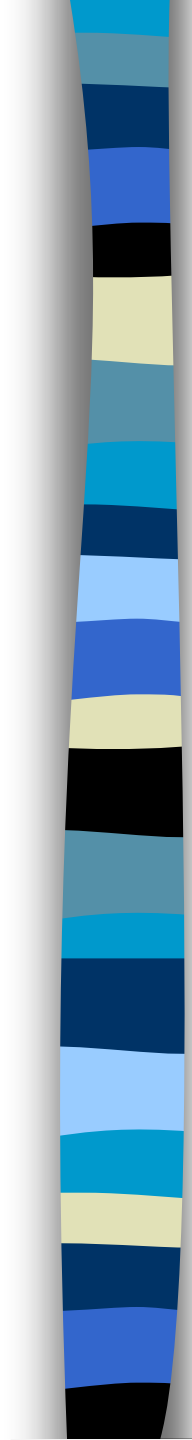
- 
- 4. Explain the evolutionary steps taken for LMS and compare them to the steps outlined in the rest of the Chapter. Why are they called evolutionary, instead of simply steps?**
 - 5. Could the standard system development life cycle have been used for LMS? Why or why not?**
 - 6. Why were the three items listed at the end of the Vignette critical to LMS implementation success? Use the material in the chapter if you must.**



CASE APPLICATION 16.1: State of Washington's Department of Labor

Case Questions

- 1. Why is it quicker to make massive changes in an application when ES is used? Why is it a lengthy process otherwise?**
- 2. What is the advantage of rapid prototyping?**
- 3. How one can retrain traditional computer programmers to work with**



APPENDIX 16-A: How to Build a Knowledge Base (Rule-based) System

- **Using Expert Systems for Wine and Food Pairing**
 - **Choosing an appropriate wine match to a certain food is not simple**
 - **Requires Expertise**
 - **This ES can help**



Building the ES

- **Step 1. Specify the problem. Pairing wine and food at a restaurant**
- **Step 2. Name the system (Sommelier)**
- **Step 3. Write a starting text**
- **Step 4. Decide on an appropriate coding for uncertainty (0 to 10)**
- **Step 5. Decide on any other parameters as required by the shell**
- **Step 6. Make a list of the choices or alternatives (12 wines)**
- **Step 7. Build the what-if rules**
- **Step 8. Prepare any concluding note for the user**



Rule Writing

- **Standard format required by the shell**
- **EXSYS uses qualifiers for the factors**



- **Qualifier No. 1**

- **The meat menu selection is**

- (1) prime rib**

- (2) grilled steak**

- (3) filet mignon**

- **One Rule**

- IF the meat menu selection is prime rib**

- THEN Pinot Noir,**

- confidence =**

- 9/10**

- AND Merlot,**

- confidence = 8/10**

- AND aged cabernet sauvignon,**

- confidence =**

- 6/10.**



To Build

- **The IF PART**
 - Select the qualifier by number (No. 1)
 - Select the appropriate value (1 for prime rib)
- **The THEN PART**
 - Pick Choices (Step 6)
 - Select the appropriate values with the appropriate confidence level
- **When Running - All rules may be consulted**
- **Each Choice tallies a confidence level**
- **They are sorted from high to low**