Expert Systems Project Management

■ Bruce G. Buchanan
■ Reid G. Smith

Sixth National Conference on Artificial Intelligence

Conference Tutorial Program

July, 1987

Copyright © 1987 by Bruce G. Buchanan and Reid G. Smith
PREFACE

It is increasingly important to understand how to turn research in expert systems into actual products and services in a business environment. This involves both a basic grasp of vocabulary and techniques and an understanding of the pragmatics of expert system development. The goal of this introductory tutorial is to enable designers and managers to understand the criteria for making decisions about expert system projects, including differentiating interesting prototypes from finished products. Several examples of successful (and other) projects will be used to illustrate the tutorial.

CONTENT

The tutorial is divided into eight sections.

1. Introduction: What Is An Expert System? We present the basic capabilities and architectural characteristics that distinguish expert systems from traditional programs.

2. Detailed Example: We use the Dipmeter Advisor system to demonstrate the basic technology. Both the system itself and the process of building a commercially viable version will be discussed.

3. Technical Details I: Representation--Three practical methods for encoding knowledge are logic, rules, and objects. Some of their relative strengths and weaknesses will be compared.

4. Technical Details II: Inference--Reasoning methods use the contents of a knowledge base to make inferences that solve a problem. Some of the methods to be discussed are: forward and backward chaining, event-driven inference, and inexact inference.

5. Technical Details III: Shell Systems--We discuss the utility of tools and shells, give examples, and develop the idea of an integrated development environment for expert systems. Criteria for selecting shells for development are contrasted with criteria for selecting run-time environments.

6. Pragmatics: Issues such as "How to choose a problem?", "How to select a shell?", "How to staff a project?", "What performance to expect?", and "What cost to expect?" are addressed. These issues go beyond the technical capabilities of AI methods to include economic, sociological, and political considerations.

7. Validation: We discuss methods for expert system testing and quality assurance. A clear statement of the problem to be solved is a major step in understanding what to validate but there are several methods for demonstrating that a system "solves the problem".

8. Future Potential and Current Assessment: We discuss current research areas, and likely progress over the next five years. An assessment of the current state of the art and its successes completes the tutorial.

INTENDED AUDIENCE

This tutorial is addressed to people who intend to manage or participate in the development of expert systems. It is also appropriate for those who need a basic understanding of the technology--the state of the art, suitable applications, considerations in tool purchase, current and potential impact. There are no prerequisites for the tutorial.

After this tutorial, attendees should be able to understand and participate in the decisions that must be made during expert system development. They will be familiar with the vocabulary and issues. They will understand the criteria involved in determining the suitability of problems, how to choose appropriate tools, realize performance and cost
expectations, basic issues in technology transfer for expert systems—in summary, the pragmatics of expert system development.

SPEAKERS

Dr. Bruce G. Buchanan, Professor of Computer Science Research and Professor of Medicine (by courtesy) at Stanford University, received his B.A. in Mathematics from Ohio Wesleyan University (1961), and his M.A. and Ph.D. from the Department of Philosophy at Michigan State University (1966). He was Instructor of Philosophy at Michigan State University and then, in 1966, joined Stanford as a Research Associate in Computer Science. In 1976 he was appointed to his present position. Professor Buchanan was a major contributor to the DENDRAL, Meta-DENDRAL, and MYCIN programs. He is currently working on several projects, including the interpretation of data about the 3-dimensional structure of proteins, constraint satisfaction in project management, and knowledge acquisition by various methods. Professor Buchanan is on the editorial boards of Artificial Intelligence, Expert Systems, Machine Learning, and The Journal of Automated Reasoning, and is Secretary-Treasurer of the American Association for Artificial Intelligence.

Dr. Reid G. Smith is a research manager for knowledge-based computer-aided engineering at Schlumberger Palo Alto Research. He received his Ph.D. from Stanford University (1979) and the M.S. from Carleton University (1969). He is the author of several papers on knowledge-based systems, object-oriented programming, man-machine interfaces, distributed problem solving, machine learning, and signal processing. He is also the author of A Framework For Distributed Problem Solving (UMI Research Press, 1981). Dr. Smith serves on the editorial board of Expert Systems: Research and Applications. His current interests lie in knowledge-intensive development environments and machine learning. He has lectured extensively on the pragmatics of knowledge-based system design and application.
1. Introduction

2. Detailed Example
   Dipmeter Advisor

3. Technical Details for Managers
   - Representation
   - Inference

<Questions and Break>
   - Hardware/Software/Shells

4. Pragmatics

5. Summary: State of the Art

<Questions>
FOUR AREAS OF COMPUTING

<table>
<thead>
<tr>
<th>Type of Processing</th>
<th>NUMERIC</th>
<th>SYMBOLIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALGORITHMIC</td>
<td>traditional scientific calculations</td>
<td>data processing</td>
</tr>
<tr>
<td>HEURISTIC</td>
<td>computation-intensive application with heuristic control (manipulators)</td>
<td>Artificial Intelligence</td>
</tr>
</tbody>
</table>
WHAT ARE EXPERT SYSTEMS?

