4

- A catalogue of basic frames
  - required behaviour
  - commanded behaviour
  - information display
  - simple workpieces
  - transformation

Basic classes and frames

- In decomposing a problem in sub-problems, a small number of standard patterns emerged
- Five identified by M. Jackson (not the singer)
  - Required behaviour
  - Commanded behaviour
  - Information display
  - Simple workpieces
  - Transformation
- One proposed by Bray & Cox (but not generally included in the “canon”)
  - Simulator

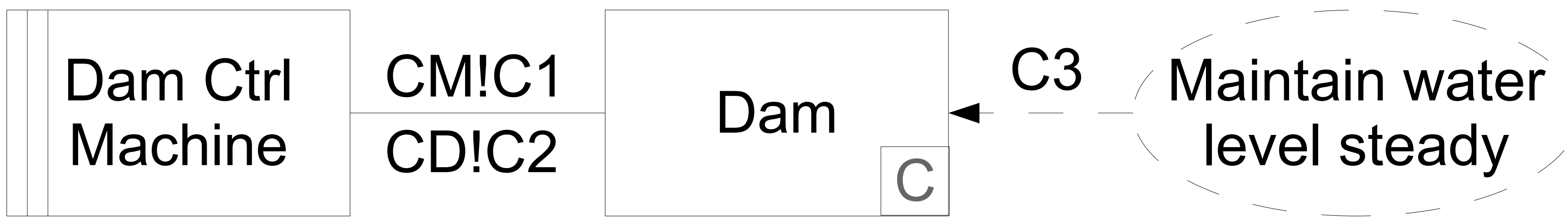
Basic classes and frames

- All basic frames differ in
  - Requirements (better, requirements schema)
  - Properties of the domains (causal, biddable, etc.)
  - Involvement of the domains (extent to which they are controlled or constrained)
  - Concerns (things to prove to assure that the frame is well-formed and solution appropriate)
- We will briefly show the basic “canonic” frames and their correctness concerns
  - Supplementing them with HCI concerns and distributed concerns

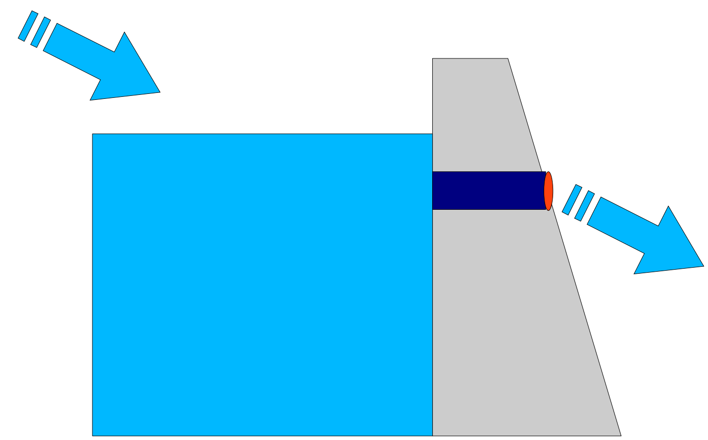
Required behaviour

- There is some part of the physical world whose behaviour is to be controlled so that it satisfies certain conditions.
- The problem is to build a machine that will impose that control.

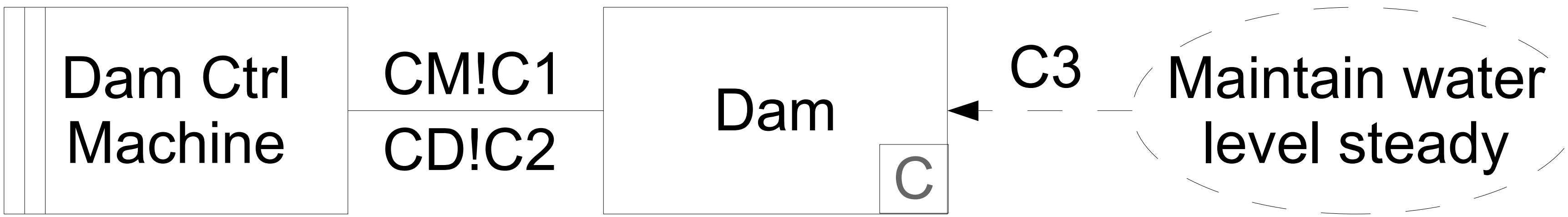
Required behaviour example



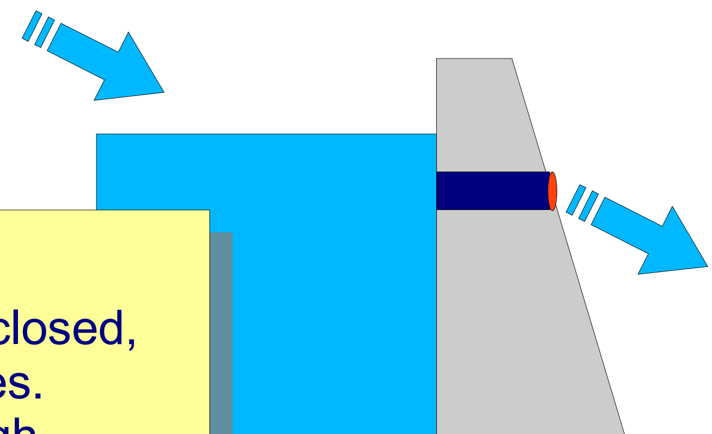
C1: OpenGate, CloseGate
C2: SensorReading
C3: WaterLevel



Required behaviour example



C1: OpenGate, CloseGate
C2: SensorReading
C3: WaterLevel



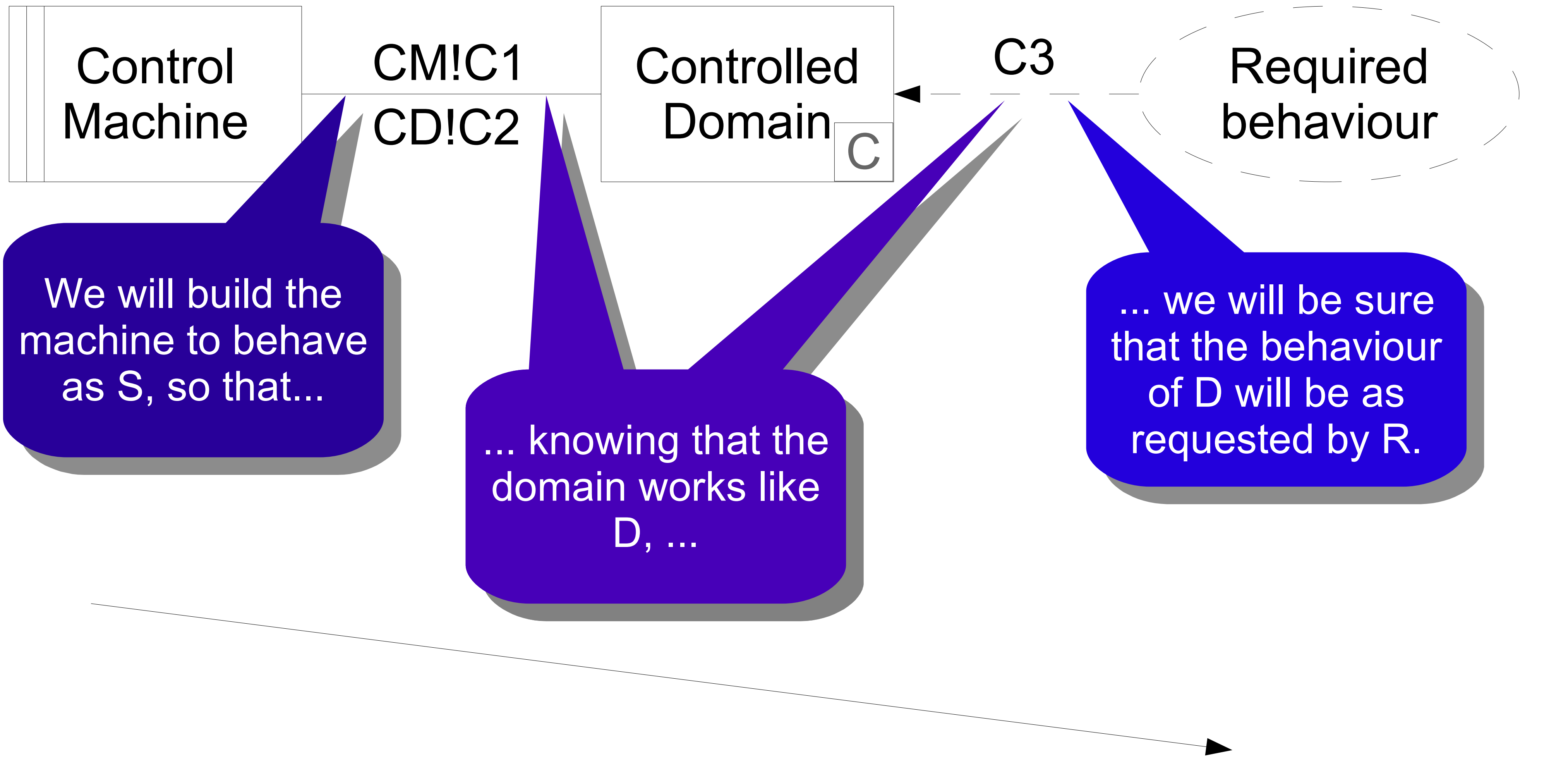
Correctness argument

Domain properties of Dam are such that when the gate is closed, water level increases; when it is open, water level decreases. Moreover, the Dam will report the current water level through sensors.

Specification of the machine will be that when water level is below requested, it will command closing the gate; when it is above, it will command opening the gate.

