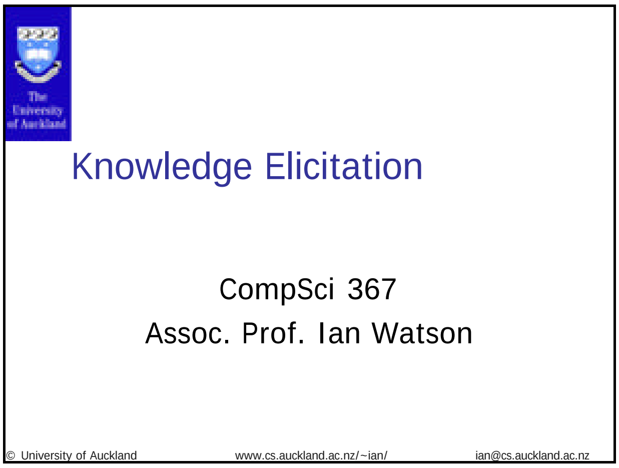
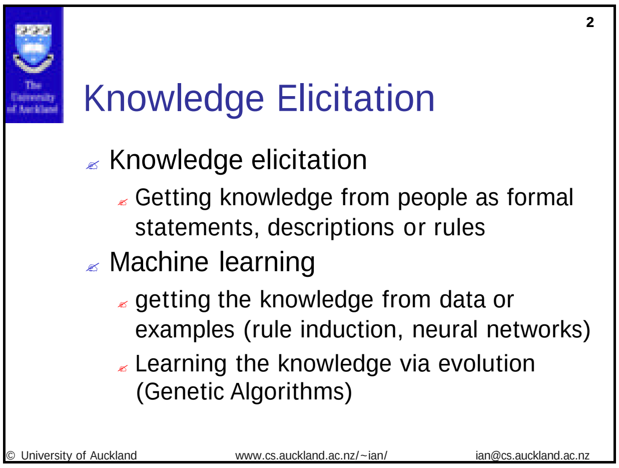


Knowledge Elicitation

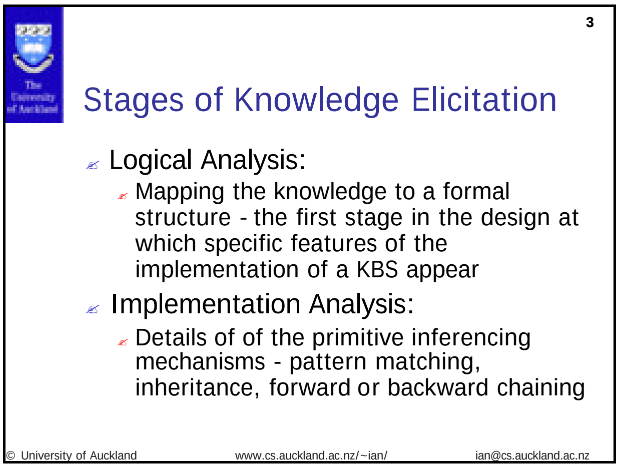
CompSci 367

Assoc. Prof. Ian Watson



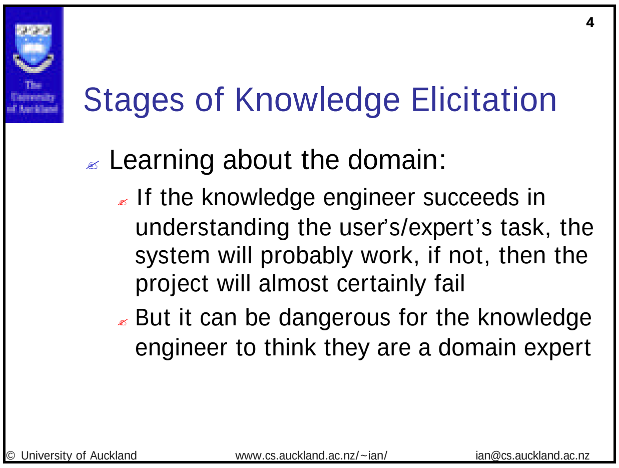
Knowledge Elicitation

- Knowledge elicitation
  - Getting knowledge from people as formal statements, descriptions or rules
- Machine learning
  - getting the knowledge from data or examples (rule induction, neural networks)
  - Learning the knowledge via evolution (Genetic Algorithms)