1. AI Programs <methodology>
   symbolic information
   heuristic processing

2. Expert-level performance <quality>

3. Flexible <design>

4. Understandable <design>

5. Key Element = <implementation>
   separation of knowledge base
   from inference procedures
Why Build An Expert System?

- Replicate Expertise
- Combine Expertise

Motivations

avoid delays

distribute expertise to remote sites

make expertise available to less experienced personnel

preserve corporate knowledge

increase consistency of decisions

handle routine reasoning and bookkeeping

leave an "audit trail"
Some Applications of Expert Systems

Scheduling

*Westinghouse* — Plan manufacturing steps in a plant to avoid bottlenecks and delays

Configuration

*Digital* — Translate customers' orders for computer systems into shipping orders

Route Planning

*U.S. Air Force* — Plan an aircraft's route from base to target and back to avoid detection and threats

Loading

*U.S. Army* — Design loading plan of cargo and equipment into aircraft of different types

Equipment Design

*Delco* — Design special-purpose, low-voltage electric motors

Therapy Management

*Stanford Medical Center* — Assist in managing multi-step chemotherapy for cancer patients
Some Applications of Expert Systems

Portfolio Management

*First Financial Planning Systems (Travelers Insurance)* — Analyze an individual's financial situation and recommend investments

Equipment Tuning

*Lawrence Livermore National Laboratory* — Specify parameter settings to align a mass spectrometer

Intelligent Front Ends

*Shell Oil* — Advise persons on selecting and using subroutines in large Fortran library

Training

*Elf-Aquitaine Oil Company* — Train drillers to identify causes and repair drill bit sticking in oil wells
Some Applications of Expert Systems

Equipment Diagnosis
  General Motors — Determine causes of noises and recommend repairs

Data Interpretation
  Schlumberger — Interpret down-hole data from oil well boreholes to assist in prospecting

Risk Assessment
  St. Paul Insurance Co. — Assess risk of insuring large commercial clients

Monitoring
  IBM — Monitor operations of MVS operating system

Screening
  U.S. Environmental Protection Agency — Screen requests for information with respect to confidentiality

Troubleshooting In Manufacturing
  Hewlett Packard — Diagnose causes of problems in photolithography steps of wafer fabrication

Crop Management
  Virginia Polytechnical Institute — Assist in managing apple orchards
Basic Knowledge-Based System Architecture

Major Architectural Lesson
Knowledge Acquisition

SYSTEM

KNOWLEDGE BASE

COMPUTER SCIENTIST
(Knowledge Engineer)

DOMAIN SPECIALIST
(Expert)

Data, Texts

KE: Conceptualization
Construction
Refinement

KE: Using the framework

KE: Designing the framework

Performance
Why Automate Any Task?

- Money
- Time
- Information

"The initial overenthusiasm, which inevitably accompanies a project of this scope, can and does make the job harder..."

"The greatest benefits to be derived from a computer will probably consist of information impossible to obtain previously..."

"Our experience has shown that the computer is more adaptable to some projects than others..."

"It is impossible to overemphasize the desirability of providing for convenient corrections or deletion of errors in data..."

"The maximum justifiable amount of flexibility for extending or integrating applications must be included in the initial programming..."

Manager's Choices
DIPMETER ADVISOR SYSTEM: OVERVIEW

INPUT:

- Well Logs
  - Correlated to indicate subsurface dip
  - Conventional logs
- Geological Assertions
  - Local area knowledge
  - Specific feature knowledge

OUTPUT:

- Structural Dip Analysis
- Tectonic Feature Analysis
- Stratigraphic Feature Analysis
Gamma Ray

Dip Angle

0 50 100 0 10 20 30 40 50

Uniform Structural Dip in Shale

Downward Dip Increase Due to Drape

Random dips in Reef Core Limestone

Downward Decreasing Dips in Fore reef Talus

Uniform Structural Dip in Pre-Reef Shale
IF

there exists a normal fault pattern \((p)\), and
there exists a red pattern \((p_1)\),
such that the length of \(p_1 < 50\) ft., and
such that \(p_1\) is above the fault plane pattern of \(p\),

THEN

specialize \(p\) to be a late fault pattern

Dipmeter Advisor System
Late Fault Rule
**NORMAL-FAULT-0240**

Has been asserted by the rule NFR30 with TOP = 8958.0 with SOUT = 9252.0 match variables were:

- UNCONFORMITY = UNCONFORMITY-0241
- NORMAL-FAULT = NORMAL-FAULT-0240
- RED - RED-PATTERN = 5137
- MISSING-SECTION = MISSING-SECTION-0196

Has been modified by the rule NFR3A

**NFRA3A**

Source: J. A. Gibreath
Author: P. Pruszkowski & B. Smith, altered by D. Hammock 1-17-85

(If there is a missing section such that the magnitude of structural dip below the missing section is greater than the magnitude above, by at least 1 degree, but the amount of dip does not reverse—see rule NFR2A—and there is a red pattern associated with the missing section then the missing section is probably the result of the well crossing a normal fault or an angular unconformity.)

