



# HUMAN-COMPUTER INTERACTION

THIRD  
EDITION

DIX  
FINLAY  
ABOWD  
BEALE



## chapter 15

## task models

# What is Task Analysis?

Methods to analyse people's jobs:

- what people do
- what things they work with
- what they must know

# An Example

- in order to clean the house
  - get the vacuum cleaner out
  - fix the appropriate attachments
  - clean the rooms
  - when the dust bag gets full, empty it
  - put the vacuum cleaner and tools away
- must know about:
  - vacuum cleaners, their attachments, dust bags, cupboards, rooms etc.

# Approaches to task analysis

- Task decomposition
  - splitting task into (ordered) subtasks
- Knowledge based techniques
  - what the user knows about the task and how it is organised
- Entity/object based analysis
  - relationships between objects, actions and the people who perform them
- lots of different notations/techniques

# general method

- observe
- collect unstructured lists of words and actions
- organize using notation or diagrams

# Differences from other techniques

## **Systems analysis**      **vs.**      **Task analysis**

system design - focus - the user

## **Cognitive models**      **vs.**      **Task analysis**

internal mental state - focus - external actions

practiced 'unit' task - focus - whole job

# Task Decomposition

## Aims:

- describe the actions people do
- structure them within task subtask hierarchy
- describe order of subtasks

## Variants:

- Hierarchical Task Analysis (HTA)

  - most common

- CTT (CNUCE, Pisa)

  - uses LOTOS temporal operators

# Textual HTA description

## Hierarchy description ...

- 0. in order to clean the house
  - 1. get the vacuum cleaner out
  - 2. get the appropriate attachment
  - 3. clean the rooms
    - 3.1. clean the hall
    - 3.2. clean the living rooms
    - 3.3. clean the bedrooms
  - 4. empty the dust bag
  - 5. put vacuum cleaner and attachments away

... and plans

Plan 0: do 1 - 2 - 3 - 5 in that order. when the dust bag gets full do 4

Plan 3: do any of 3.1, 3.2 or 3.3 in any order depending  
on which rooms need cleaning

**N.B. only the plans denote order**



# Generating the hierarchy

- 1 get list of tasks
- 2 group tasks into higher level tasks
- 3 decompose lowest level tasks further

## Stopping rules

How do we know when to stop?

Is “empty the dust bag” simple enough?