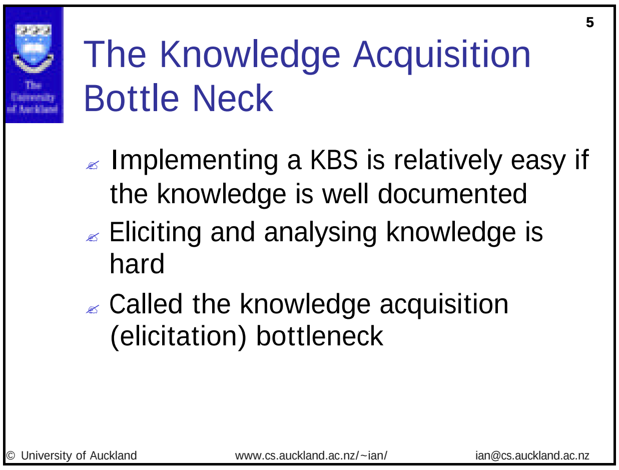
Stages of Knowledge Elicitation

- Logical Analysis:
  - Mapping the knowledge to a formal structure - the first stage in the design at which specific features of the implementation of a KBS appear
- Implementation Analysis:
  - Details of of the primitive inferencing mechanisms - pattern matching, inheritance, forward or backward chaining



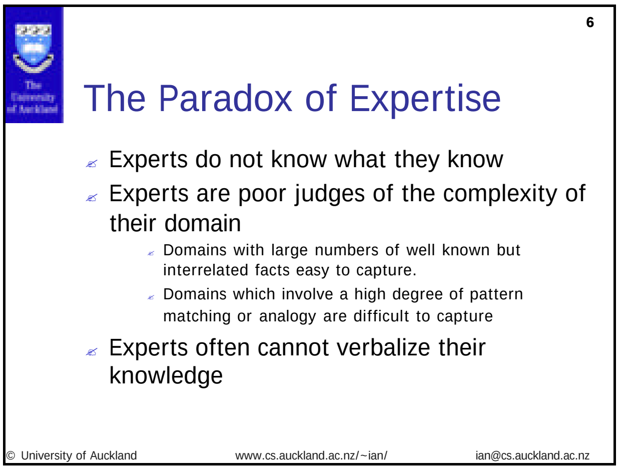
Stages of Knowledge Elicitation

- Learning about the domain:
  - If the knowledge engineer succeeds in understanding the user's/expert's task, the system will probably work, if not, then the project will almost certainly fail
  - But it can be dangerous for the knowledge engineer to think they are a domain expert



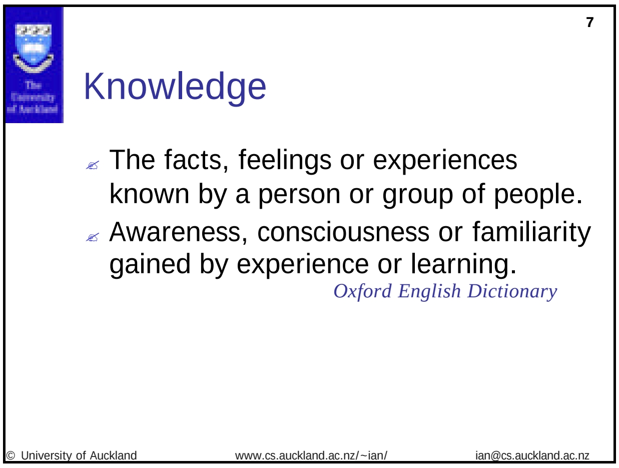
The Knowledge Acquisition Bottle Neck

- Implementing a KBS is relatively easy if the knowledge is well documented
- Eliciting and analysing knowledge is hard
- Called the knowledge acquisition (elicitation) bottleneck



The Paradox of Expertise

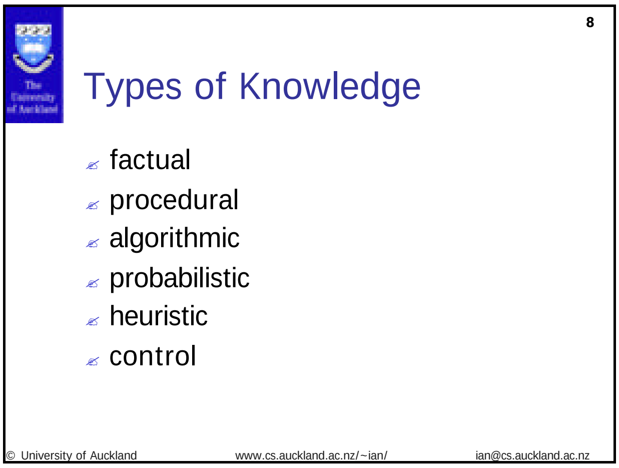
- Experts do not know what they know
- Experts are poor judges of the complexity of their domain
  - Domains with large numbers of well known but interrelated facts easy to capture.
  - Domains which involve a high degree of pattern matching or analogy are difficult to capture
- Experts often cannot verbalize their knowledge



Knowledge

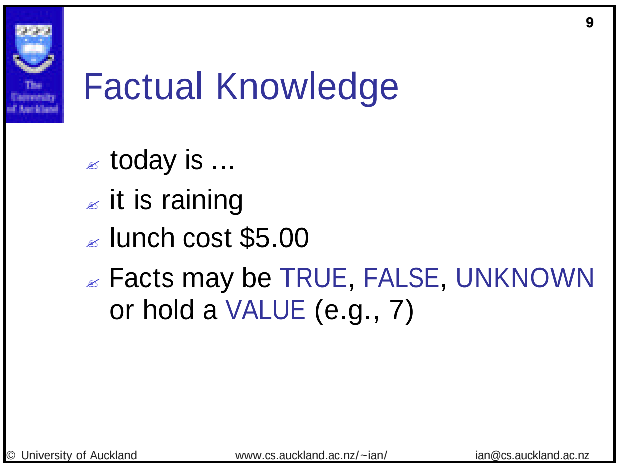
- The facts, feelings or experiences known by a person or group of people.
- Awareness, consciousness or familiarity gained by experience or learning.

Oxford English Dictionary



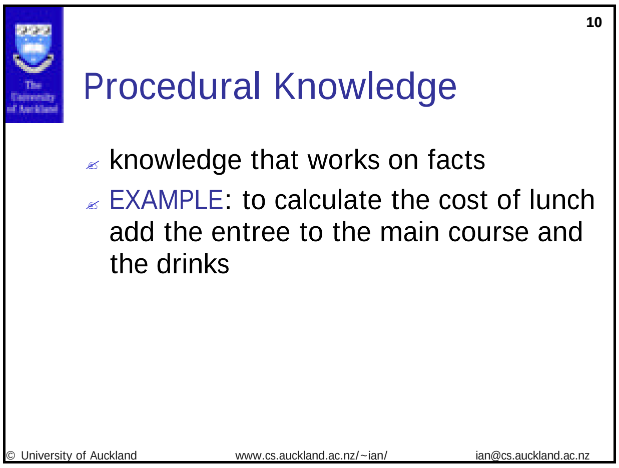
Types of Knowledge

- factual
- procedural
- algorithmic
- probabilistic
- heuristic
- control



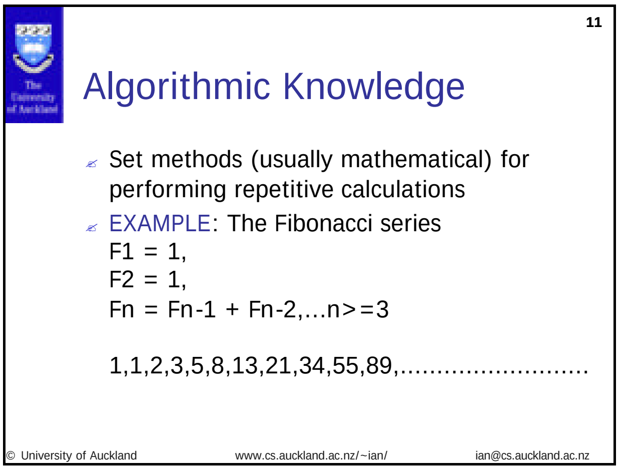
Factual Knowledge

- today is ...
- it is raining
- lunch cost $5.00
- Facts may be TRUE, FALSE, UNKNOWN or hold a VALUE (e.g., 7)



Procedural Knowledge

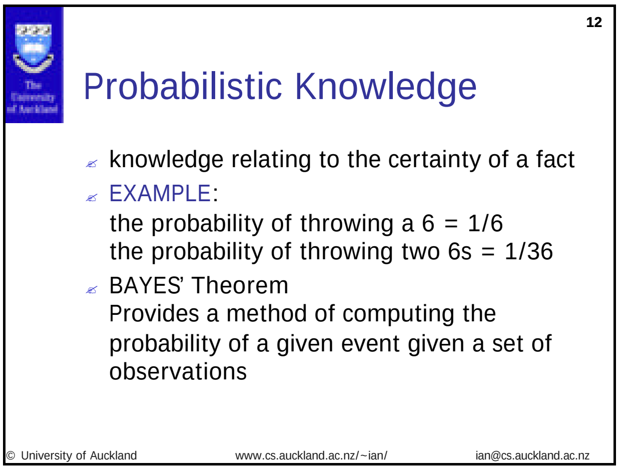
- knowledge that works on facts
- EXAMPLE: to calculate the cost of lunch add the entree to the main course and the drinks



Algorithmic Knowledge

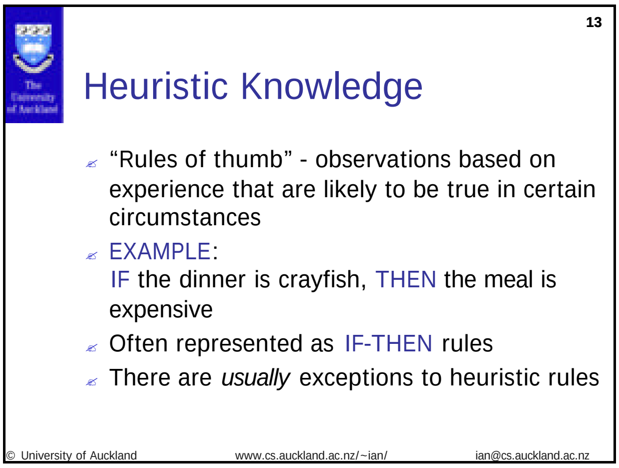
- Set methods (usually mathematical) for performing repetitive calculations
- EXAMPLE: The Fibonacci series
  F1 = 1,
  F2 = 1,
  Fn = Fn-1 + Fn-2,...n>=3

  1,1,2,3,5,8,13,21,34,55,89,.........................



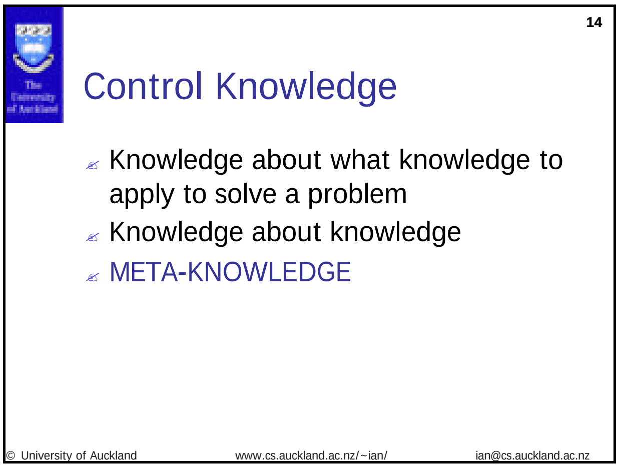
Probabilistic Knowledge

- knowledge relating to the certainty of a fact
- EXAMPLE:
  the probability of throwing a 6 = 1/6
  the probability of throwing two 6s = 1/36
- BAYES' Theorem
  Provides a method of computing the probability of a given event given a set of observations



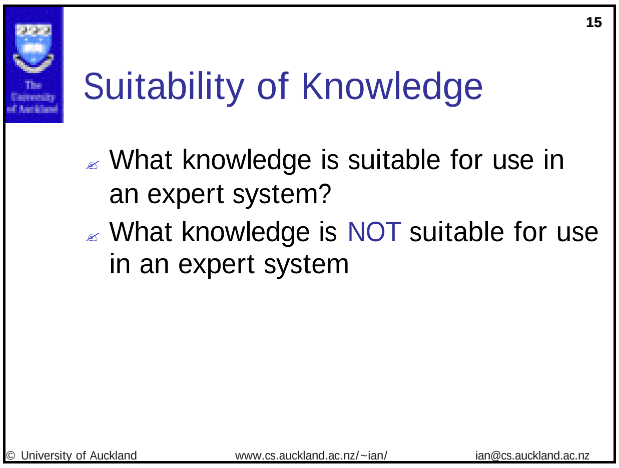
Heuristic Knowledge

- “Rules of thumb” - observations based on experience that are likely to be true in certain circumstances
- EXAMPLE:
  IF the dinner is crayfish, THEN the meal is expensive
- Often represented as IF-THEN rules
- There are *usually* exceptions to heuristic rules



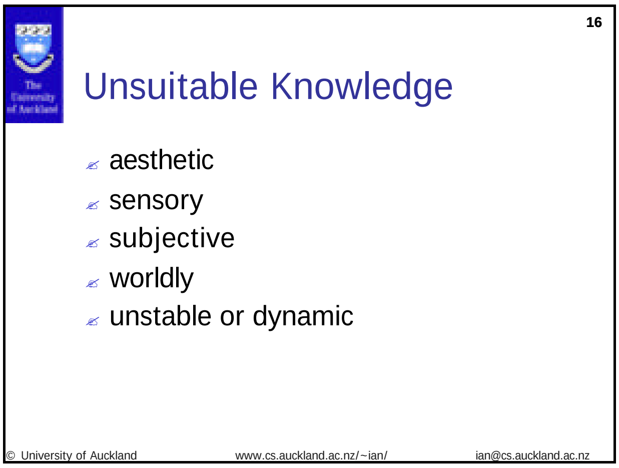
Control Knowledge

- Knowledge about what knowledge to apply to solve a problem
- Knowledge about knowledge
- META-KNOWLEDGE



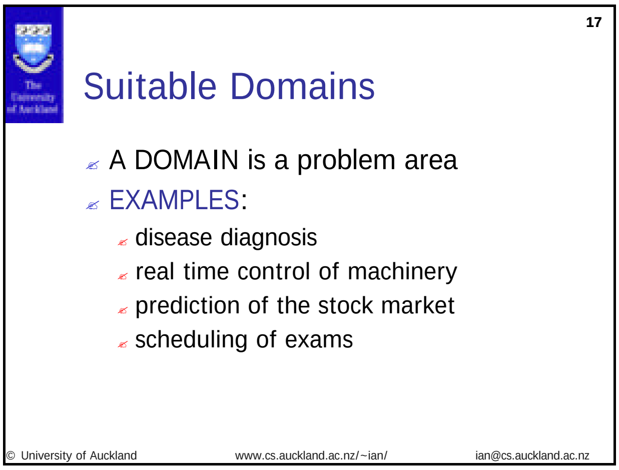
Suitability of Knowledge

- What knowledge is suitable for use in an expert system?
- What knowledge is NOT suitable for use in an expert system



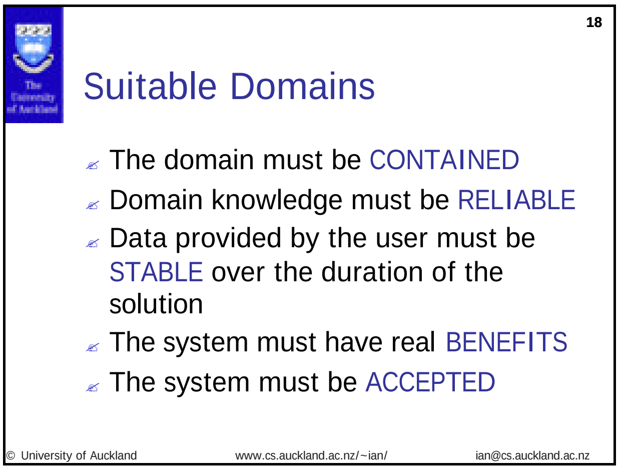
Unsuitable Knowledge

- aesthetic
- sensory
- subjective
- worldly
- unstable or dynamic



Suitable Domains

- A DOMAIN is a problem area
- EXAMPLES:
  - disease diagnosis
  - real time control of machinery
  - prediction of the stock market
  - scheduling of exams

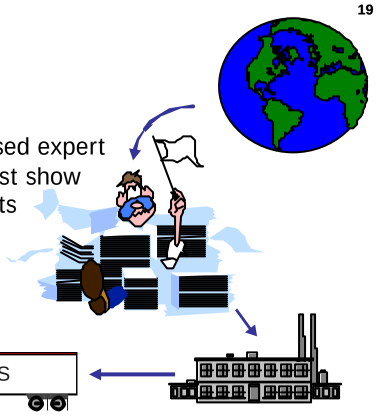


Suitable Domains

- The domain must be CONTAINED
- Domain knowledge must be RELIABLE
- Data provided by the user must be STABLE over the duration of the solution
- The system must have real BENEFITS
- The system must be ACCEPTED

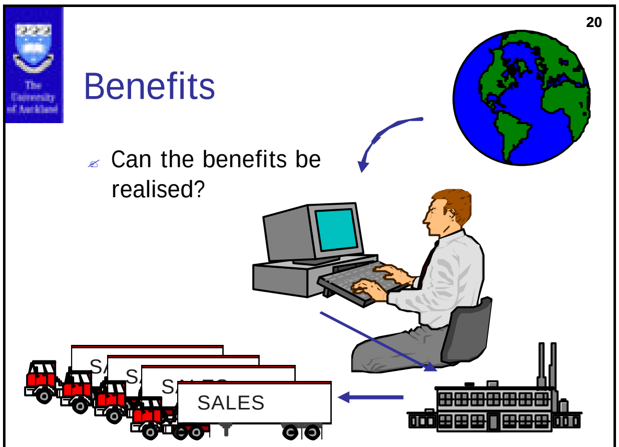
Benefits

- The proposed expert system must show real benefits



Benefits

- Can the benefits be realised?



When to apply KBS

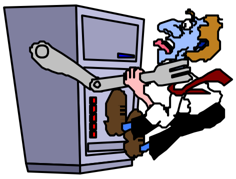
- when knowledge is rare
- when knowledge is valuable - if staff turn over is high and training expensive (e.g., customer service help desks)
- when knowledge is critical - if decisions have to be double or triple checked (safety critical)
- when the problem needs to be better understood

Knowledge Elicitation

- A knowledge engineer elicits knowledge from:
- books, reports, manuals, etc.
- computer systems
- organisational systems
- and people

K-Elicitation Techniques

- unstructured interviews
- semi-structured interviews
- structured interviews
- protocol analysis
- other techniques



Concept Elicitation

Learning the Terminology

- Initial Terminology
  - Card Sorting
  - Clustering
  - Vocabulary from discussions and protocol analysis
- Techniques from Personal Construct Theory
  - 20 questions
  - Card Sort
  - Laddered grid
  - Repertory grids

Task Analysis

- Who are the real users.
  - Never believe what managers or supervisors tell you about what is done
  - Talk to the staff who actually do the work
- WIN-WIN situations - To make a successful applications, all users/stake holders must benefit.
  - All users are stake holders!
  - It may require extra functionality to provide extra benefits to particular stake holders

Unstructured Interviews

- no agenda
- good for getting an overview of the problem
- good for establishing a relationship

Semi-Structured Interviews

- use an agenda
- use a tape recorder or video camera
- keep returning to agenda items
- write a summary of the interview
- ask the expert to comment on summary

Structured Interview

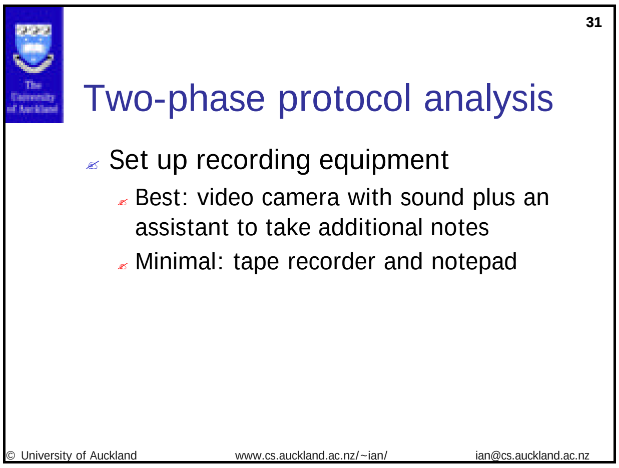
- repeat a few questions over and over
  - WHAT
  - WHEN
  - WHAT ELSE
  - WHEN NOT
  - WHY

Two-phase protocol analysis

- Preliminary Study
  - Read technical material relevant to task domain
  - Observe experts & users at work
  - Get a feel for typical, easy and hard tasks

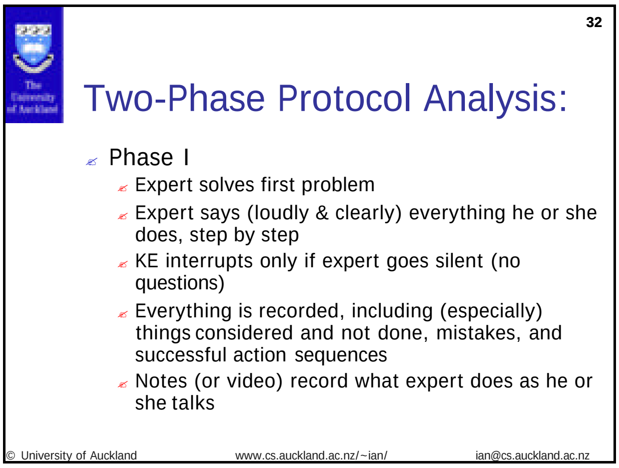
Two-phase protocol analysis

- Task Selection
  - Select a piece of the domain (deep and narrow) in consultation with the expert
  - Acquire typical problems (ones solved in the past totally contained in the domain)
  - Rank problems according to expert's estimate of complexity



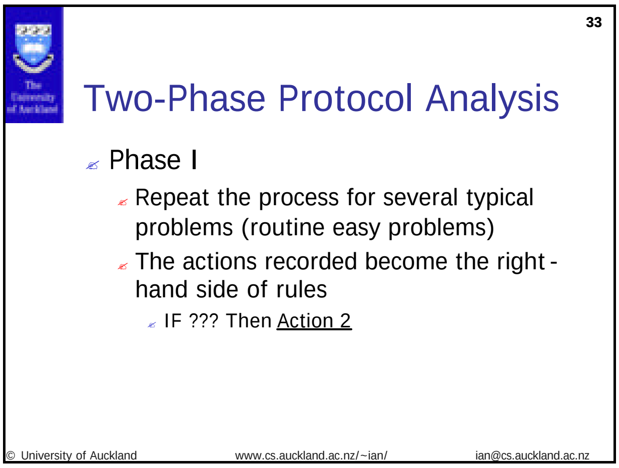
Two-phase protocol analysis

- Set up recording equipment
  - Best: video camera with sound plus an assistant to take additional notes
  - Minimal: tape recorder and notepad



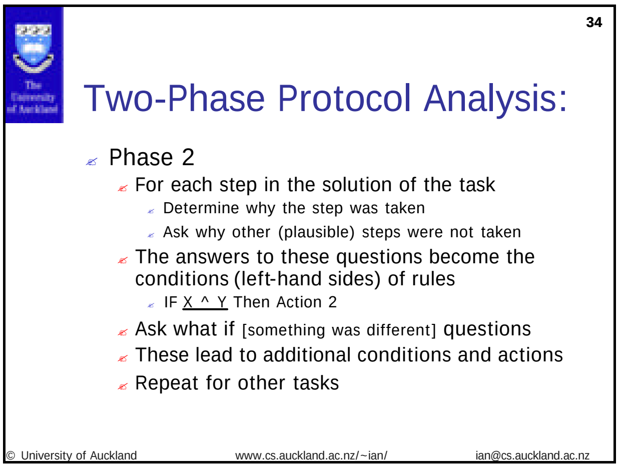
Two-Phase Protocol Analysis:

- Phase I
  - Expert solves first problem
  - Expert says (loudly & clearly) everything he or she does, step by step
  - KE interrupts only if expert goes silent (no questions)
  - Everything is recorded, including (especially) things considered and not done, mistakes, and successful action sequences
  - Notes (or video) record what expert does as he or she talks



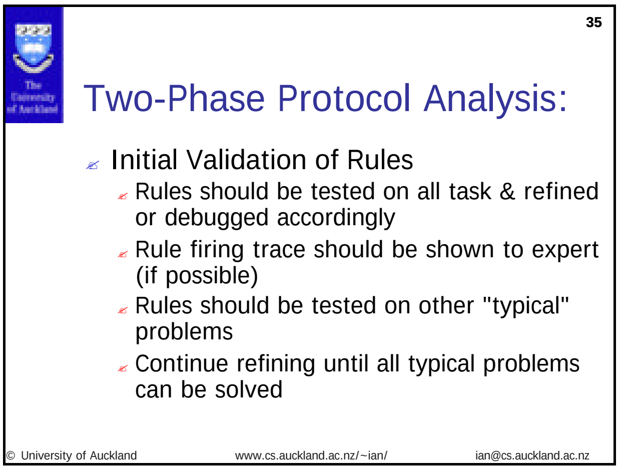
Two-Phase Protocol Analysis

- Phase I
  - Repeat the process for several typical problems (routine easy problems)
  - The actions recorded become the right-hand side of rules
    - IF ??? Then Action 2