**NFRA3A**

Source: J. A. Gibreath
Author: P. Pruszkowski & B. Smith, altered by D. Hammock 10-26-84

(In a region where the primary type of distortion is rollover, if there is a normal fault with a red pattern greater than 200 ft. in length associated with it then the fault is probably a growth fault; the fault cuts the well somewhere below the bottom of the red pattern, the strike of the fault is perpendicular to the azimuth of the pattern, the direction to the downthrown block is opposite the azimuth of the pattern & the length of the pattern gives a rough number for the minimum out of the fault.)
DIPMETER ADVISOR SYSTEM: ACCOMPLISHMENTS

• Consistent, high-quality interpretations within areas of expertise

• Vehicle for codification of interpretation expertise

• Provocation of serious discussion among experts

• Laboratory for interpretation experimentation and investigation

• Powerful interactive workbench supporting manual interpretation
Example: Dipmeter Advisor System

Evolution

Dipmeter Advisor $\rightarrow$ Geological Workstation

- Scope
- Precision
- Effort
  - prototype $\rightarrow$ fielded system $\rightarrow$ current system
distribution of code
- Style of Doing Business
Dipmeter Advisor Prototype Fact Sheet

- Blackboard Architecture
  (major redesign)
  65 classes (e.g., fault, dune)
  (x 10 - 20)
  5 attributes/object
  (small increase)

- Forward-Chained Rule Interpreter
  (major redesign)
  90 Production Rules
  (x 2 -3, customized)
  15 Rule Sets
  (x 2 -3, customized)

- Rule Language
  30 Predicates & Functions
  (small increase)

- Feature Detection Algorithms
  (x 10)

- User Interface
  (substantial effort)
DIPMETER ADVISOR SYSTEM CODE

<table>
<thead>
<tr>
<th>Feature</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inference Engine</td>
<td>8%</td>
</tr>
<tr>
<td>Knowledge Base</td>
<td>22%</td>
</tr>
<tr>
<td>Feature Detection</td>
<td>13%</td>
</tr>
<tr>
<td>User Interface</td>
<td>42%</td>
</tr>
<tr>
<td>Support Environment</td>
<td>15%</td>
</tr>
</tbody>
</table>
Example: Dipmeter Advisor System

Features

• I/O

  natural interaction style, vocabulary
  graphical output
  mouse input
  explanation (text & graphics)
  precomputed graphics vs generated graphics

• Customized For Client

  I/O
  methods
Example: Dipmeter Advisor System

Architecture

- Interactive Assistant
  - user control of tasks, I/O modes
  - hypothetical variations
  - volunteered data
  - modifications to conclusions

- Simple Representation & Inference

- Integration
  - objects
  - rules
  - rulesets
  - procedures
    - *signal processing* + *symbolic inference*

- Flexibility
Knowledge Representation

Desiderata

• Expressive Power
  (e.g., uncertainty)

• Efficiency
  human understandability
  computational tractability

• Extensibility

• Flexibility

... Knowledge Programming
Knowledge Representation

Ways to Model a Domain

- **Action-Centered**
  
  *how to ...*

  diagnose meningitis
  detect a late fault

  inferences
  procedures (e.g., optimization)

- **Object-Centered**
  
  *what is ...*

  meningitis
  a late fault

  concept descriptions
  relations, (e.g., taxonomies)
Action-Centered Paradigm: Rules & Logic

- Primitive Unit

  Fact

- Primitive Action

  Draw Conclusion From Facts
  ... Take Arbitrary Action
THINKING IN RULES

Situation / Action

if temp > 300°C then turn off boiler.

Premise / Conclusion

if stain is grampos then organism is strep.

Antecedent / Consequent

if x is a dog then x is an animal
XCON Example Rule

A rule-based program that configures Vax and PDP-11 computer systems [~3000 rules in OPS5 shell; used for 20,000 orders (Jan 84)]

IF:
the most current activity context is distributing massbus devices, and
there is a single-port disk that has not been assigned to a massbus, and
there are no unassigned dual-port disk drives and the number of devices that each massbus should support is known, and
there is a massbus that has been assigned at least one disk drive and that should support additional disk drives, and
the type of cable needed to connect the disk drive to the previous device on the massbus is known

THEN:
assign the disk drive to the massbus
An early rule-based program that diagnoses bacteremias

Rule 27

IF:
the gram stain of the organism is gram negative, and
the morphology of the organism is rod, and
the aerobicity of the organism is anaerobic

THEN:
there is suggestive evidence (.7) that the identity of the organism is Bacteroides
Action-Centered Paradigm: Rules & Logic

Representation of Facts

*Feature Vector*
*Attribute-Value Pairs*
*Attribute-Value-Object Triples*  
+ degree of certainty

Relations Among Facts

*A and B implies C*

... with certainty x

*For All x, y.*
*f(x) and g(y) implies h(x,y)*

... with certainty in f,g
Rules & Logic: Difficulties

- Temporal Relations
- Sequencing ... Procedures
- Modularity
- Consistency
- Descriptive Models
  structural
  causal
Object-Centered Paradigm: Frames & Objects