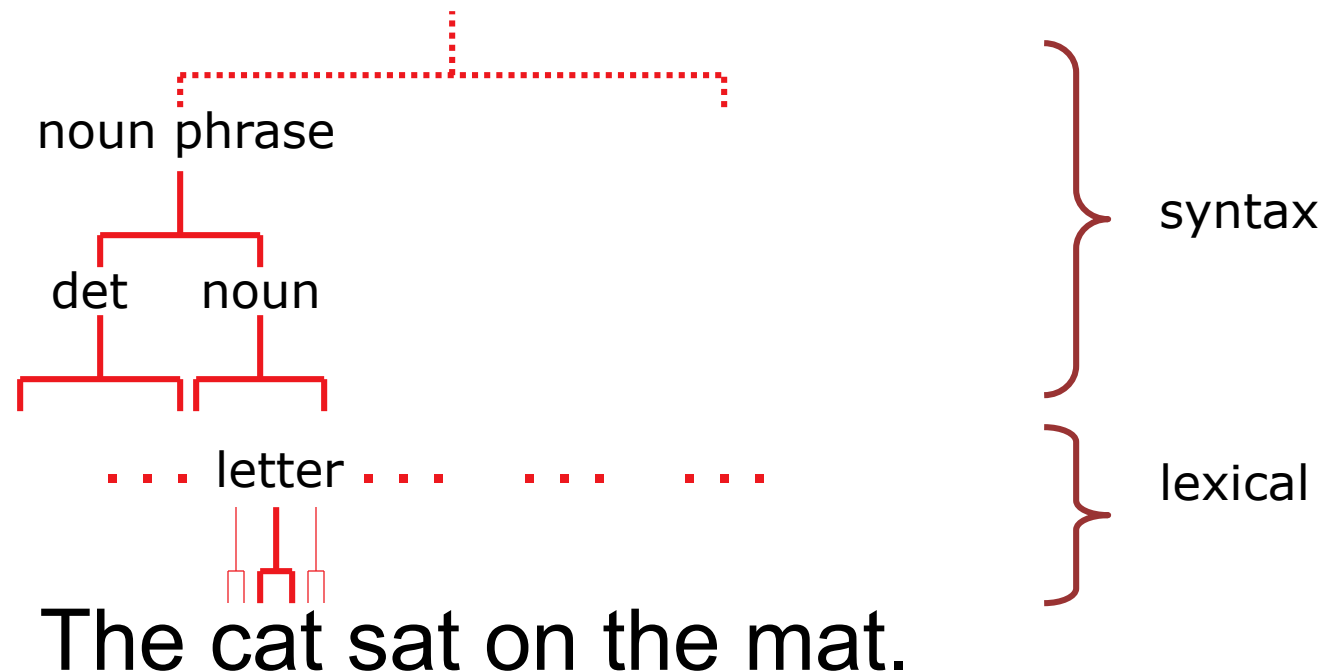
Purpose: expand only relevant tasks

Motor actions: lowest sensible level

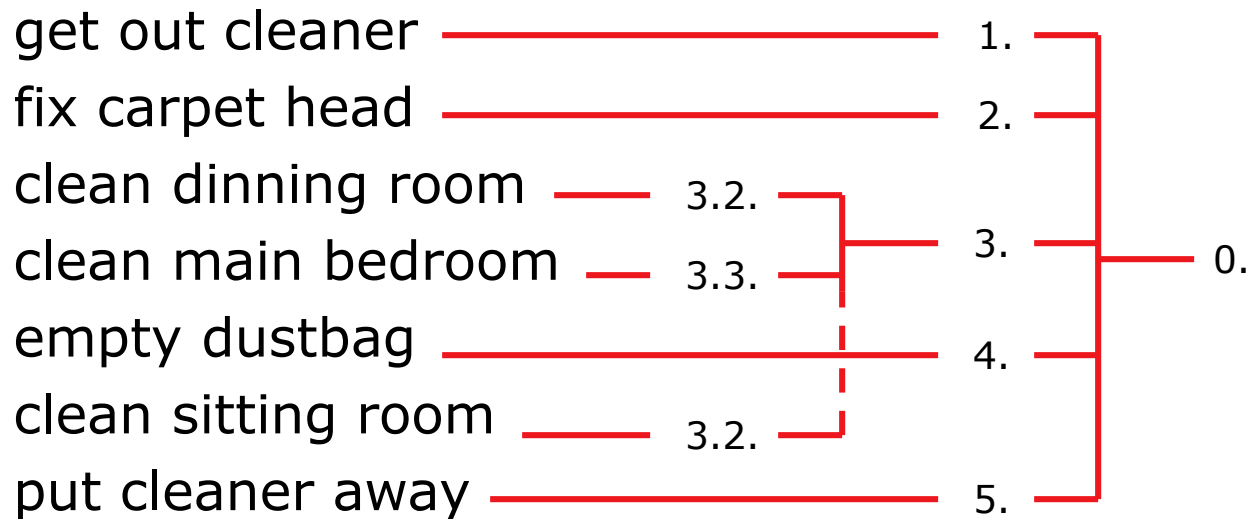
# Tasks as explanation

- imagine asking the user the question:  
**what are you doing now?**
- for the same action the answer may be:
  - typing ctrl-B
  - making a word bold
  - emphasising a word
  - editing a document
  - writing a letter
  - preparing a legal case

- can parse sentence into letters, nouns, noun phrase, etc.