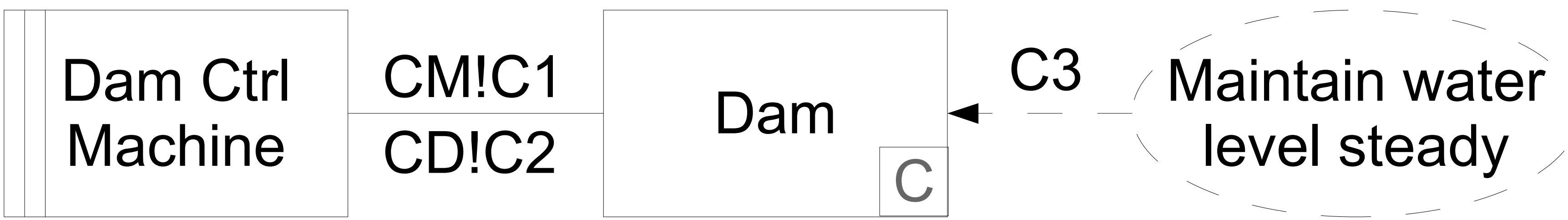
Hence, **Requirements** about maintaining steady level will be satisfied.

Required behaviour concerns

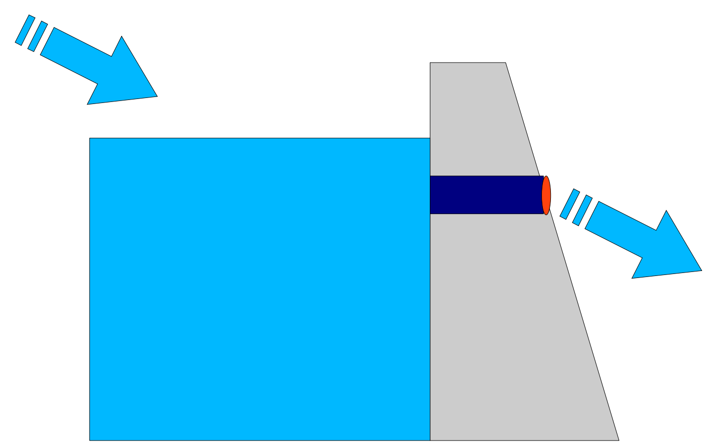


Exercise

- Can you write formally S, D and R for the dam example and satisfy the concern?



C1: OpenGate, CloseGate
C2: SensorReading
C3: WaterLevel



Required behaviour h-concerns

- The required behaviour frame assumes that the controlled domain is a *causal* domain
- Humans are always (at most) biddable
- Hence, **no human-centric concerns for the required behaviour frame**
  - (phew, that was easy)

Are you serious??

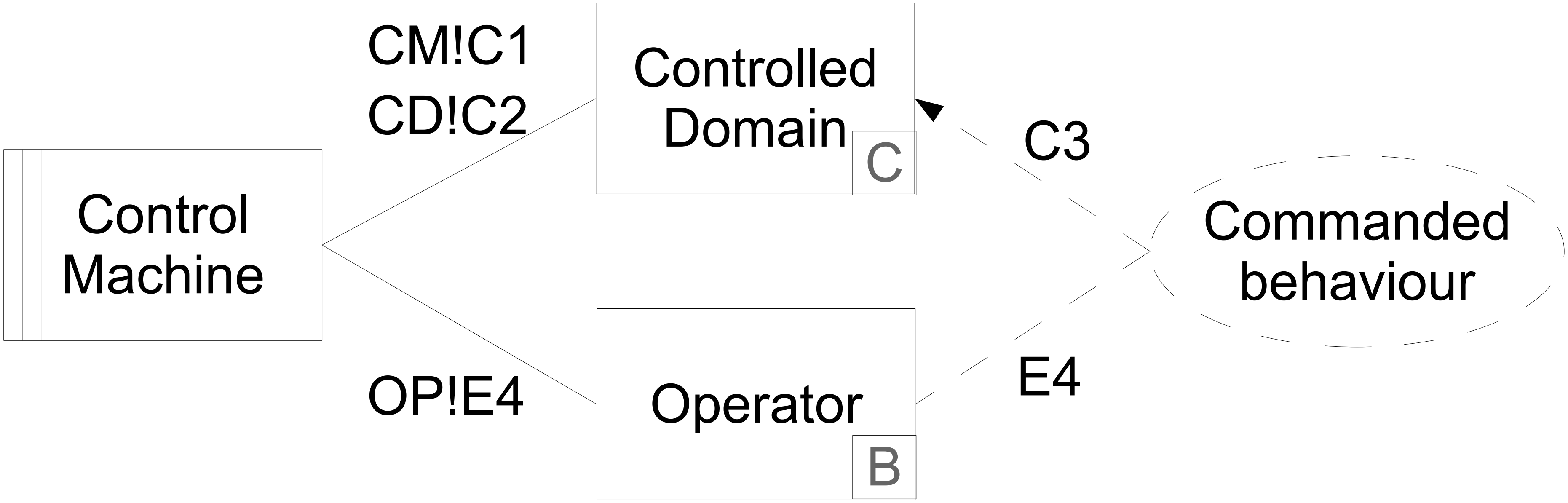
- The machine is *totally* autonomous? No human at all?
  - No, most probably the machine will be *configured* (e.g., to know what the desired steady level in the dam should be)
    - We will see later how a human can provide such configuration information
  - Also, there will be most probably *On*, *Off*, *Reset* etc. events, caused by humans, that affect the machine
- But we are talking about basic subproblems
  - Real scenario come from composing them

Exercise

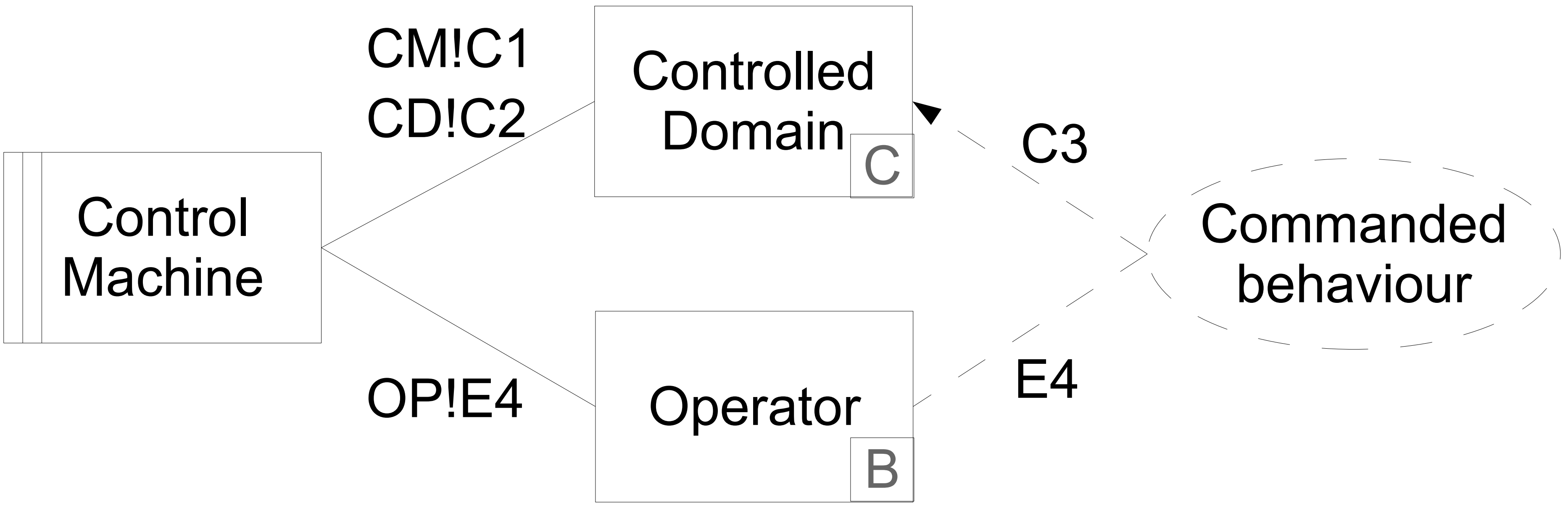
- Assume now that DM is linked to the actual dam through a connection domain
- Identify properties that the connection domain must satisfy so that the correctness argument still holds
- Consider several common implementation technologies for the connection domain, and discuss their properties as above

Commanded behaviour

- There is some part of the physical world whose behaviour is to be controlled in accordance with commands issued by an operator
- The problem is to build a machine that will accept the operator's commands and impose the control accordingly.