• Primitive Unit

  Object with Slots & Values

• Primitive Action

  Message to Object
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Strike</td>
<td>0</td>
</tr>
<tr>
<td>TimeOfFault</td>
<td></td>
</tr>
<tr>
<td>Slip</td>
<td>50.0</td>
</tr>
</tbody>
</table>

Slots
Object-Centered Paradigm: Frames & Objects

Representation of Objects

Attribute-Value-Object Triples
Methods
Facets
Attached Procedures

Relations Among Objects

A is-a B

A <relation> B

... with certainty x
Object: NormalFault

Synonyms:

Groups:

Type: CLASS

Edited: 13-Sep-84 13:08:06 By: REID

Picture:

HangingWallBlock \{DownthrownBlock\}:
UpperDistortionRegion:
BrecciaRegion \{CrushedZone\}:
FaultPlane:
LowerDistortionRegion:
FootWallBlock \{UpthrownBlock\}:
Strike:
FaultAngle \{Hade\}:
DirectionToDownthrownBlock:
Slip:
Throw:
TimeOfFaulting:
Draw: DrawFault
Instantiate: InstantiateFault
Detect: (RuleNFR1 RuleNFR3 RuleNFR4 RuleNFR5 RuleNFR7)
Specialize: (RuleNFR6 RuleNFR9 RuleNFR10 RuleNFR11 RuleNFR12)
Frames & Objects: Difficulties

• No Inference Mechanism

• Soft Subclasses

• Consistency of Descriptions
different static/dynamic views
Inference: Desiderata

- Appropriate Use of Data
- Models of Reasoning
- Accuracy

- Efficiency

- Uncertain Reasoning

- Understandability

- Control of Interaction
- I/O
Inference as Search:
The Generator

Random or Systematic:

- Selection from a List
- Successor Function
- Plausible Move Generator
A Rule Interpreter
Data-Driven Reasoning: Schematic Example

Data... x1, x2, x5

Rules...

• R1: IF x1 & x2 THEN y1
• R2: IF x3 & y1 THEN y2
• R3: IF x3 & x4 THEN y3
• R4: IF y2 & x5 THEN z1
• R5: IF y1 & x5 THEN z2

Conclusions... (y1), z2
FORWARD CHAINING

if stain is grampos then organism is strep.

if stain is gramneg then organism is e.coli.

if organism is strep or bacteroides then penicillin is indicated.

if a drug is indicated and don't know whether allergic to the drug then ask whether allergic to the drug.

if a drug is indicated and not allergic to the drug then prescribe the drug.
Goal-Driven Reasoning: Schematic Example

Goal... z2 \textit{(i.e., is z2 true?)}

Rules...

- R1: IF x1 & x2 THEN y1
- R2: IF x3 & y1 THEN y2
- R3: IF x3 & x4 THEN y3
- R4: IF y2 & x5 THEN z1
- R5: IF y1 & x5 THEN z2

Questions...

- Q1: y1 (internal subgoal)
- Q2: x1 (?) KNOWN
- Q3: x2 (?) KNOWN
- Q4: x5 (?) KNOWN
Q: What about prescribing penicillin?

if stain is grampos then organism is strep.

if stain is gramneg then organism is e.coli.

if organism is strep or bacteroides then penicillin is indicated.

if a drug is indicated and don't know whether allergic to the drug then ask whether allergic to the drug.

if a drug is indicated and not allergic to the drug then prescribe the drug.

A: Prescribe penicillin if the stain is gramos and patient is not allergic to penicillin.
Event-Driven Reasoning
Schematic Example

Goal... interpret events

Rules...

- R1: IF x1 & x2 THEN y1
- R2: IF x3 & y1 THEN y2
- R3: IF x3 & x4 THEN y3
- R4: IF y2 & x5 THEN z1
- R5: IF y1 & x5 THEN z2

<table>
<thead>
<tr>
<th>Event</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>y1</td>
</tr>
<tr>
<td>x2</td>
<td>z2</td>
</tr>
<tr>
<td>x4</td>
<td>expect x5</td>
</tr>
<tr>
<td>x5 (maybe)</td>
<td>--</td>
</tr>
</tbody>
</table>
THE DIAGNOSTIC PROCESS

• NOTICE that a problem exists.

• ISOLATE the problem.

• GENERATE alternative hypotheses.

• EXPERIMENT to gather more information.

• RANK hypotheses.

• SELECT the best explanation.

• CONFIRM the choice.