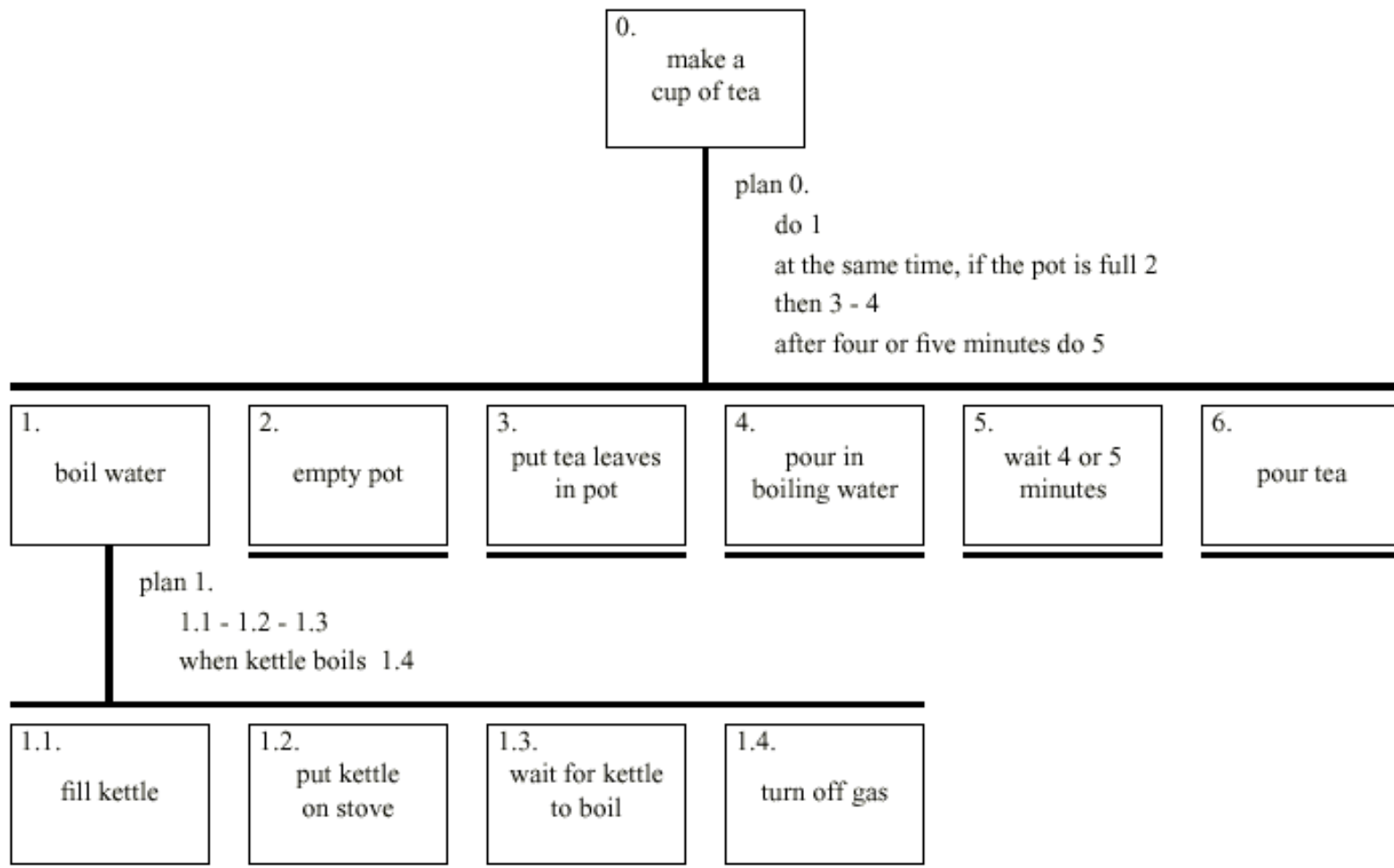


# parse scenario using HTA



- 0. in order to clean the house
  - 1. get the vacuum cleaner out
  - 2. get the appropriate attachment
  - 3. clean the rooms
    - 3.1. clean the hall
    - 3.2. clean the living rooms
    - 3.3. clean the bedrooms
  - 4. empty the dust bag
  - 5. put vacuum cleaner and attachments away

# Diagrammatic HTA



# Refining the description

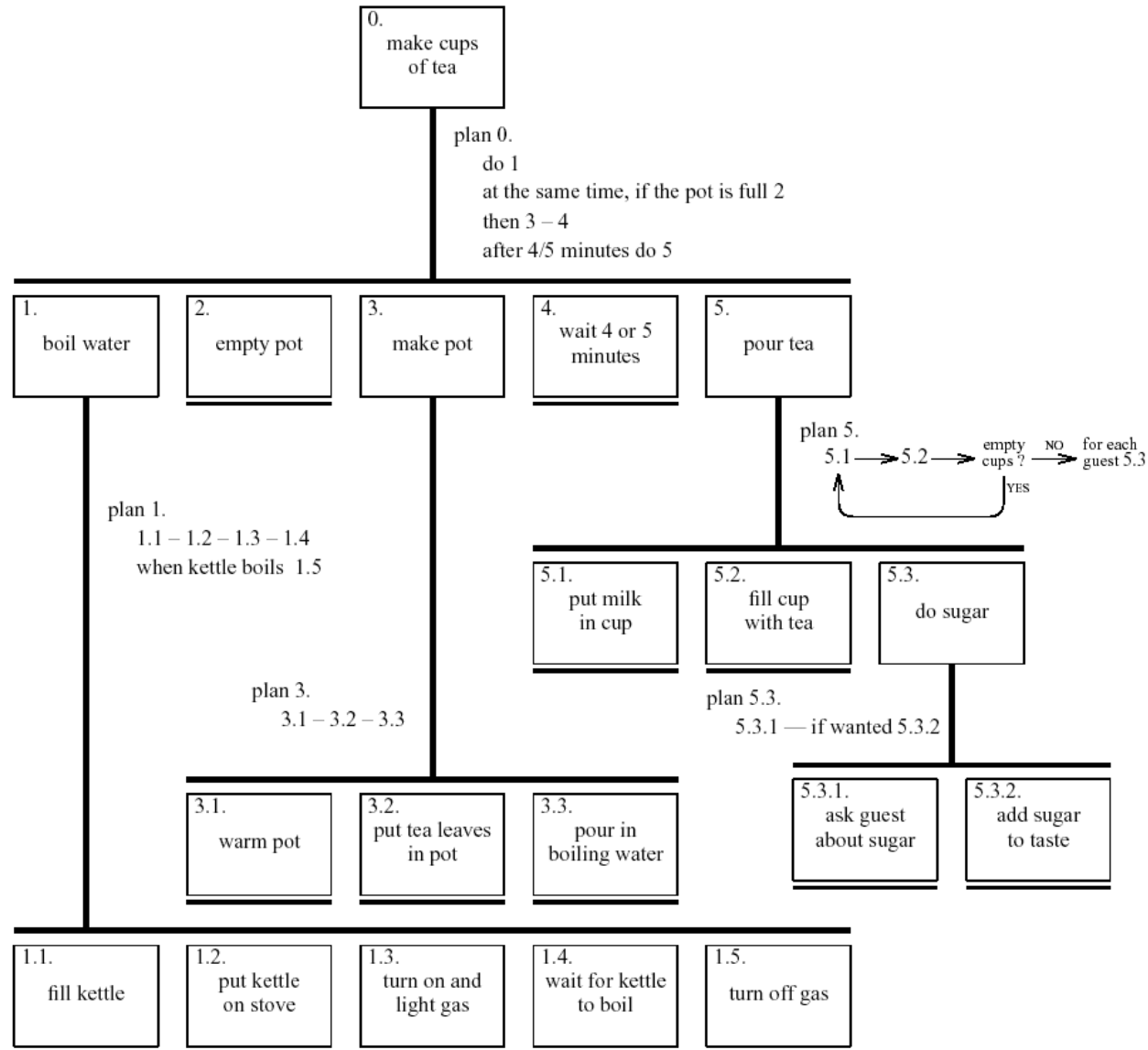
Given initial HTA (textual or diagram)

How to check / improve it?

Some heuristics:

- |                |  |
|----------------|--|
| paired actions | e.g., where is 'turn on gas'                 |
| restructure    | e.g., generate task 'make pot'               |
| balance        | e.g., is 'pour tea' simpler than making pot? |
| generalise     | e.g., make one cup ..... or more             |

# Refined HTA for making tea



# Types of plan

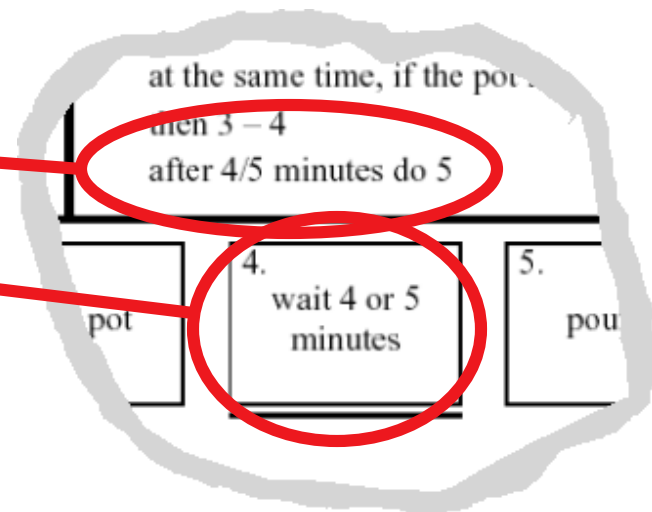
- |                 |   |
|-----------------|---|
| fixed sequence  | - 1.1 then 1.2 then 1.3                       |
| optional tasks  | - if the pot is full 2                        |
| wait for events | - when kettle boils 1.4                       |
| cycles          | - do 5.1 5.2 while there are still empty cups |
| time-sharing    | - do 1; at the same time ...                  |
| discretionary   | - do any of 3.1, 3.2 or 3.3 in any order      |
| mixtures        | - most plans involve several of the above     |





# waiting ...

- is waiting part of a plan?  
... or a task?
- generally
  - task – if 'busy' wait
    - you are actively waiting
  - plan – if end of delay is the event
    - e.g. "when alarm rings", "when reply arrives"
- in this example ...
  - perhaps a little redundant ...
  - TA not an exact science



see chapter 19 for more on delays!

# Knowledge Based Analyses

Focus on:

Objects – used in task

Actions – performed

+ Taxonomies –  
represent levels of abstraction

# Knowledge-Based Example ...

motor controls

steering *steering wheel, indicators*

engine/speed

direct *ignition, accelerator, foot brake*

gearing *clutch, gear stick*

lights

external *headlights, hazard lights*

internal *courtesy light*

wash/wipe

wipers *front wipers, rear wipers*

washers *front washers, rear washers*

heating *temperature control, air direction,  
fan, rear screen heater*

parking *hand brake, door lock*

radio *numerous!*

# Task Description Hierarchy

Three types of branch point in taxonomy:

- XOR – normal taxonomy  
object in one and only one branch
- AND – object must be in both  
multiple classifications
- OR – weakest case  
can be in one, many or none

wash/wipe AND

function XOR

wipe	<i>front wipers, rear wipers</i>
wash	<i>front washers, rear washers</i>

position XOR

front	<i>front wipers, front washers</i>
rear	<i>rear wipers, rear washers</i>

# Larger TDH example

```
kitchen item AND
/____shape XOR
/      |____dished  mixing bowl, casserole, saucepan,
/      |            soup bowl, glass
/      |____flat    plate, chopping board, frying pan
/____function OR
{____preparation    mixing bowl, plate, chopping board
{____cooking        frying pan, casserole, saucepan
{____dining XOR
    |____for food    plate, soup bowl, casserole
    |____for drink   glass
```

N.B. ` / | { ' used for branch types.

# More on TDH

Uniqueness rule:

- can the diagram distinguish all objects?

e.g., plate is:

```
kitchen item/shape(flat)/function{preparation,dining(for food)}/
```

nothing else fits this description

Actions have taxonomy too:

```
kitchen job OR
```

```
|____ preparation beating, mixing
```

```
|____ cooking frying, boiling, baking
```

```
|____ dining pouring, eating, drinking
```

# Abstraction and cuts

After producing detailed taxonomy  
    'cut' to yield abstract view

That is, ignore lower level nodes  
    e.g. cutting above shape and below dining, plate becomes:  
        kitchen item/function{preparation,dining}/

This is a term in Knowledge Representation Grammar (KRG)

These can be more complex:  
    e.g. 'beating in a mixing bowl' becomes:  
        kitchen job(preparation) *using a*  
            kitchen item/function{preparation}/

# Entity-Relationship Techniques

Focus on objects, actions and their relationships

Similar to OO analysis, but ...

- includes non-computer entities
- emphasises domain understanding not implementation

Running example

‘Vera's Veggies’ – a market gardening firm

owner/manager: Vera Bradshaw

employees: Sam Gummage and Tony Peagreen

various tools including a tractor ‘Fergie’

two fields and a glasshouse

new computer controlled irrigation system



# Objects

Start with list of objects and classify them:

Concrete objects:

simple things: spade, plough, glasshouse

Actors:

*human actors*: Vera, Sam, Tony, the customers  
what about the irrigation controller?