Commanded behaviour

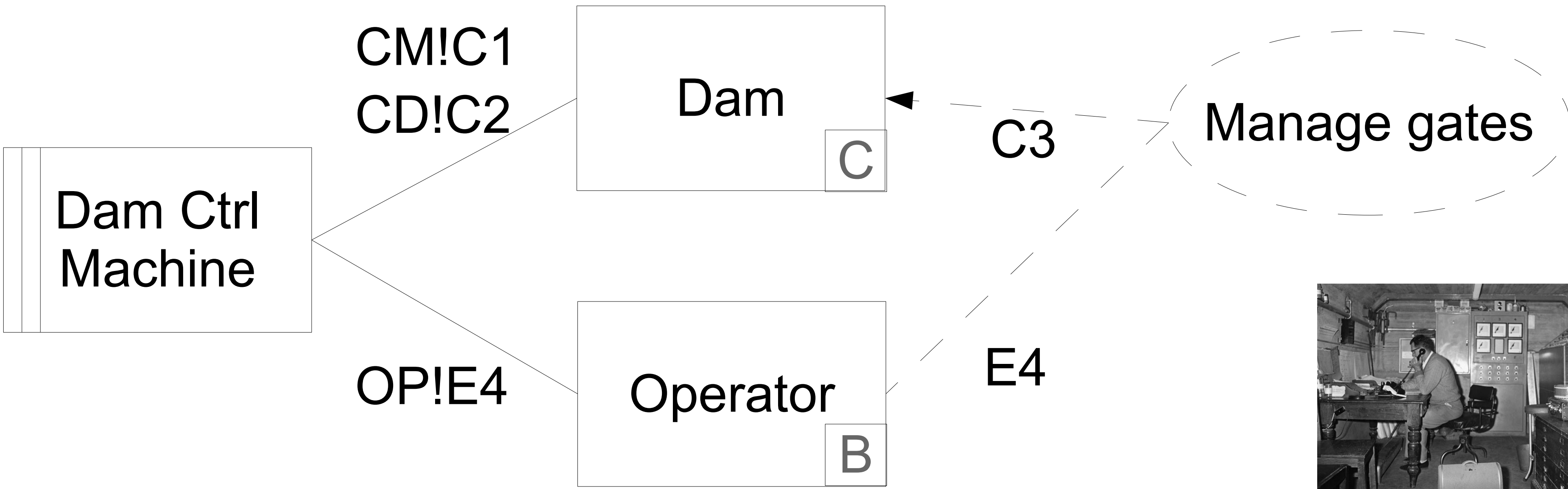


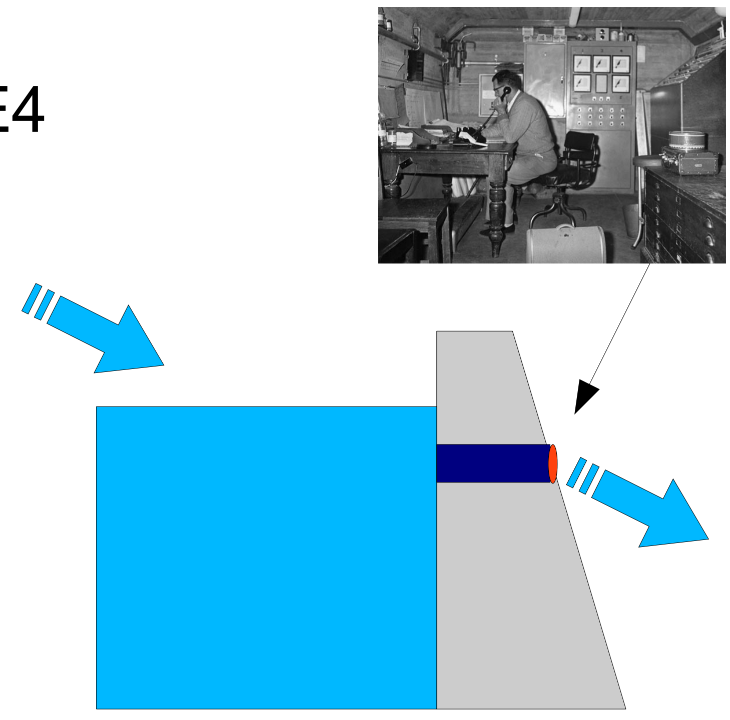
- Similar to the required behaviour frame, but
  - The Operator can issue commands to the system
  - The Operator is a biddable domain; neither the machine nor the requirements constrain his ability to spontaneously issue commands

Commanded behaviour example

- Same as the Dam problem, but
  - The control machine has two modes, *Auto* and *Manual*
  - In Auto mode, the machine autonomously controls the level
  - In Manual mode, the Operator can send OpenGate and CloseGate commands
  - The Operator can switch between Auto and Manual at will
  - In any case, the water level should not go over a safety level

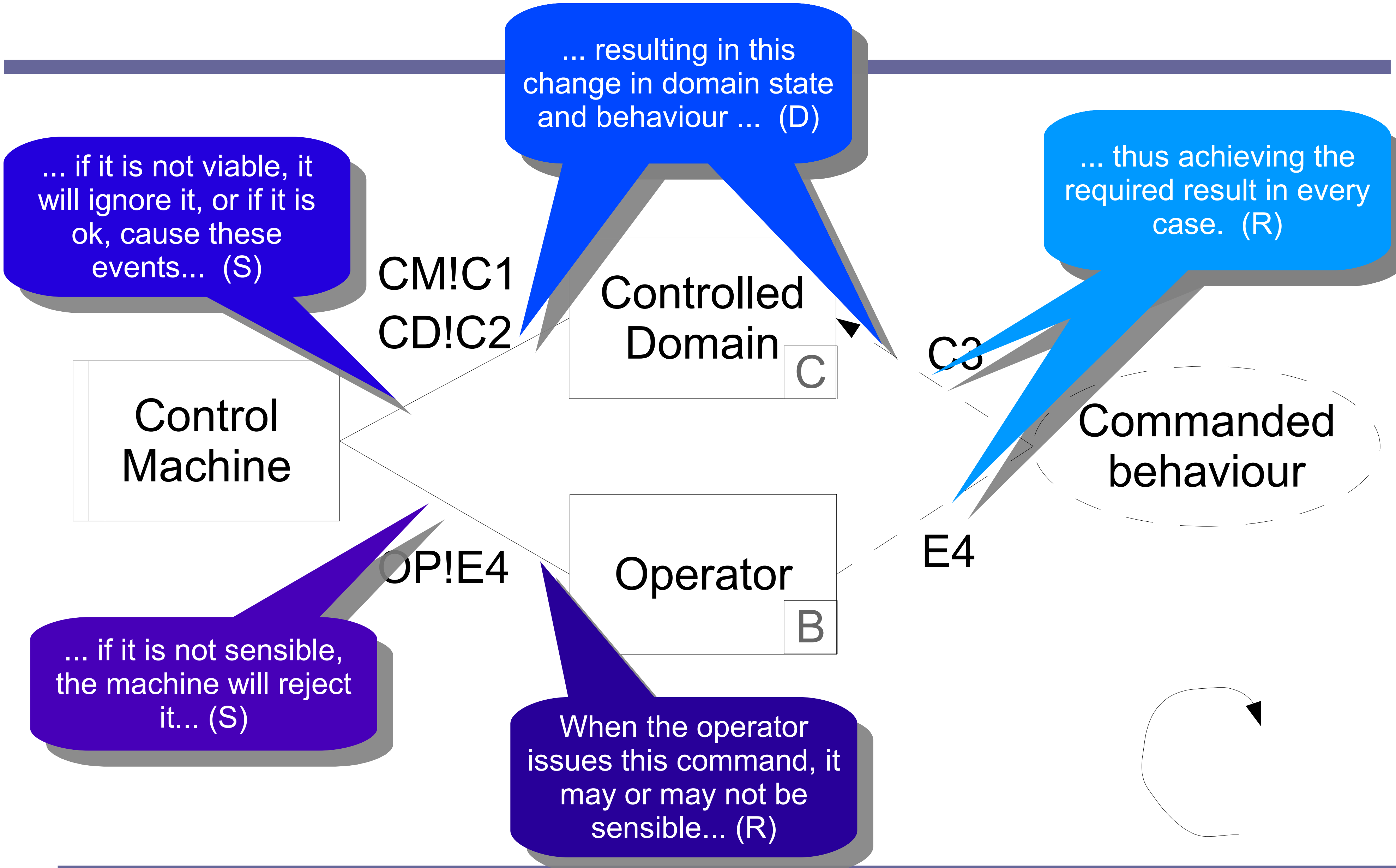
Commanded behaviour example





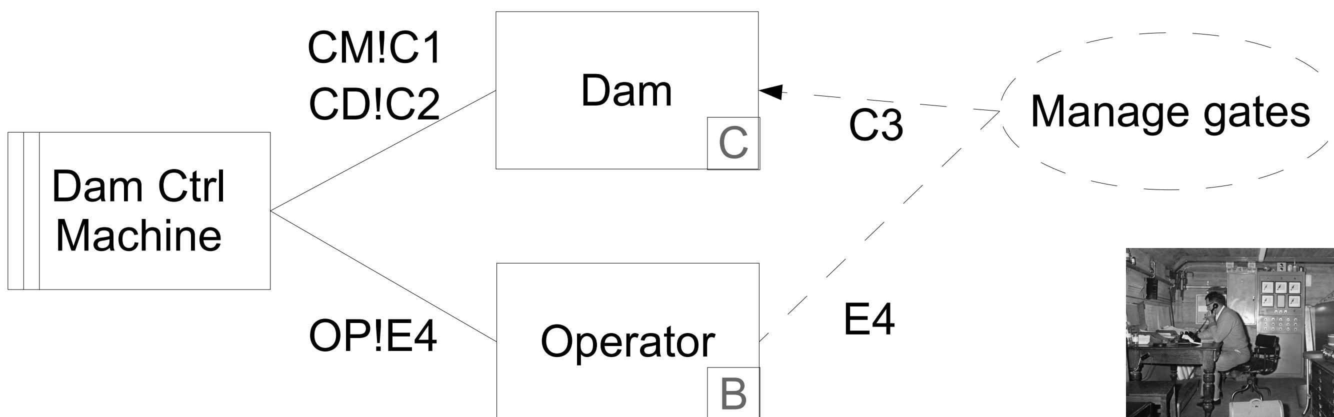
C1: OpenGate, CloseGate
C2: SensorReading
C3: WaterLevel
E4: GoAuto, GoManual, CmdOpen, CmdClose

Commanded behaviour concerns

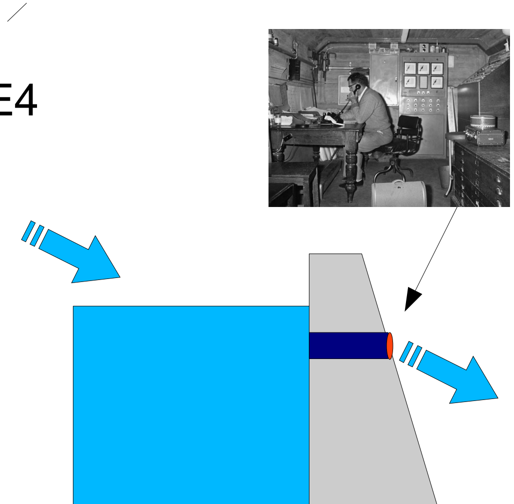


Commanded behaviour exercise

- Can you write formally S, D and R for the dam example and satisfy the concern?
  - A little more complex, try it offline



C1: OpenGate, CloseGate
C2: SensorReading
C3: WaterLevel
E4: GoAuto, GoManual, CmdOpen, CmdClose



Commanded behaviour h-concerns

- Here we have an operator; HCI concerns!
  - Can we **prevent** issuing of not-sensible commands?
    - GUI style
      - "Ghost out" menu entries or other gadgetry
      - Make the screen or window containing the controls inaccessible
    - Command line
      - Structured editing
      - Up-front parsing and reporting (red squiggles style)
    - Mechanical (buttons, levers, etc.)
      - Physical blocking mechanism
      - Red light when a command should not be issued
    - Remote entry (through a phone, dialing or voice recog)
      - Any idea?