• ACT on the diagnosis.
Loops Gauges
procedure UpdateMeterDisplay (Setting)
    Send(ClearDisplay)
    Send(SetDisplay Setting)
end
Inference: Efficiency

- Policies — usually implicit
e.g., satisficing, plausible set

- Heuristics — should be in knowledge base
e.g., rules, demons

- Strategies — trend to declarative form
Strategic Reasoning: Implementations

**General Advice about What to Do**

- **Meta-Rules**
  - Prune
  - Reorder

- **Task Definitions**
  - Rule Sets
  - Procedure Schemas

- **Inherited Procedures**

- **Default Reasoning**
Uncertainty

- Evidence Gathering Model
- Uncertain or Incomplete Data
- Probabilistic or Uncertain Rules
- Some Calculi of Uncertainty
Uncertain Reasoning: Some Calculi

1. CASNET/EXPERT
   - thresholds of major & minor findings

2. INTERNIST
   - $\Sigma(\text{pos.findgs}) - \Sigma(\text{neg.findgs})$

3. MYCIN/EMYCIN
   - $CF = \text{increased importance} = \frac{X + Y(1-X)}{-1 \quad 0 \quad 1} + 1$, $X \times (1-X)$

4. Bayes' Theorem
   \[ E = (e_1 & e_2) \]
   \[ H = \{h, h_2, \ldots, h_n\} \]
   \[
P(h | E) = \frac{P(e_2 | h & e_1) \times P(h | e_1)}{\Sigma P(e_2 | h_i & e_1) \times P(h_i | e_1)}
\]
Uncertain Reasoning: Some Calculi

5. Liklihood Ratios

6. Fuzzy Logic

7. Dempster-Shafer Belief Rules
Inference: Understandability

• Run time
  confidence in correctness
  responsibility
  personal
  audit trail

• Development time
  confidence
  debugging - localizing errors

• Types of Explanations
  WHY questions
  Factual questions / Definitions
  HOW questions
  WHY NOT questions
14) Is the patient's illness with ORGANISM-1 a hospital-acquired infection? ** WHY

[i.e., WHY is it important to determine whether or not the infection with ORGANISM-1 was acquired while the patient was hospitalized?]

[2.0] This will aid in determining the identity of ORGANISM-1. It has already been established that
[2.1] the morphology of ORGANISM-1 is rod
[2.2] the gram stain of ORGANISM-1 is gramneg
[2.3] the aerobicity of ORGANISM-1 is facul

Therefore, if
[2.4] the infection with ORGANISM-1 was acquired while the patient was hospitalized

then
there is weakly suggestive evidence (.2) that the identity of ORGANISM-1 is pseudomonas
[RULE050]
** WHAT WAS THE IDENTITY OF ORGANISM-1? 
= ORGANISM-1
The identity of ORGANISM-1 is e.coli (.51),
salmonella (.45), klebsiella (.4), pseudomonas (.22),
enterobacter (.21), proteus (.19), pseudomonas-
aeruginosa (.08), serratia (.02)
** HOW DID YOU DECIDE THAT ORGANISM-1 WAS AN E.COLI? 

I used rule084 to conclude that the identity of ORGANISM-1 is e.coli. This gave a cumulative c.f. of (.51).

I used rule003 to conclude that the identity of ORGANISM-1 is e.coli. This gave a cumulative c.f. of (.43).
** WHY Didn’T YOU THINK THAT ORGANISM-1 WAS NEISSERIA?**

The following rules could have been used to determine that the identity of ORGANISM-1 is neisseria: 9 However, none of these succeeded in the context of ORGANISM-1. If you would like an explanation for why any of these rules failed, please enter their numbers:

** 9

Clause 2 of rule009 ["the morphology of the organism is coccus"] was already known to be false for ORGANISM-1, so the rule was never tried.
Inference: Control of Interaction

- Hired Gun Model
e.g., DENDRAL, PUFF

- Consultant Model
e.g., MYCIN, Prospector

- Assistant Model
e.g., Dipmeter Advisor, Oncocin

- Tutor Model
e.g., Guidon
Inference: Difficult Issues

- Common Sense Reasoning
- Accuracy vs Precision Tradeoff
- Knowing What You Know
- Brittleness
- Reasoning with Very Large, Unstructured Data
- Non-monotonic Reasoning
Hardware for Expert Systems

Possible Choices

- PC
- Lisp Machine
- Workstation
- Mainframe

Evaluation Criteria

- Cost, Availability, Support
- Performance
- Graphics & Interaction
- Development Environment including language support
- Standards including networking

Development Platform
vs
Delivery Platform
Application-Specific Knowledge Base

Graphics

Knowledge Acquisition

Inference Engine

Explanation

Knowledge Representation

Shell

The Importance of Powerful Tools
Evaluating Shells
—What To Look For—

• Representation Choices
  objects, rules, tasks

• Inference Mechanisms

• Built-In Problem-Solving Methods
  heuristic classification

• Specific Features
  multiple hypothesis support
  dependencies
  uncertainty

• Extensibility

• Ability to Scale Up
Evaluating Shells
—What To Look For—

• Editing/Debugging Facilities
  * browsing
  * managing complexity

• Graphics & User Interaction

• Use in Fielded Systems

• Intended Users
  * novice, expert, programmer
Evaluating Shells

—What To Look For—

• Efficiency
  *compilers*

• Software Engineering Tools
  *release management*
  *performance analysis*

• Access to Standard Languages

• Integration

• Portability & Standards

• Cost, Vendor Support, ...

