


# Knowledge Engineering

CompSci 367  
Assoc. Prof. Ian Watson


© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz



# Knowledge Engineering (KE)

- ⚡ What is It?
- ⚡ Where did it come from?
- ⚡ What does it involve?
- ⚡ Why is it important?
- ⚡ How can it help me?
- ⚡ Where do I find out more?


© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz



# What is KE?

- ⚡ The theory and practice of mechanized problem solving in a manner imitating what is believed to be the human thinking process
- ⚡ The practice involves the design and crafting of results of AI research into practical system applications, for either computer processing or similarly precise domain model manipulation.


© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz



# What is KE?

- ⚡ An ad hoc artistic process ....
- ⚡ A kind of non-scientific engineering, based on experience not experiment. Heuristic rather than algorithmic.
- ⚡ A systematic approach to capturing and employing expert knowledge to solve problems for non-experts
- ⚡ A process for implementing knowledge management systems involving
  - ⚡ Knowledge Elicitation
  - ⚡ Knowledge Acquisition
  - ⚡ Knowledge Analysis
  - ⚡ Knowledge Representation - usually as If-Then Rules


© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz



# Where Did KE Come From?

- ⚡ Artificial Intelligence research
- ⚡ Specifically Expert System and Rule-Based System development
- ⚡ Didn't we go through all this about 25 years ago?
  - ⚡ That was before we discovered the need to manage **Explicit** and **Tacit** Knowledge
  - ⚡ Most commercial KBS deal **ONLY** with explicit knowledge. The often valuable but more difficult to obtain tacit knowledge is usually ignored

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz




# What Does KE Involve?

- ⚡ Collecting K from many sources (books, reports, systems & people)
- ⚡ Developing problem descriptions & solutions
- ⚡ Evaluating existing systems
- ⚡ Conducting interviews
- ⚡ Speaking the language of the problem domain
- ⚡ Thinking logically, symbolically, abstractly, creatively...

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

7




## What Does KE Involve?

- ⌞ Understanding:
  - ⌞ How various variables and rules are interrelated
  - ⌞ What data are input, and in what order
  - ⌞ The hierarchy of rules to consider
  - ⌞ How important and accurate the data items are (noise)
  - ⌞ Which data might be missing in a given situation
  - ⌞ Dealing with assumed or inferred data

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

8




## What Does KE Involve?

- ⌞ **Understanding:**
  - ⌞ How rule conflicts might be resolved
  - ⌞ What alternative strategies for problem solving might exist
  - ⌞ How to deal with uncertainty or confidence
  - ⌞ The levels of confidence an expert has in different rules or recommendations or data as predictors of outcomes

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

9




## What Does KE Involve?

- ⌞ Rapid prototyping
- ⌞ Developing incrementally
- ⌞ Knowledge base development
- ⌞ Inferencing processes
- ⌞ Separating knowledge from reasoning

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

10




## What Does KE Involve?

- ⌞ *Capturing and reusing* structured knowledge
- ⌞ *Capturing and sharing* lessons learned from practice
- ⌞ *Structuring and mapping* knowledge needed to enhance performance
- ⌞ *Capturing and reusing* unstructured knowledge

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

11



## What Does KE Involve? A Development Approach

**Planning**


- ⌞ Suitability of problem to heuristic solution
- ⌞ Availability of expertise
- ⌞ Management commitment, inter-organizational communication

**Design**

- ⌞ Modular or class-based approach
- ⌞ Domain knowledge modelling
- ⌞ Interfaces
- ⌞ Verification and validation

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

12




## Knowledge Elicitation

- ⌞ Examples of familiar tasks
- ⌞ Examples of tough cases
- ⌞ Unstructured interviews
- ⌞ Semi-structured interviews
- ⌞ Structured interviews
- ⌞ Protocol Analysis
- ⌞ Observational Protocols
- ⌞ Behavioral Protocols
- ⌞ Procedural Simulation

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

13




## Why is KE Important?

- ⚡ A process for implementing knowledge management systems - in order to have or obtain:
  - ⚡ Improved decision-making and action-taking
  - ⚡ Competitive advantage
  - ⚡ Management of intellectual assets
  - ⚡ Increased innovation
  - ⚡ Enhanced productivity
  - ⚡ Improved customer service, care and retention

© University of Auckland      www.cs.auckland.ac.nz/~ian/      ian@cs.auckland.ac.nz

14




## Where Do I Go to Find Out More?

- ⚡ IEEE Transactions on Knowledge and Data Engineering
- ⚡ IEEE Transactions on Systems, Man & Cybernetics
- ⚡ IEEE Expert Intelligent Systems & Their Applications
- ⚡ Journal of Intelligent Information Systems
- ⚡ American Association of AI ([www.aaai.org](http://www.aaai.org))
- ⚡ Lecture Handouts ⚡

© University of Auckland      www.cs.auckland.ac.nz/~ian/      ian@cs.auckland.ac.nz