Commanded behaviour h-concerns

- Here we have an operator; HCI concerns!
  - How should we **reject** not-sensible commands?
    - GUI style
      - MessageBox “You entered an invalid command” [Ok] - then?
    - Command line
      - Error message (Style? Wording? Apologetic? Suggestions?)
      - Disconnect/invalidate session? What about web apps?
    - Mechanical (buttons, levers, etc.)
      - Physical restoring mechanism (motor to pull back a lever)
    - Remote entry (through a phone, dialing or voice recog)
      - Recorded message, maybe a data entry/recog problem?

Commanded behaviour h-concerns

- Here we have an operator; HCI concerns!
  - How should we **report** not-viable commands?
    - GUI style
      - MessageBox "Your command was ignored because..." [Ok] - then?
    - Command line
      - Error message (Style? Wording? Apologetic? Suggestions?)
    - Mechanical (buttons, levers, etc.)
      - Physical restoring mechanism (motor to pull back a lever)
    - Remote entry (through a phone, dialing or voice recog)
      - Recorded message, maybe a data entry/recog problem?
  - In any case
      - Should we reset interaction state?
      - Should we offer alternative ways to reach the end?

Commanded behaviour h-concerns

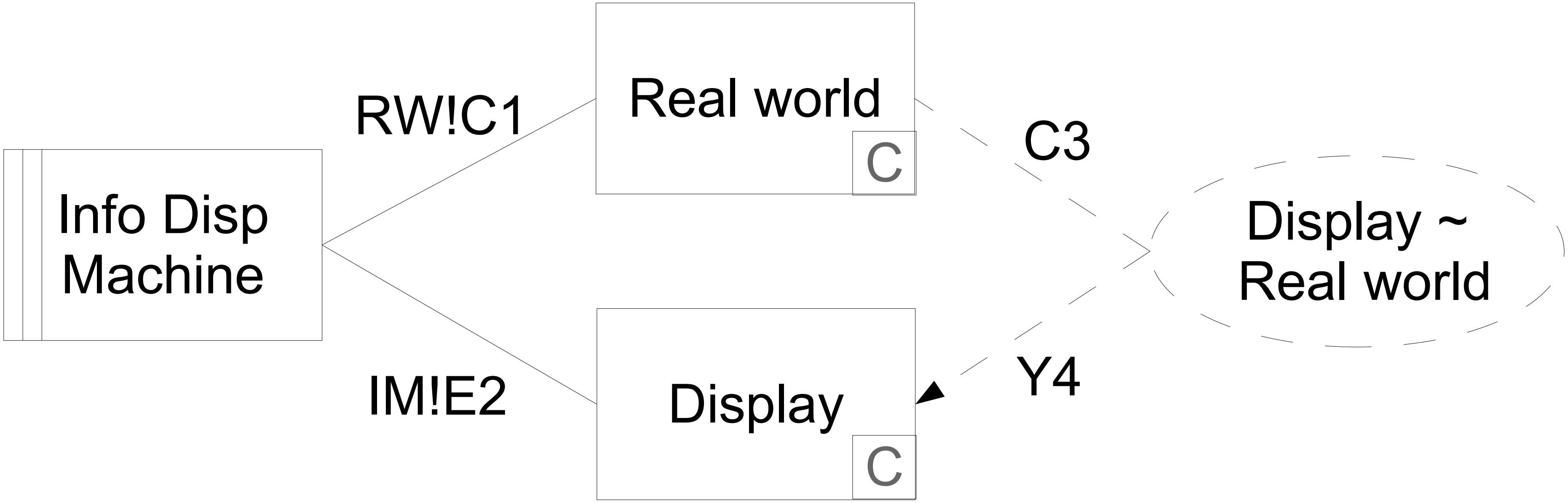
- Here we have an operator; HCI concerns!
  - How can we **reconcile** the state of the domain and the world model of the operator?
    - Present a full account of how things are (see later the information display frame)
    - Try to understand the mental context of the Operator, and act accordingly
    - Attempt bringing the domain in synch with the stated desires, inside the acceptable space as defined by R
      - Planning system, veering towards AI!
    - Provide means of notification of events happening in the domain

Commanded behaviour h-concerns

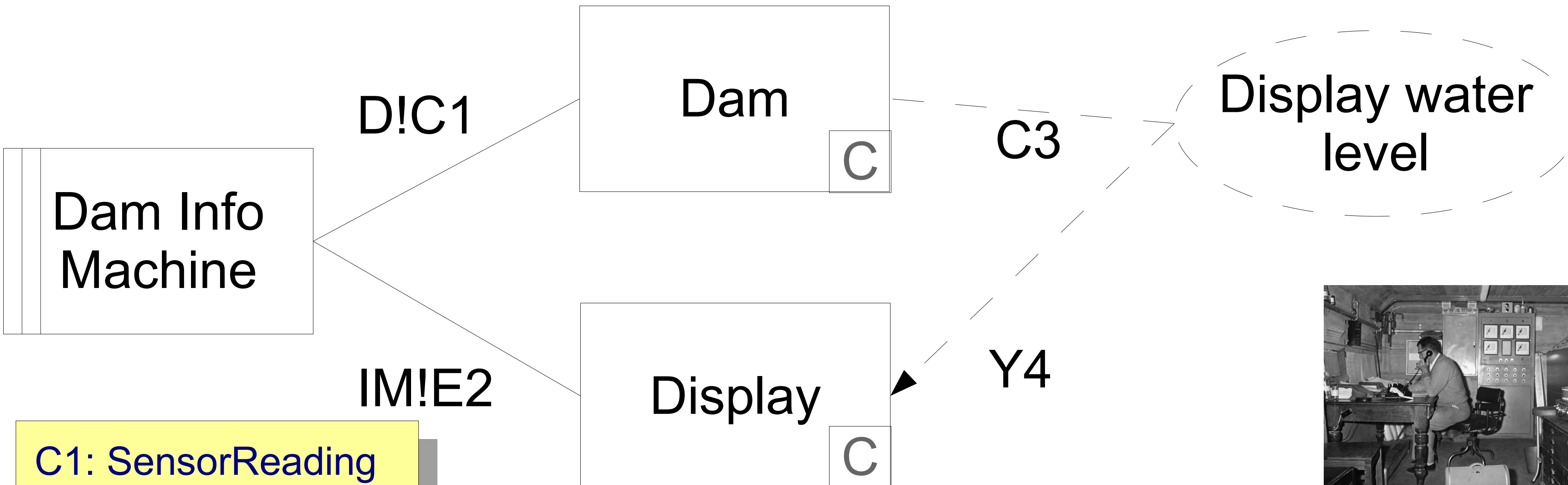
- Here we have an operator; HCI concerns!
  - How can we **cooperate** with the Operator?
    - Always present a view of which commands are sensible *and* viable
      - Notification when the view changes spontaneously, due to domain events?
      - Can be costly when the domain is complex
  - How can we **avoid surprising** the Operator?
    - Inform the Operator in advance about commands issued autonomously by the Machine
    - Inform the Operator in advance about events in the domain which change the viability of his/hers commands

Information display

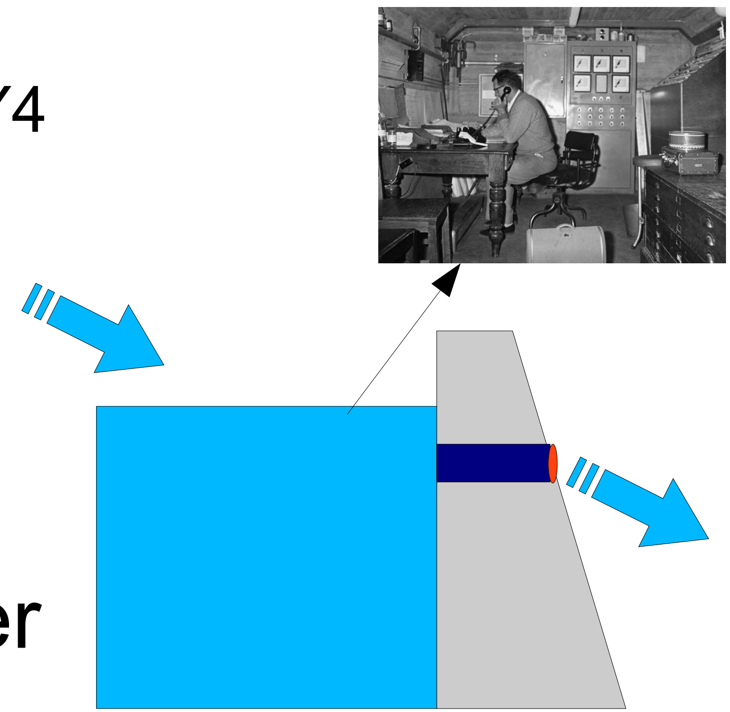
- There is some part of the physical world about whose states and behaviour certain information is continuously needed.
- The problem is to build a machine that will obtain this information from the world and present it at the required place in the required form.