Development Environment
*vs*
Delivery Environment
Direct Development Costs

<table>
<thead>
<tr>
<th>1K$</th>
<th>10K$</th>
<th>100K$</th>
<th>1M$</th>
</tr>
</thead>
</table>

**Hardware**

- **PC**
- **Workstation**
- **Mainframe**

**Shell**

- **inference only**
- **with environment**

**People**

- 0.5 - 2
- 20-50

Midrange Project:

3 workstations & shells, 5-10 person-years
Pragmatics

- Selecting an application
- Steps in constructing an expert system
- The development team
- Technology transfer: Steps in fielding an expert system
- Pitfalls
- Models of Successful Efforts
What Makes A Good Application?

Problem Definition

Importance:

the task has a high payoff

the benefits of using a system justify the costs of developing and using it
What Makes A Good Application?

Expertise

there are recognized experts

the experts are provably better than amateurs

there is general agreement about the knowledge
What Makes A Good Application?

Managerial Components

the commitment of an expert can be obtained

there is a supportive manager with clout

adequate computation resources, machines and staff, exist

a product development organization exists
What Makes A Good Application?

Target Community

the target users have been defined

the target users want a system and are ready to use it

the context of use has been defined

users can exercise common sense

the users and the experts share a conceptual framework
What Makes A Good Application?

Problem Definition

Scope:

the skill can be routinely taught to neophytes

the task takes an expert a few minutes to a few hours

the knowledge is bounded

the knowledge is primarily symbolic

algorithmic solutions are not practical

incremental progress is possible

data and test cases are available
Steps in Constructing an Expert System

- Identification
  
  Problem  
  Target Community  
  Resources  

- Conceptualization & Formalization
  
  Concepts  
  Methods  
  Representation  

- Implementation by Exploratory Programming
  
  Incremental Refinement  
  Experimentation with Real Data and Real Users  
  Revision, Extension ... Redesign
Steps in Constructing an Expert System

Time

• Identification
days/weeks

• Conceptualization & Formalization
weeks

• Implementation
months

Total
6 months to 2 years for systems that "interest" the target community

Assumptions
Developers, machines, expertise exist in-house
Tools that fit the problem exist in-house along with knowledge of how to use them
Expert System Timeline

Feasibility and Experimental Prototype (EXP)

- Seminar for managers/users
- 2nd Feasibility Study (~ 1 week)
- Background Study
- Environment/Tool Familiarization
- Knowledge Engineer Training
- Build and Refine Knowledge Base
- Domain Expert (in house)
- Test Cases/Validation
- Interface Development
- System Support ... Development Environment ... Target Environment

Engineering Prototype (ENP)

- KB Maintenance
- KB Extension
- User Training

Time

6 Month to 2 Year Range
The Development Team

• Domain Expertise

• Prototypical User

• Shell & Tool Design

• Knowledge Engineering

• System/Programming Support

• Software Engineering
Development Team Training

• a few days
  criteria, intuition

• a few weeks
  how to..., hands-on experience

• a few years—degree program
  conceptual understanding
Technology Transfer: Steps in Fielding an Expert System

• Testing and Validation
  
  Performance
  \textit{Scope, Accuracy, Efficiency}
  
  Human Engineering
  \textit{Efficiency, Robustness}

• Software Engineering for Target Environment
  
  \textit{Hardware, Network}
  \textit{Software}
  \textit{Interface}
  \textit{Integration}

• Documentation

• Training

• Marketing & Sales

• Maintenance
Incremental Development

- Tool Development
- Knowledge Expansion

Performance vs. Time
Pragmatics: Pitfalls

- Excessive Aspirations
- Inadequate Resources
- Inadequate Management Support
- Poor Problem Selection
- Forgetting the User
- Premature Optimization
- Technology Transfer & Sociology
Pragmatics: Ways To Be Successful

- Digital
  *collaborate with universities*

- IBM
  *redirect computer science talent*

- Schlumberger
  *build a research group*

- General Motors
  *form strategic partnerships*

- Travelers Insurance
  *contract with AI company*

- Kawasaki Steel
  *buy shells & train programmers*

Use AI to get a single job done
*vs*
Broad commitment to computer science
Validation:

- What is the question?
- What counts as an answer?
- How do you get the answer?
Validation:
Questions to Ask

• How good is this program?
  a. conceptual framework
  b. particular knowledge base

• Is this program at least "as good as" specialists [novices, users, experts] over problems in domain D, for users of class U?

• Bottom line = productivity
  i.e., cost/benefit tradeoff
Validation:
Dimensions of Answers

• Computational
  * time & space
  * robustness
  * consistency
  * completeness
  * portability
  * extensibility

• Psychological
  * ease of use
  * ease of learning
  * understandability
  * elegance—look & feel

• Performance
  * accuracy
  * precision
  * reliability
  * scope: breadth & depth
Validation: Methods Used for Some Expert Systems

- Commercial Use

  XCON
  Dipmeter Advisor

- Comparison with Test Data

  DENDRAL
  Paradise
  ABLE/SLAC
  AI/RHEUM

- Comparison with Specialists

  INTERNIST
  MYCIN
Validation:
Summary of Considerations

- Consider validation in initial problem definition
- Define the question
- Define the gold standard
- Measure the appropriate characteristics
- Use good statistical sense in design & execution of study
State of the Art