15




## Knowledge Engineering

### Building a small knowledge system

- ⚡ A small system will probably:
  - ⚡ be rule-based
  - ⚡ contain less than 200 rules
  - ⚡ be intended to engage in a task similar to diagnosis
  - ⚡ use a backward-chaining strategy

© University of Auckland      www.cs.auckland.ac.nz/~ian/      ian@cs.auckland.ac.nz

16




## Six stages in building a small system:

1. Select a tool
2. Identify the problem, & analyze the knowledge involved.
3. Design the system on paper
4. Use the tool to develop a prototype; i.e. write a knowledge base, with appropriate inference engine and user interface attached
5. Test, revise and expand the system until it does what you want it to
6. Maintain and update the system as necessary

© University of Auckland      www.cs.auckland.ac.nz/~ian/      ian@cs.auckland.ac.nz

17




## Selecting the tool:

- ⚡ Examples of commercially -available expert system tools: Xi+, Crystal, M1. (all popular in the 1980s). Leonardo XpertRule (more recent)
- ⚡ **Case-based reasoning** systems are also popular.

© University of Auckland      www.cs.auckland.ac.nz/~ian/      ian@cs.auckland.ac.nz

18



## Selecting the problem

- ⚡ Should typically take about 30 minutes for a person to solve
- ⚡ Should not involve physical examination: should be solvable by a phone conversation.
- ⚡ Knowledge involved should be simple rules, and simple calculations, not large amounts of maths.
- ⚡ Should be a choice of no more than a few dozen faults or diagnoses

© University of Auckland      www.cs.auckland.ac.nz/~ian/      ian@cs.auckland.ac.nz

19

## Designing the system on paper

- Draw flow diagrams of the consultation that takes place
- It may be possible to draw up a matrix, with conditions as the columns, and conclusions as the rows (this is one of the **intermediate representations** mentioned below);
- Write draft rules, if there is a matrix, each line will be a rule

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

20

## Designing the system on paper

Warning light	temp	vibration	diagnosis
on	125	high	Low coolant
on	95	high	Low oil
off	125	normal	Broken thermostat

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

21

## Designing the system on paper

- Write appropriate questions, to satisfy the conditions of the rules, and specify ranges of answers
- Choose a top-level goal
- It may help to draw a decision tree through the rules starting from the top-level goal as the root
- The leaves of the tree are diagnoses

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

22

## Designing the system on paper

```

graph TD
    A[I'm Hungry] --> B[VEGETARIAN?]
    B -- yes --> C[VEGAN?]
    B -- no --> D[LIKE FISH?]
    C -- yes --> E[Tofu & Nut Loaf]
    C -- no --> F[Egg & Spinach Pie]
    D -- yes --> G[Cod & Chips]
    D -- no --> H[Steak & Chips]
  
```

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

23

## Developing a prototype

- Assuming an expert is involved, he/she should work with the knowledge engineer to build this.

It is important that the expert should be committed to the task – a project champion.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz


24

## Testing, revising & expanding the system

- At this stage, more advanced features of the tool may be used - e.g. facility for the user to volunteer information without waiting for questions
- Textual explanations can added
- Real users should try out the system, and it should be modified on the basis of their experiences.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

25




## Maintaining & updating the system

- ⚡ Every time a situation occurs which the system cannot cope with, it should be recorded, and the addition of a rule to the system should be considered
- ⚡ An individual must be responsible for maintenance; their job is to change the knowledge base in response to changes in the external world

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

26




## Building a large expert system

- ⚡ Stages:
  - ⚡ Select the domain and the task on which the system is to advise
  - ⚡ Select the tool which will be used to build the system
  - ⚡ Develop a prototype
  - ⚡ Expand this into a complete system
  - ⚡ Evaluate the system
  - ⚡ Integrate the system into the organisation's work
  - ⚡ Maintain the system

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

27




## Differences between building large & small KBS

- ⚡ Built by a team rather than by an individual
- ⚡ One or more knowledge engineers, one or more experts (to provide the knowledge), one or more managers.
- ⚡ Tackle larger tasks - typically a task which would take a human expert something over 3 hours to solve. Might contain 1000s of rules

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

28




## Differences between building a large system and a small system

- ⚡ Have larger hardware requirements
- ⚡ Have large initial development costs - this means that the project should promise a large R.O.I.
- ⚡ Have a longer development period - may be several person-years, for both the knowledge engineers and the experts

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

29




## Stage 1: Selecting the domain and the task.

- ⚡ Choosing the right task is crucial to the success of the project (the system must be technically feasible).
- ⚡ Expert systems are good at tasks in fairly narrow domains, where the knowledge can be documented or taught, but which don't require common sense or sensory discrimination.
- ⚡ It must be possible to evaluate the outcome of the task.
- ⚡ The costs and benefits need to be calculated. An expert system working in an inappropriate domain may cost more than it saves.
- ⚡ The system will have to be acceptable to the end users.
- ⚡ An expert must be found who is prepared to contribute the required expertise.
- ⚡ A development plan must be drawn up, specifying goals for the system to achieve.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