Information display example

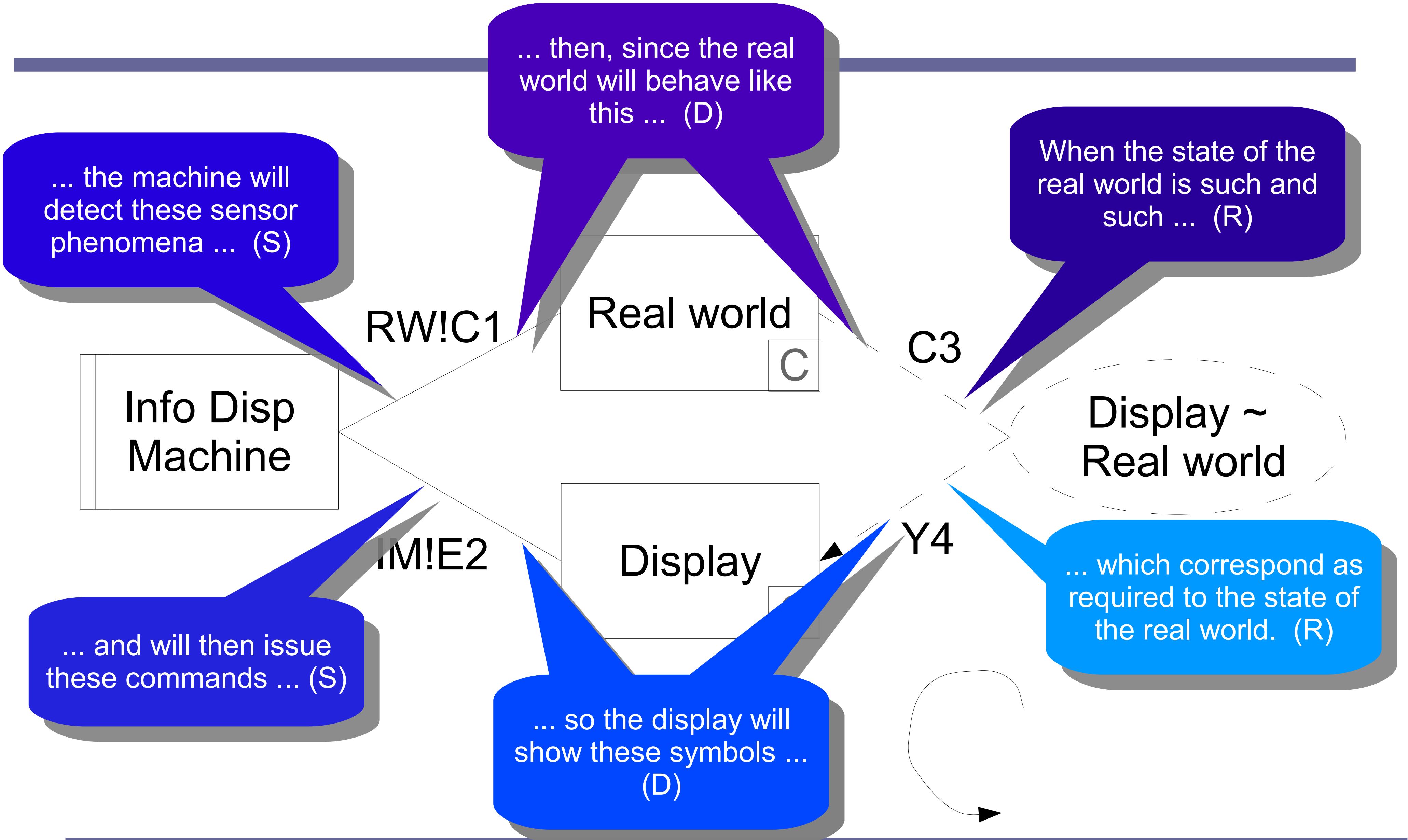


C1: SensorReading
E2: UpdateDisplay
C3: WaterLevel
Y4: ShownValue



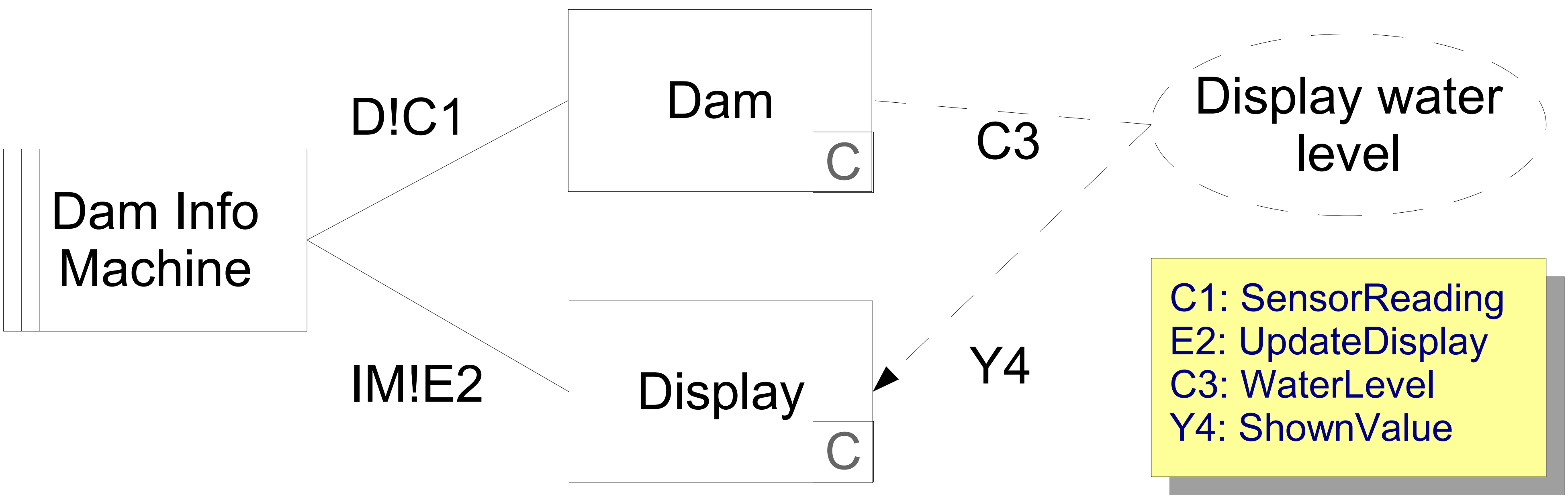
- The display has to show continuously the current water level in the dam

Information display concerns



Information display exercise

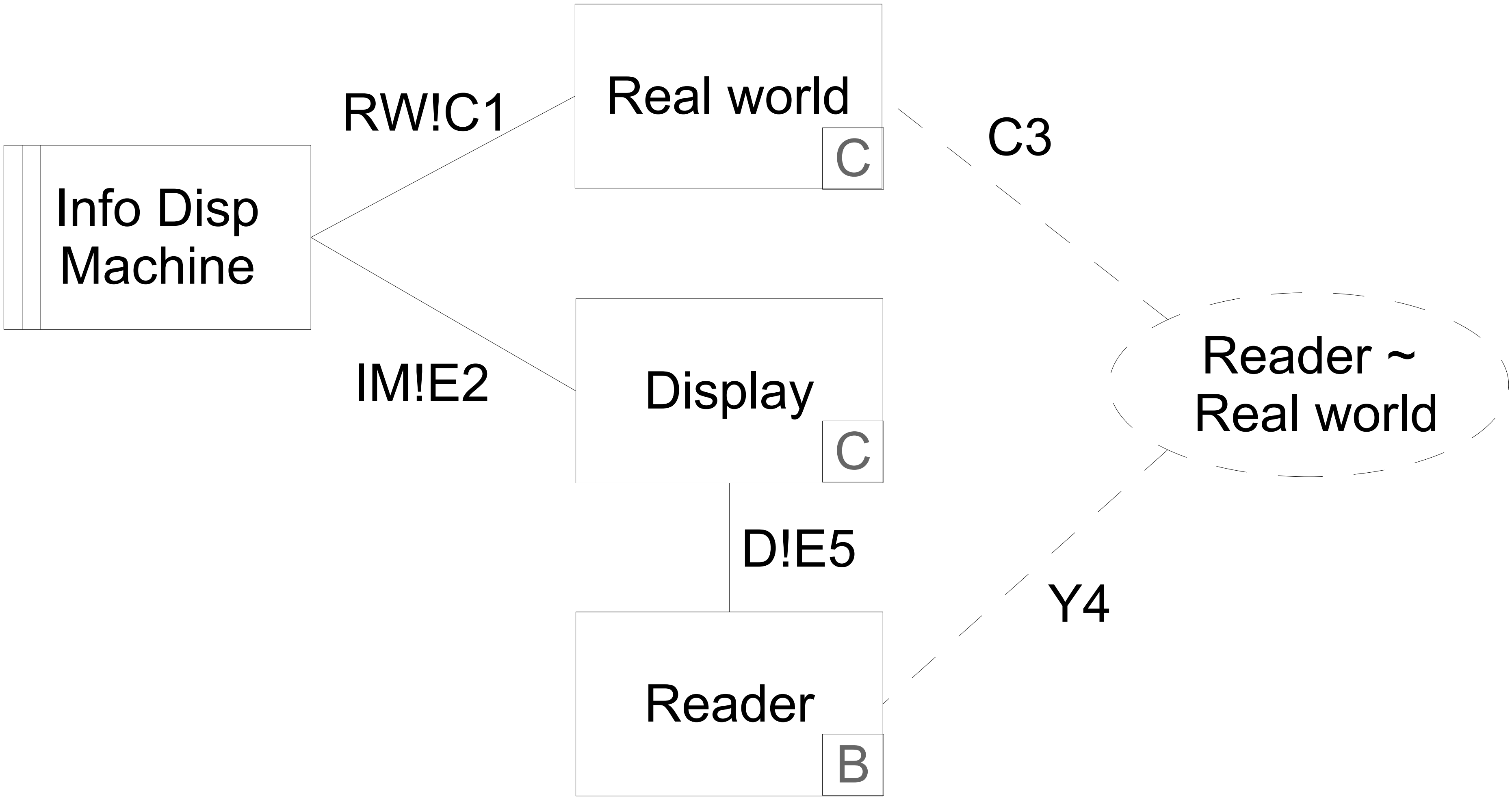
- Describe and prove the correctness concern for the Dam display problem
  - You will need to write R, D and S
  - What can be learned about risks with the dam?