• Level of Effort
• Problem Size
• Problem Scope
• Shells
• Limitations/Current Research
• Some Scenarios
COSTS OF BUILDING
KNOWLEDGE-BASED SYSTEMS

ASSESSMENT
  a few days or weeks

PROTOTYPING
  1-2 man-years knowledge engineer
  0.5 man-years domain specialist

DEVELOPMENT
  2-5 man-years knowledge engineer
  half-time from domain specialist

FIELDING
  software engineering
Sizes of Some Solution Spaces

MYCIN \( \binom{120}{6} \) \( \approx 10^9 \) \( \rightarrow 6 \times 10^6 \)

INTERNIST \( \binom{571}{3} \) \( \approx 31 \times 10^6 \)

DIPMETER ADVISOR \( \approx 500^{65} \)

XCON \( \binom{94}{20} \)
RULE-BASED & OBJECT-CENTERED EXPERT SYSTEMS

(# RULES / # OBJECT NAMES)

<table>
<thead>
<tr>
<th>System</th>
<th>Percentage</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MYCIN</td>
<td>62.3</td>
<td>(1059 / 17)</td>
</tr>
<tr>
<td>XCON</td>
<td>61.0</td>
<td>(5739 / 94)</td>
</tr>
<tr>
<td>XSEL</td>
<td>27.1</td>
<td>(2148 / 79)</td>
</tr>
<tr>
<td>XFL</td>
<td>21.8</td>
<td>(1618 / 74)</td>
</tr>
<tr>
<td>INTERNIST</td>
<td>5.2</td>
<td>(2600 / 500)</td>
</tr>
<tr>
<td>DIPMETER</td>
<td>1.4</td>
<td>(90 / 65)</td>
</tr>
<tr>
<td>TEKNOWL.</td>
<td>0.4</td>
<td>(1242 / 3317)</td>
</tr>
</tbody>
</table>
VOCABULARY SIZE

(#obj + #attrib + #vals)

MYCIN  715+  = (17 + 257 + 441+)
INTERNIST 4674  = (571 + 4100 + 3)
XCON  934+  = (94 + 840 + ??)
XSEL  408+  = (79 + 329 + ??)
XFL  326+  = (74 + 252 + ??)

NOTES:
1. Attributes may take continuous numerical values.
2. Objects may be instantiated many times.
3. Rules may apply to many different contexts.
# RANGES OF KB SIZE

## Vocabulary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td># objects [concepts]</td>
<td>10 - 100</td>
</tr>
<tr>
<td># attributes/object</td>
<td>10 - 1000</td>
</tr>
<tr>
<td># legal values/attribute</td>
<td>3 - 100</td>
</tr>
</tbody>
</table>

## Inferential Relations*

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth of Inference Chains</td>
<td>4 - 10</td>
</tr>
<tr>
<td>Degrees of Uncertainty</td>
<td>continuous</td>
</tr>
</tbody>
</table>

## Data / Case Information

- Noisy Data: some
- Missing Data: some
- Inconsistent Data: some

*The number of inferential links (rules) is dependent on the number of things being linked and the complexity of the inferences in the domain.
Problem Scope

• Importance
  • small $\neq$ unimportant
  • cost/benefit analysis
  • number of experts
  • training time for new persons
  • lost time from not getting it right the first time

• Feasibility
  • telephone test
  • training manual

• Size
  • number of input descriptors
  • size of solution space
  • size of total vocabulary
  • average time for experts
  • size of manuals & handbooks
Problem Solving Shells

- Representation Choices
- Inference Mechanisms
- Run-Time Environment
  - Explanation
  - Presentation
  - Options - data entry
  - task
  - Integration
- Development Environment
  - Case management
  - Editor
  - Explanation
  - Debugging aids
  - Compiler
  - Software Engineering Tools
Problem Types: Questions

• Diagnosis/ Troubleshooting  
  *What is the cause of the problem?*

• Data Interpretation  
  *What do these data mean?*

• Monitoring/ Real-Time Control  
  *What’s going on?*
Problem Types: Questions

- Scheduling/Planning/Therapy
  What is the sequence of steps to get to the goal?

- Configuration/Layout
  What is the 2-d plan that satisfies the constraints?

- Design/Spatial Arrangement
  What 3-d configuration fits the specifications?

- Constraint Satisfaction
  What description satisfies all of the constraints?
Problem Types:
Methods

• Search = General Model

• Classification / Evidence Gathering

• Skeletal Planning / Stepwise Refinement

• Stepwise Construction / Plan-Generate-Test

• Means-Ends Analysis / Subgoaling

• Constraint Propagation
WHAT IS THE STATE OF THE ART?