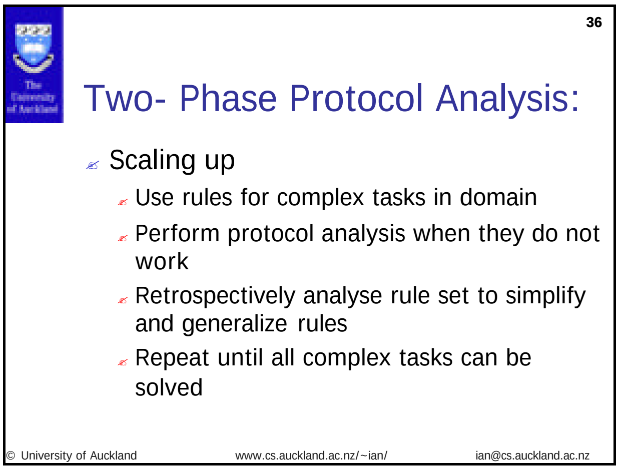
Two-Phase Protocol Analysis:

- Phase 2
  - For each step in the solution of the task
    - Determine why the step was taken
    - Ask why other (plausible) steps were not taken
  - The answers to these questions become the conditions (left-hand sides) of rules
    - IF X ^ Y Then Action 2
  - Ask what if [something was different] questions
  - These lead to additional conditions and actions
  - Repeat for other tasks



Two-Phase Protocol Analysis:

- Initial Validation of Rules
  - Rules should be tested on all task & refined or debugged accordingly
  - Rule firing trace should be shown to expert (if possible)
  - Rules should be tested on other "typical" problems
  - Continue refining until all typical problems can be solved



Two- Phase Protocol Analysis:

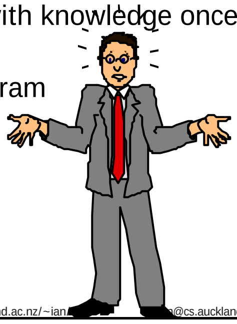
- Scaling up
  - Use rules for complex tasks in domain
  - Perform protocol analysis when they do not work
  - Retrospectively analyse rule set to simplify and generalize rules
  - Repeat until all complex tasks can be solved

Other Methods

- 1001 techniques from psychology, cognitive science, artificial intelligence, etc...
- repertory grids
- laddered hierarchies
- card sorting
- etc...

Knowledge Modeling

- what do we do with knowledge once we've got it?
- put it into a program
- model it



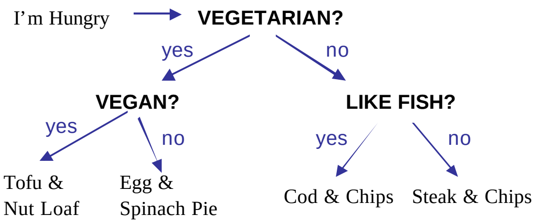
Knowledge Modeling

- models of knowledge or intermediate knowledge representations help to:
- identify errors
- identify omissions
- identify contradictions
- reduce the cost of re-writes

Decision Trees

- good for diagnostic & classification problems
- experts can understand them
- easy to create
- translate into rule-based systems easily (a path from root to leaf is a rule)
- can become difficult to edit if large (solution - create a forest of trees)

Decision Trees



I'm Hungry → VEGETARIAN?

VEGETARIAN? yes → VEGAN?

VEGETARIAN? no → LIKE FISH?

VEGAN? yes → Tofu & Nut Loaf

VEGAN? no → Egg & Spinach Pie

LIKE FISH? yes → Cod & Chips

LIKE FISH? no → Steak & Chips

Decision Tables

- good for modeling selection criteria
- experts understand them
- can consider all possible combinations of criteria
- translate into rules easily (a row is a rule)
- but can become unwieldy

Decision Tables

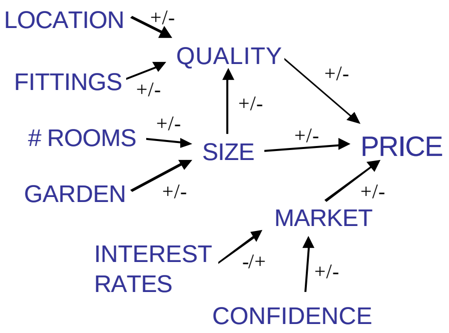
AGE	GRADES	EXPERIENCE	ACTION
18	B+	none	**accept**
22	C	none	**reject**
30	C	10 years	**interview**
18	B	none	**interview**

Inference Nets

- good for modeling causal knowledge
- can be understood by experts
- translate into rules and OOPS well

- Note: also called:
  - Influence diagrams
  - Belief networks

Inference Nets


LOCATION
+/-
QUALITY
FITTINGS
+/-
+/-
+/-
# ROOMS
+/-
SIZE
+/-
PRICE
GARDEN
+/-
+/-
MARKET
INTEREST RATES
-/+
+/-
CONFIDENCE