Information display h-concerns

- Apparently, there is none: no human domain
- However, often the *real* goal is not just to have the information shown (or printed, etc.), but to have someone read it
  - Although it is not always so
    - For example, in a public display of buses arrival times, it is not really our concern whether passengers read the times or not
    - We can be satisfied with providing the possibility
- Can we go “deeper in the world”?

Information display h-concerns

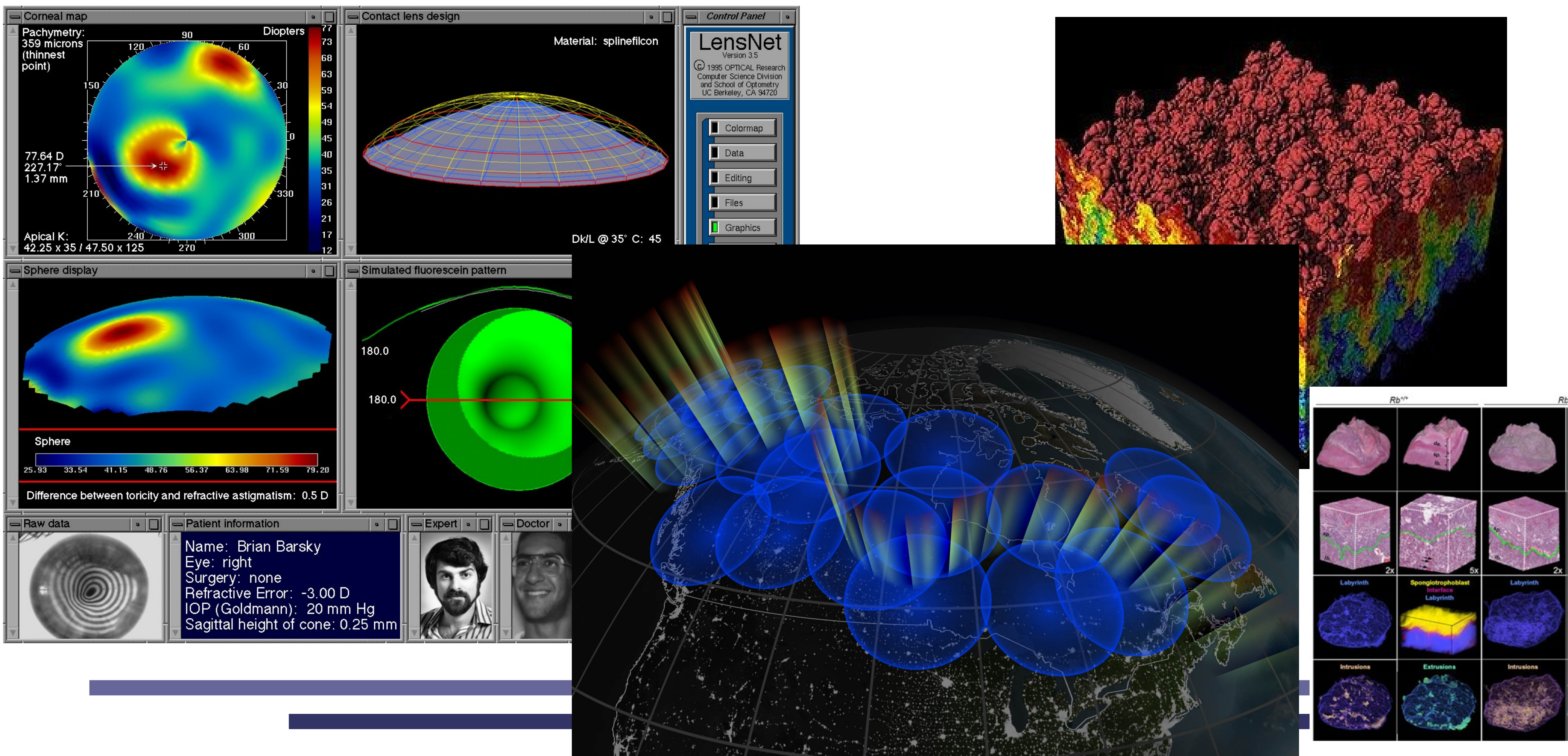


Information display h-concerns

- The HCI concern here is how to make sure that the Reader has an accurate knowledge of what the Display is showing
  - Standard HCI theme: information representation
    - Accurate values: show floating point numbers
    - Accurate timing: notify through multiple channels at any change of the value
    - Accurate appreciation: show clearly qualitative data, use analogy with familiar models, metaphor
    - Accurate dynamics: keep track of values in time
    - Accurate prediction: add forecasts, countdowns, etc.

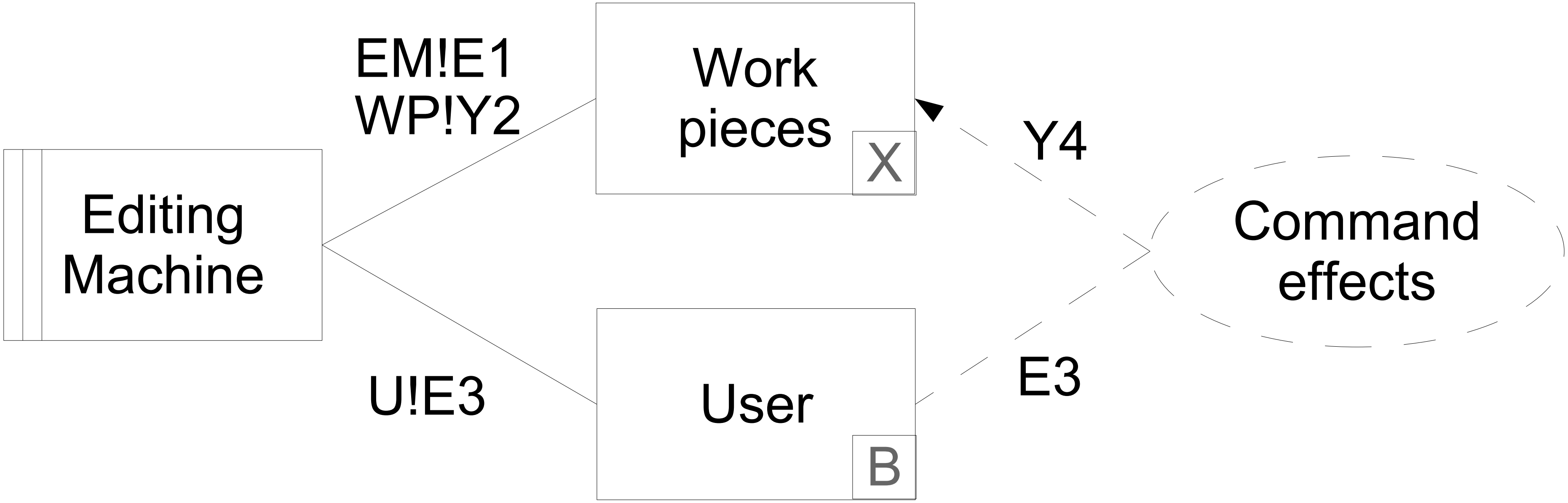
Information display h-concerns

- The HCI challenge becomes significant when data is complex, dynamic, multi-dimensional...



Simple workpieces

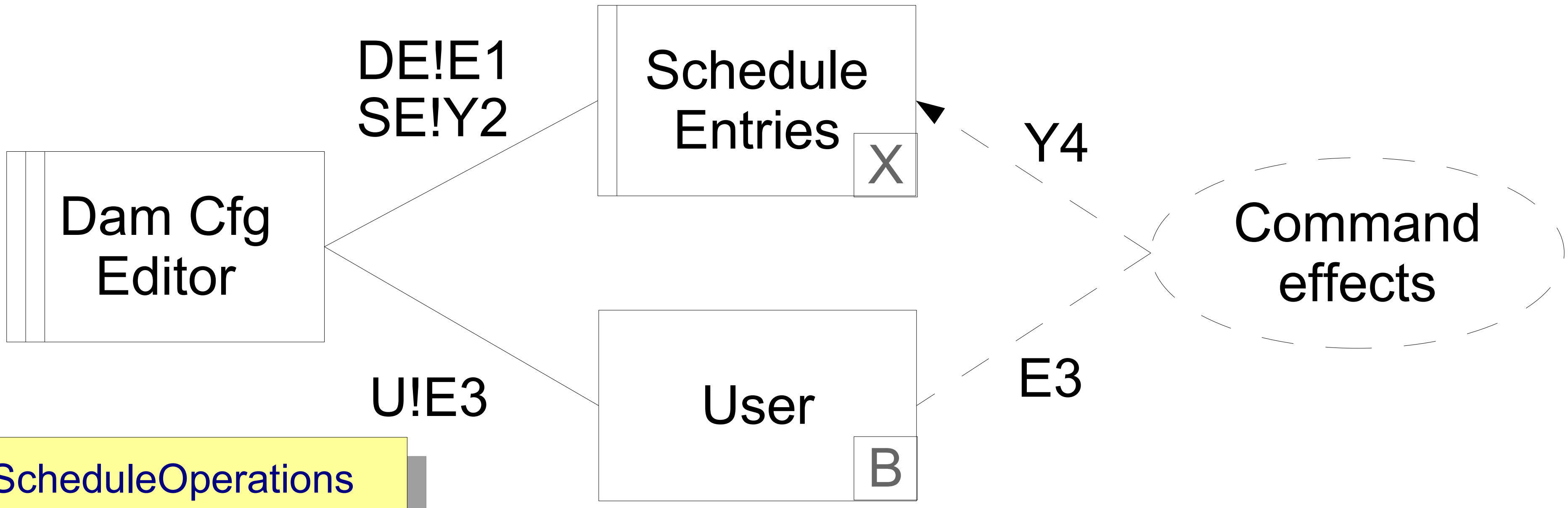
- A tool is needed to allow a user to create and edit a certain class of computer-processable text or graphic objects, or similar structures, so that they can be subsequently copied, printed, analysed or used in other ways.
- The problem is to build a machine that can act as this tool.



Simple workpieces example

- Let us assume that the steady level desired in our dam varies with seasons and times
  - For example, the dam is part of a hydro power plant; level should be higher at night because the dam is used as reservoir to adjust to higher electricity request during the day
- We need **a tool** to let a user set these levels
  - Each entry (**workpiece**) will be <dateStart, dateEnd, weekDays, timeStart, timeEnd, desiredLevel>
  - **Commands** to create/delete/modify entries
  - Default level for times not covered by any entry

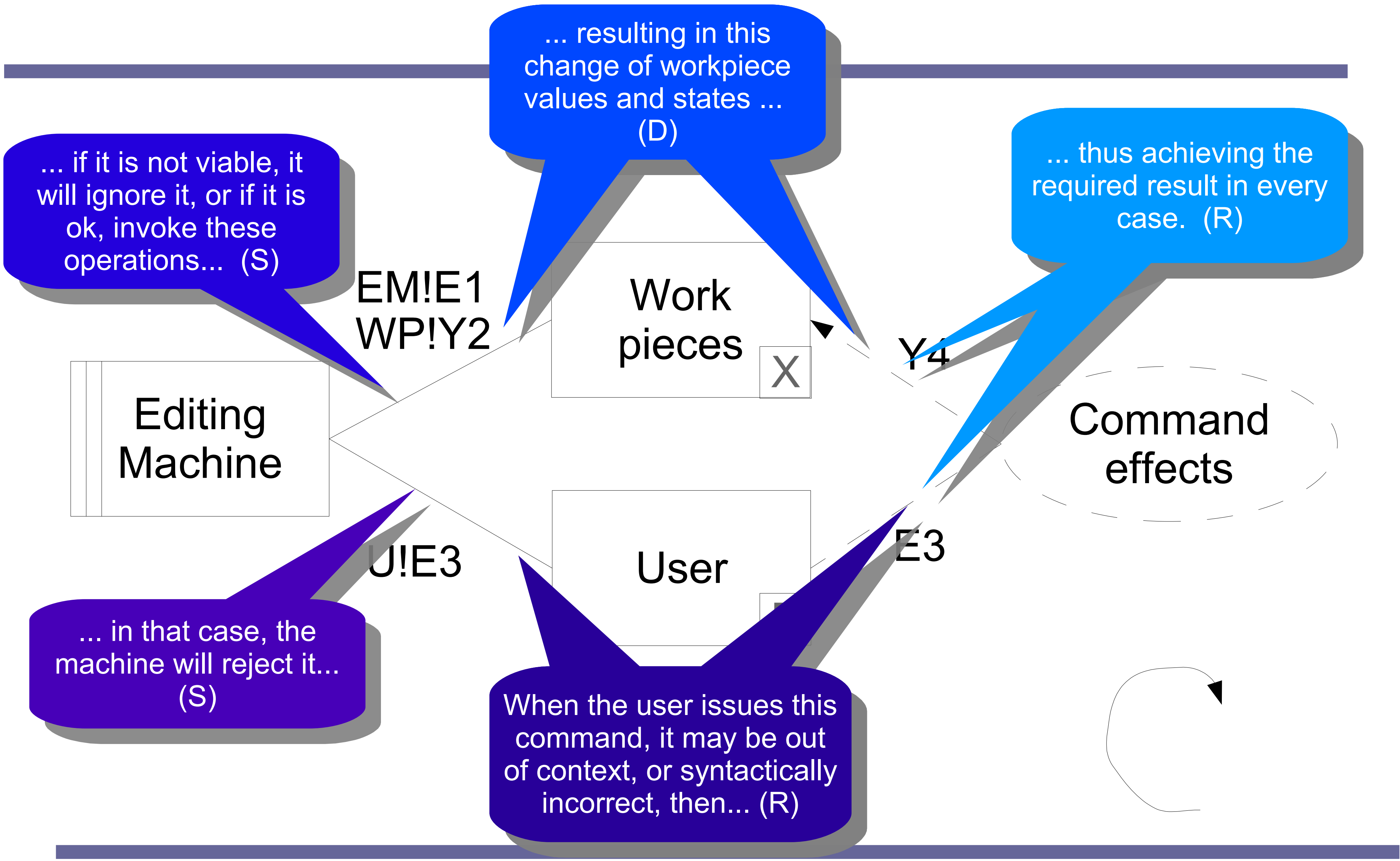
Simple workpieces example



E1: ScheduleOperations
Y2: CurrentSchedule
E3: Commands
Y4: Effects

– Oversimplified here
  - We should describe all the different commands, with their parameters and effects,the lexical structure of the schedule (default level, {entries}), all the constraints on the valid commands

Simple workpieces concerns



Simple workpieces un-exercise

- Due to the relative complexity of the lexical domain and of the commands, we cannot run an exercise on this frame
- However, in real cases it is **critical** to develop a complete and accurate description of the domain and of the phenomena
  - In most cases, theory of formal languages (e.g., lexical grammars and action grammars) can help
  - Applicable also to graphical editing: command events are *symbolic*, not textual!

Simple workpieces concerns

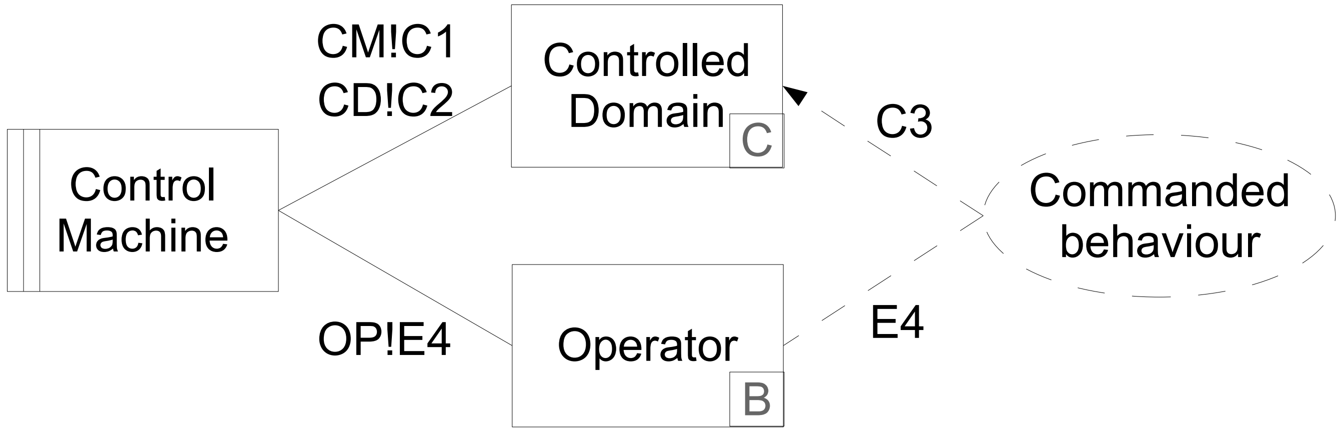
- The correctness concern we have seen before; other concerns include:
  - Lexical domain description
    - Grammars, metamodels, UML, ontologies, etc.
  - Editing operations description
    - Set of operations with arguments
  - Editing operations semantics
    - Describe how each command alters the lexical domain
  - User commands description
    - Describe how the user will enter the commands

Simple workpieces concerns

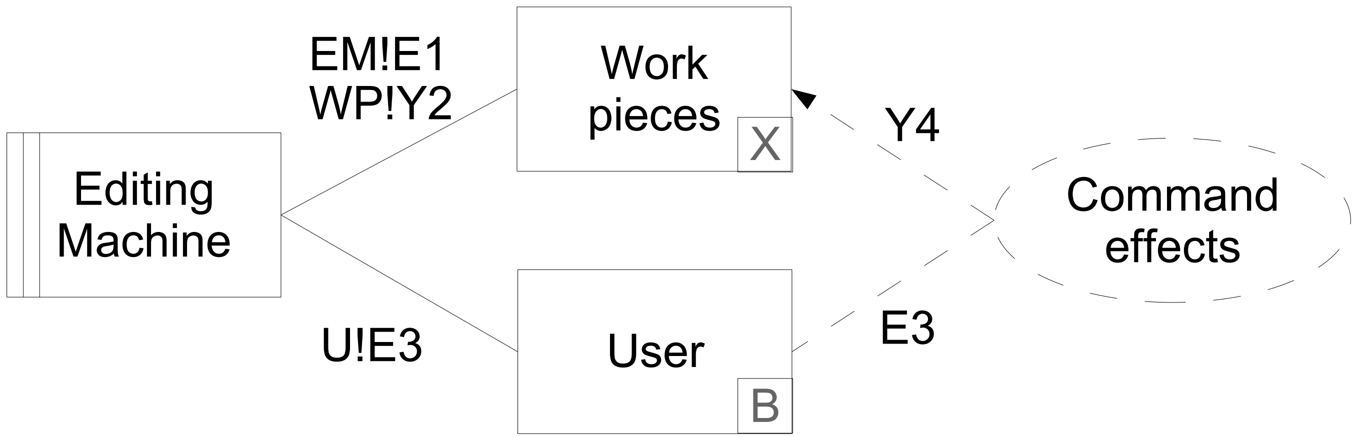
- In the requirements, we should specify:
  - User commands validity
    - Syntax, valid context
  - User commands effects (in various context)
    - Invariants of applying the commands
- The specification should state
  - Machine reaction to valid commands
    - How they are performed on the lexical domain
  - Machine reaction to invalid commands
    - How they are rejected