Expert-level performance on narrow problems
Sufficient knowledge to solve important problems
Understandable, but limited explanation of line of reasoning
Natural human interface, both graphical and text, but with stylized language and limited vocabulary
Flexible knowledge bases
Requirement for an experienced "knowledge engineer"
Limited to one expert as the "knowledge czar"
# The Current State of Some Hard Problems

<table>
<thead>
<tr>
<th>Practice</th>
<th>Theory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inexact Reasoning</td>
<td>CF Model</td>
</tr>
<tr>
<td>Knowledge Engineering</td>
<td>An Art</td>
</tr>
<tr>
<td>Learning By Induction</td>
<td>Hand-Crafted</td>
</tr>
<tr>
<td>Default Reasoning</td>
<td>Inheritance</td>
</tr>
<tr>
<td>Common-Sense Knowledge</td>
<td>Add Items</td>
</tr>
<tr>
<td>Strategies</td>
<td>Meta-Level</td>
</tr>
</tbody>
</table>

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Over-Developed</td>
</tr>
<tr>
<td>Emerging</td>
</tr>
<tr>
<td>Puzzling</td>
</tr>
<tr>
<td>Not Well Explored</td>
</tr>
</tbody>
</table>

---

*MA2-106*
Some Scenarios

- Autonomous Agent
- Consultant
- Assistant
- Critic
- Tutor
Embedding a Knowledge-Based System: An Intelligent Assistant

A user gets a number of advantages from using the system—one of which is symbolic inference.

In watching the system operate, an observer might never realize that any intelligence is involved.
References

GENERAL

Buchanan, B.G.
Also KSL Report No. 85-37.

Hayes-Roth, Frederick, Donald A. Waterman, and Douglas B. Lenat[eds.].
*Building Expert Systems.*
Addison-Wesley, Reading, MA, 1983.

Jackson, Philip C., Jr.
*Introduction to Expert Systems.*

Mishkoff, H.C.
*Understanding Artificial Intelligence.*
Texas Instruments Information Publishing Center, P.O. Box 225474, MS-8218,
Dallas, Texas 75265, 1985.

Waterman, D.

TECHNICAL

J. Bachant and J. McDermott.
R1 Revisited: Four Years in the Trenches.

B. G. Buchanan and E.H. Shortliffe.
Addison-Wesley, Reading, MA, 1984.

R. Fikes and T. Kehler.
The Role of Frame-Based Representation in Reasoning.
Harmon, P.
Inventory of Existing Expert Systems Applications.

Nii, H. P.
Blackboard Systems Part One: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures.

Nii, H. P.
Blackboard Systems Part Two: Blackboard Application Systems, Blackboard Systems from a Knowledge Engineering Perspective.

Richer, M.H.
An evaluation of expert system development tools.

R. G. Smith.

M. J. Stefik and D. G. Bobrow.
Object-Oriented Programming: Themes and Variations.

**JOURNALS & NEWSLETTERS**

_Applied Artificial Intelligence Reporter_, published by University of Miami. Subscription inquiries to ICS Research Institute, P.O. Box 1308-EP, Fort Lee, New Jersey 07024.

_AI Magazine_, published by the American Association for Artificial Intelligence. Subscription inquiries to AAAI, 445 Burgess Drive, Menlo Park, California 94025.

_Expert Systems_, published by Learned Information. Subscription inquiries to Learned Information, 143 Old Marlton Pike, Medford, New Jersey 08055.


AAAI-87 TUTORIAL EVALUATION FORM

*******PLEASE TEAR THIS PAGE OUT OF BOOKLET AND LEAVE AT THE DOOR*******
OR FOLD, STAMP, AND MAIL TO ADDRESS ON THE OPPOSITE SIDE.

TUTORIAL #: NAME OF TUTORIAL: .................................................................
Speaker 1: .................................................................
Speaker 2: .................................................................

CONTENT:
Were any topics covered that might have been omitted? 

Were any topics omitted that you wanted to have covered? 

Was the conference brochure description accurate? 

Was the technical level of the tutorial appropriate?
Too general?___ Too detailed?___ Too difficult?___ Too simple?___

Were the speakers well prepared? (1) (2) 

Were the speakers understandable? (1) (2) 

OVERALL RATINGS:
Content: Excellent ___ Good ____ Fair ___ Poor ____
Speaker (1) Excellent ___ Good ____ Fair ___ Poor ___
(2) Excellent ___ Good ____ Fair ___ Poor ___

Would you recommend this tutorial to your colleagues? 

Why or why not? 

Was the advance reading material you received (if any) useful? 

Additional/other readings you would recommend? 

COMMENTS REGARDING THE TUTORIAL PROGRAM:
Any other tutorials you would especially like to attend next year? 

Any other speakers you would especially like to hear next year? 

Specific changes in the format of the tutorials you would like, such as lengthening/shortening, time for questions, etc.? 

Other advice to give the tutorial chair next year? 

ABOUT YOU:
Affiliation: Hardware Manufacturer___ Software Manufacturer/Publisher___
Computer/DP services/timesharing___ Research/education/consulting___
Government: Federal/state/local___ Banking/finance/insurance___
Transportation/communications___ Other(specify): ___

Role: Student__ Graduate student__ Staff Scientist__ Research Scientist__
Consultant__ Engineer__ Programmer/Analyst__ Middle management__
Project leader__ Systems analyst__ University/College educator__
Administrator__ Other(specify): ___

PLEASE WRITE OTHER COMMENTS IN THE MARGINS: