A Checkout Counter Controller

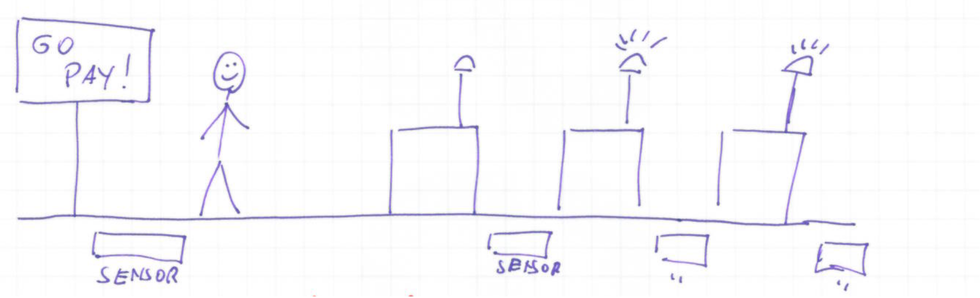
Manuel Carro

IMDEA Software Institute &
Universidad Politécnica de Madrid

Landscape so far

	Types of formulas	
	Easy (few or no quantifiers)	More involved (quantifiers)
Sequential	Integer division, $n^2 = 1 + 3 + \ldots$	Binary search
Concurrent, environment	Coffee Club	Checkout Counters

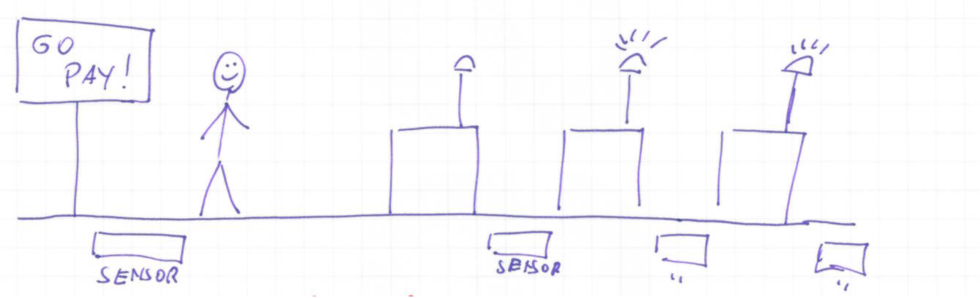
Environment



People follow the rules
(otherwise automating is pointless or very difficult)

- Clients wait: entrance screen gives permission to go to counter.
- **Additionally**: clients wait for space between screen and counter to be empty.
- Clients do not leave counter before it has noticed client.

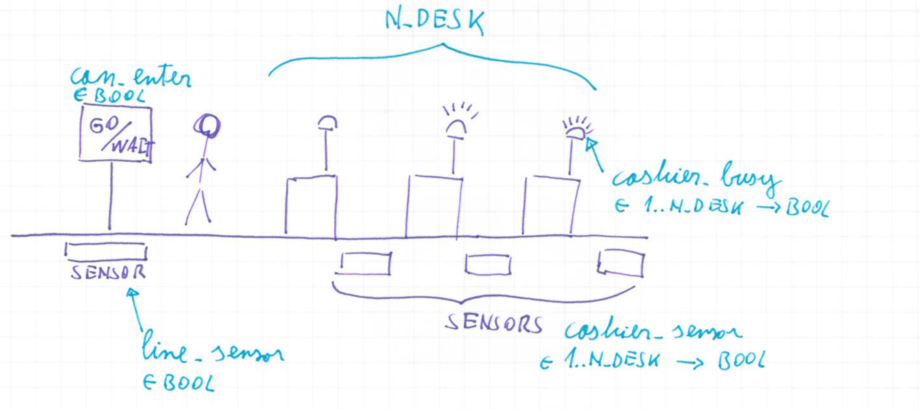
Environment



People follow the rules
(otherwise automating is pointless or very difficult)