Composite objects:

*sets*: the team = Vera, Sam, Tony

*tuples*: tractor may be < Fergie, plough >

# Attributes

To the objects add attributes:

**Object** Pump3 **simple** – irrigation pump

**Attributes:**

status: on/off/faulty

capacity: 100 litres/minute

N.B. need not be computationally complete

# Actions

List actions and associate with each:

agent – who performs the actions

patient – which is changed by the action

instrument – used to perform action

examples:

Sam (*agent*) planted (*action*) the leeks (*patient*)

Tony dug the field *with* the spade (*instrument*)

# Actions (ctd)

implicit agents – read behind the words

`the field was ploughed' – *by whom?*

indirect agency – the real agent?

`Vera programmed the *controller* to irrigate the field'

messages – a special sort of action

`Vera *told* Sam to ... '

rôles – an agent acts in several rôles

Vera as *worker* or as *manager*

# example - objects and actions

**Object** Sam **human actor**

**Actions:**

S1: drive tractor

S2: dig the carrots

**Object** Vera **human actor**

– the proprietor

**Actions:** as worker

V1: plant marrow seed

V2: program irrigation controller

**Actions:** as manager

V3: tell Sam to dig the carrots

**Object** the men **composite**

**Comprises:** Sam, Tony

**Object** glasshouse **simple**

**Attribute:**

humidity: 0-100%

**Object** Irrigation Controller  
**non-human actor**

**Actions:**

IC1: turn on Pump1

IC2: turn on Pump2

IC3: turn on Pump3

**Object** Marrow **simple**

**Actions:**

M1: germinate

M2: grow

# Events

... when something happens

- performance of action  
    'Sam dug the carrots'
- spontaneous events  
    'the marrow seed germinated'  
    'the humidity drops below 25%'
- timed events  
    'at midnight the controller turns on'

# Relationships

- object-object
  - social - Sam is subordinate to Vera
  - spatial - pump 3 is in the glasshouse
- action-object
  - agent (listed with object)
  - patient and instrument
- actions and events
  - temporal and causal
  - 'Sam digs the carrots because Vera told him'
- temporal relations
  - use HTA or dialogue notations.
  - show task sequence (normal HTA)
  - show object lifecycle

# example - events and relations

## Events:

Ev1: humidity drops below 25%

Ev2: midnight

## Relations: object-object

location ( Pump3, glasshouse )

location ( Pump1, Parker's Patch )

## Relations: action-object

patient ( V3, Sam )

- Vera tells *Sam* to dig

patient ( S2, the carrots )

- Sam digs the *carrots* ...

instrument ( S2, spade )

- ... *with* the spade

## Relations: action-event

before ( V1, M1 )

- the marrow must be sown  
*before* it can germinate

triggers ( Ev1, IC3 )

- *when* humidity drops  
below 25%, the controller  
turns on pump 3

causes ( V2, IC1 )

- ☐ the controller turns on the  
pump *because* Vera  
programmed it



# Sources of Information

## Documentation

- N.B. manuals say what is *supposed* to happen but, good for key words and prompting interviews

## Observation

- formal/informal, laboratory/field (see Chapter 9)

## Interviews

- the expert: manager or worker? (ask both!)

# Early analysis

## Extraction from transcripts

- list nouns (objects) and verbs (actions)
- beware technical language and context:  
    `the rain poured' vs. `I poured the tea'

## Sorting and classifying

- grouping or arranging words on cards
- ranking objects/actions for task relevance (see ch. 9)
- use commercial outliner

## Iterative process:

data sources ⇔ analysis

... but costly, so use cheap sources where available

# Uses - manuals & documentation

## Conceptual Manual

- from knowledge or entity-relations based analysis
- good for open ended tasks

## Procedural 'How to do it' Manual

- from HTA description
- good for novices
- assumes all tasks known

### **To make cups of tea**

boil water — see page 2  
empty pot  
make pot — see page 3  
wait 4 or 5 minutes  
pour tea — see page 4

— page 1 —

### **Make pot of tea** *once water has boiled*

warm pot  
put tea leaves in pot  
pour in boiling water

— page 3 —

# Uses - requirements & design

## Requirements capture and systems design

- lifts focus from system to use
- suggests candidates for automation
- uncovers user's conceptual model

## Detailed interface design

- taxonomies suggest menu layout
- object/action lists suggest interface objects
- task frequency guides default choices
- existing task sequences guide dialogue design

## NOTE. task analysis is never complete

- rigid task based design  $\Rightarrow$  inflexible system