30




## Stage 2: Selecting the tool

- ⚡ KBS shells capable of building large-scale systems with sophisticated features are available: e.g. ART\*Enterprise.
- ⚡ Use an AI language (Prolog LISP CLIPS).
- ⚡ One should choose a simpler tool that provides appropriate knowledge representation and inference strategies.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

31




## Stage 3: Developing a prototype

- ⚡ The expert and knowledge engineer will work together to build this. It is important that the expert should become committed to the task.
- ⚡ The prototype should test:
  - ⚡ assumptions about facts, relationships and inferences;
  - ⚡ the adequacy of the tool used to build the system.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

32




## Stage 3: Developing a prototype

- ⚡ At this stage, a detailed design for the complete expert system is also developed.
- ⚡ Performance criteria will be established - e.g. "The system will be expected to come to the same conclusion as the expert 95% of the time"
- ⚡ As analysis proceeds the knowledge elicited is built into the prototype system giving the expert the chance to observe the system working, and criticise the knowledge & reasoning it displays.
- ⚡ The system is improved by progressive refinement.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

33




## Stage 3: Developing a prototype

- ⚡ When the prototype is built, the knowledge engineer and the expert should be able to decide:
  - ⚡ whether the knowledge representation formalism is adequate
  - ⚡ whether the tool is adequate
  - ⚡ the overall design of the full-scale system
  - ⚡ the approximate size (e.g. no. of rules) in the full-scale system.
  - ⚡ the performance criteria for the full-scale system.
  - ⚡ Any major design changes (e.g. choosing a different tool) should be made at this stage.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

34




## Stage 4: Developing the complete system

- ⚡ Once the prototype system has been approved:
  - ⚡ The knowledge base may need to be rebuilt, with different objects and attributes.
  - ⚡ The process of progressive refinement, described above, continues.
  - ⚡ The 'depth' of knowledge in the knowledge base is increased by adding many more rules, and other knowledge structures.

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

35




## Stage 4: Developing the complete system

- ⚡ The 'breadth' of knowledge in the knowledge base may be increased by adding other tasks that the user is accustomed to provide advice on
- ⚡ The user interface is tailored to the contents of the knowledge base
- ⚡ At this stage, the expert should become skilled at entering rules without the knowledge engineer's assistance, and monitoring the effects that these new rules will have
- ⚡ The expert should be able to polish, elaborate and maintain the system - the first stage of technology transfer

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

36




## Stage 5: Evaluating the complete system

- ⚡ Once the system is complete:
  - ⚡ It can be tested, using the performance criteria established earlier
  - ⚡ Other experts can be invited to try the system, and assess its performance

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

37




## Stage 6: Integrating the system into the organizations work

- ⚡ Technology transfer: ensuring that the experts, users and systems personnel can and will use and maintain the expert system. This involves:
  - ⚡ Convincing the experts that the system is useful to them personally
  - ⚡ They should be shown that it frees them from boring tasks.
  - ⚡ Communicating with and providing support for other people involved with the system
  - ⚡ Interfacing the system with databases, instruments, or other systems

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

38




## Stage 6: Integrating the system into the organizations work

- ⚡ If the KBS can directly access an instrument or a database, it doesn't need to ask the user. This makes it faster and more user-friendly
- ⚡ Hardware may need to be modified if the system is to be installed in a difficult environment.
- ⚡ It may be appropriate to rewrite the software in a conventional computer language
- ⚡ Advantages: speed, and portability between machines
- ⚡ Disadvantages: flexibility is lost: the system can't easily be modified or maintained

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

39




## Stage 7: Maintaining the system.

- ⚡ Some expert should be made responsible for keeping the knowledge base current, adding/changing rules (and other knowledge structures) when necessary
- ⚡ This stage is not optional

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

40




## Scaling up a small expert system, in order to build a large one

### Advantages

- ⚡ The work involved in building a prototype is avoided: the existing system is the prototype
- ⚡ The small system has probably been more thoroughly tested than a prototype could be
- ⚡ The viability of the project is clearer than if the knowledge engineer were starting from scratch
- ⚡ It is easier to convince the end-users that the final system will be useful to them

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz

41



## Scaling up a small expert system, in order to build a large one

### Disadvantages:

- ⚡ There may be limits on the number of rules the tool used to build the small system can handle.
- ⚡ Performance, especially speed of response, may be unacceptably downgraded as the knowledge base expands
- ⚡ Some desirable form of knowledge representation (e.g. frames, model-based reasoning) may not be supported by the tool
- ⚡ Some desirable feature of the user interface (e.g. graphical data display) may not be supported by the tool.
- ⚡ The wrong objects and attributes may have been chosen in the first place, and simply scaling up the system won't cure this
- ⚡ As more rules are added, the chance that there will be some unrealised interaction between the rules increases

© University of Auckland    www.cs.auckland.ac.nz/~ian/    ian@cs.auckland.ac.nz