Simple workpieces h-concerns

- They are very similar to the ones we discussed for the Commanded behaviour frame
- Comm Beh:



- Simple W/p:

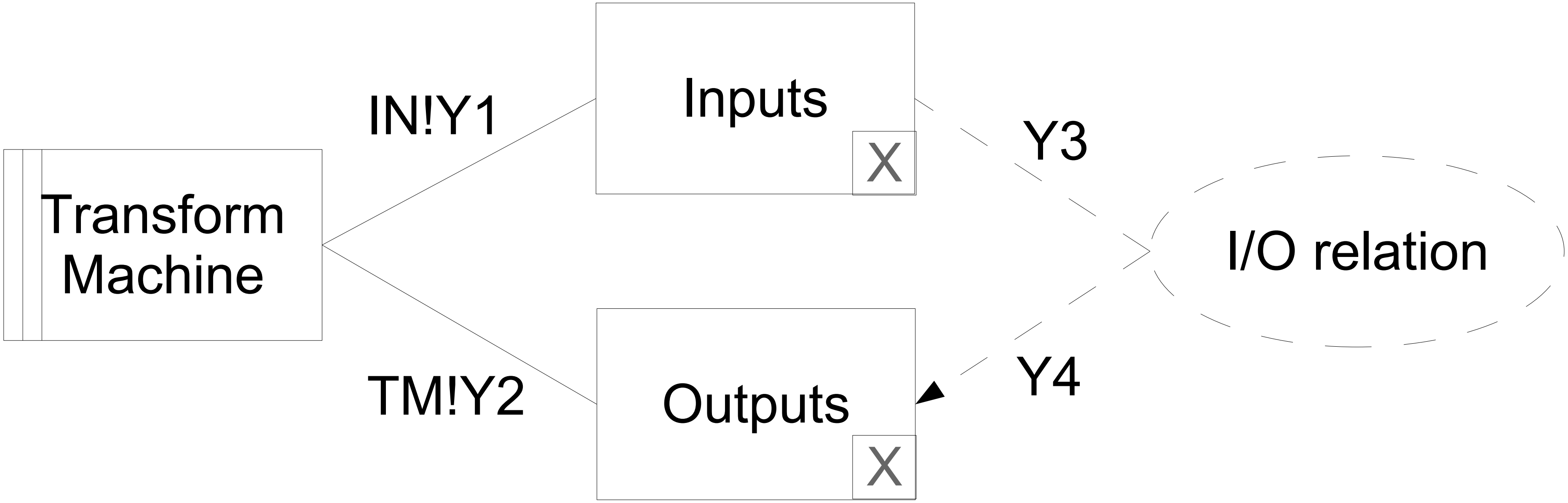


Exercise

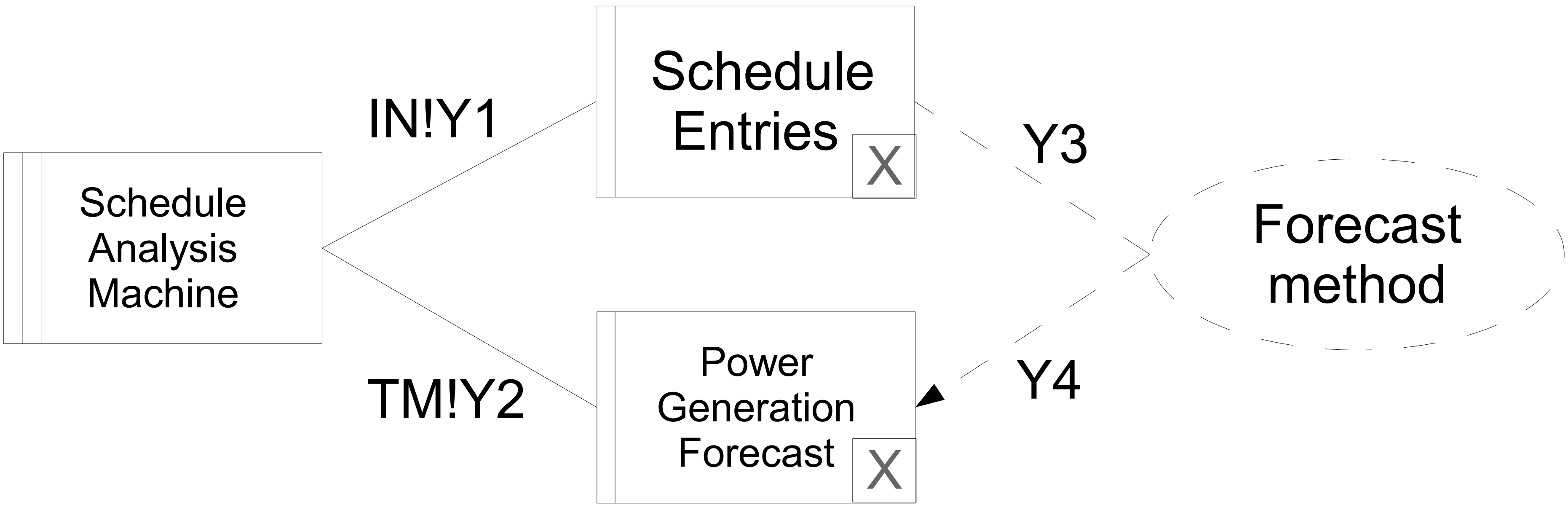
- If Commanded Behaviour and Simple Workpiece are so similar, why are they two distinct frames?
- Spot the differences!

Transformation

- There are some given computer-readable input files whose data must be transformed to give a certain required output files. The output data must be in a particular format, and it must be derived from input data according to certain rules.
- The problem is to build a machine that will produce the required outputs from the inputs.

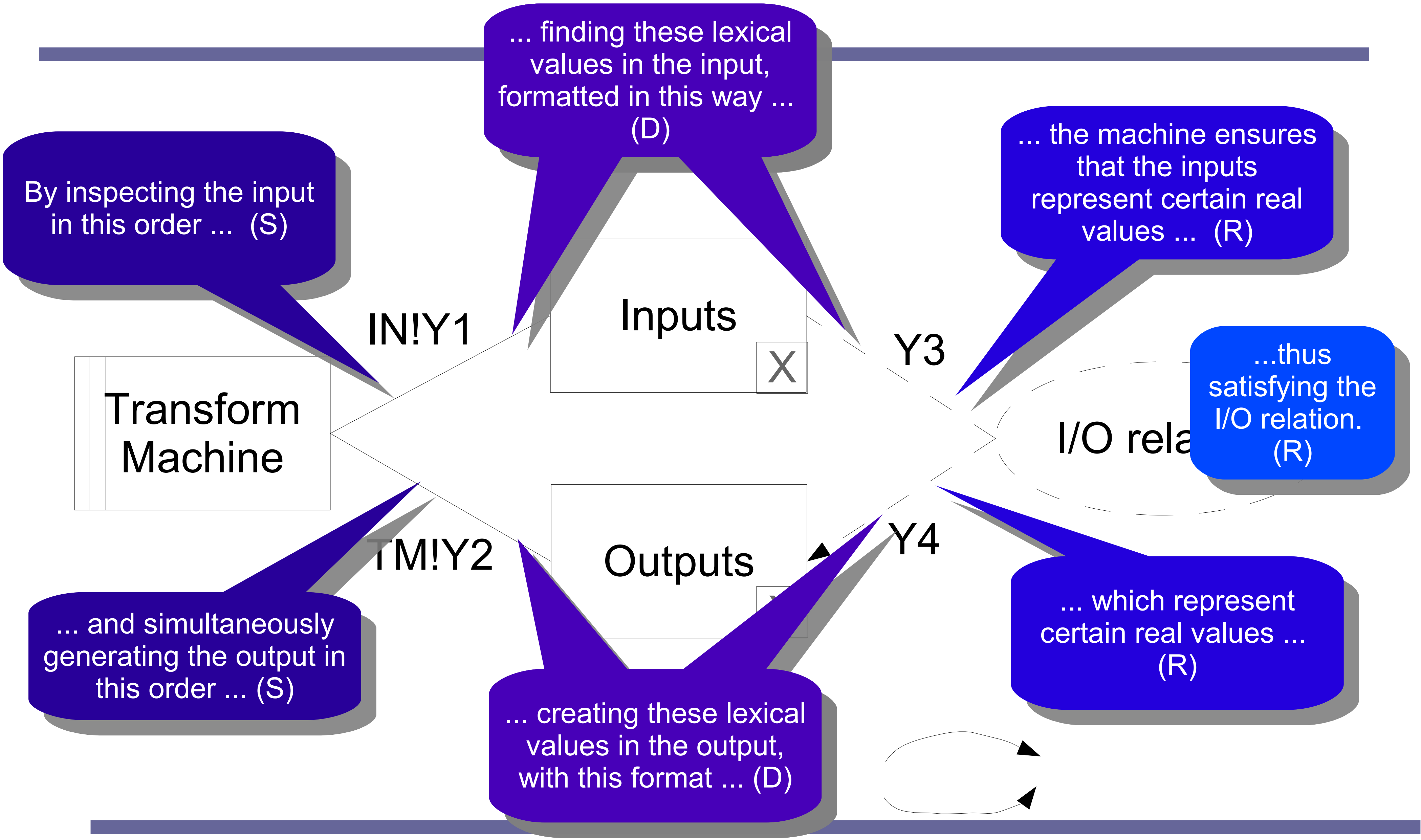


Transformation example



- Given a file with the yearly schedule for the Dam, produce a printed report of the expected power generation for each month.

Transformation concerns



Transformation h-concerns

- Really none: no humans, and no “real world”
  - The issue of error reporting can be treated separately