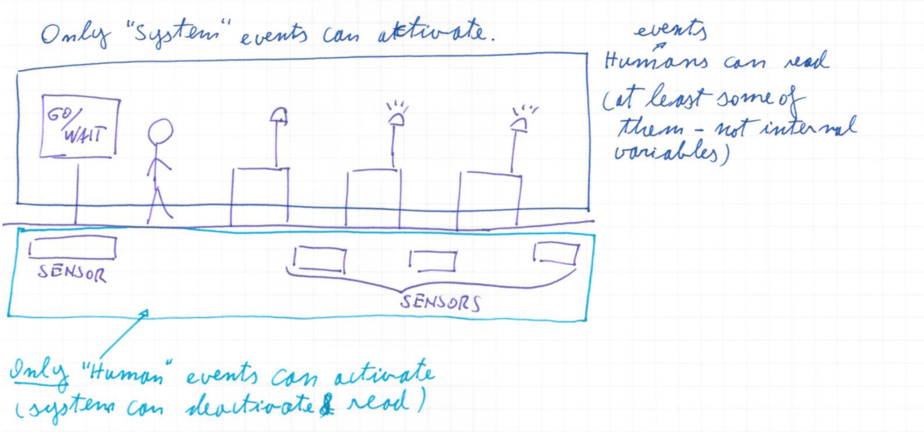
- Sensor after screen and in every counter: detect people.
- “Busy” light in every counter.
- People can go to any free counter.

Model: variables



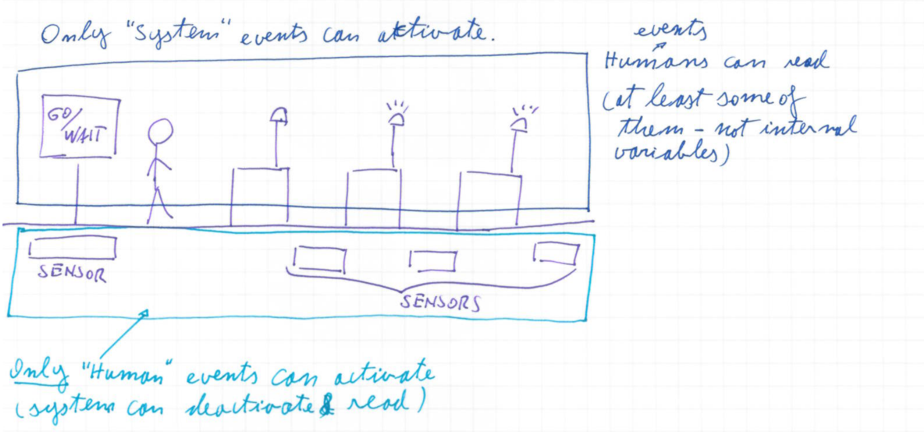
- “Signals” (screen, busy light, sensors, . . .) modeled with variables.
- Plus internal state.
- Also system characteristic (e.g., # of counters).

Types of Events



- Reactive system: neverending interaction loop (with humans).
- Correctness: we need to *simulate* (acceptable) human behavior.
- In general: model environment to ensure system works correctly.

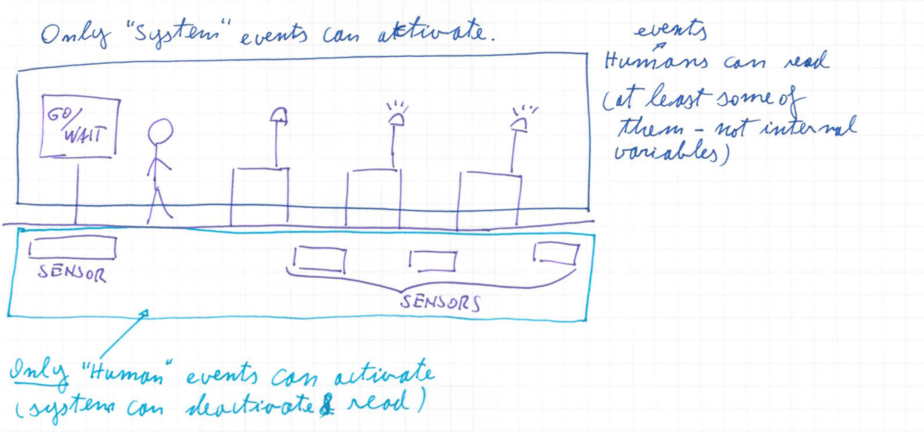
Types of Events



~~Human~~ Environment events

- Environment events read from some *system variables*.
- Give input to system by writing to variables modeling sensors.
- Cannot write on system variables / read from internal system variables.

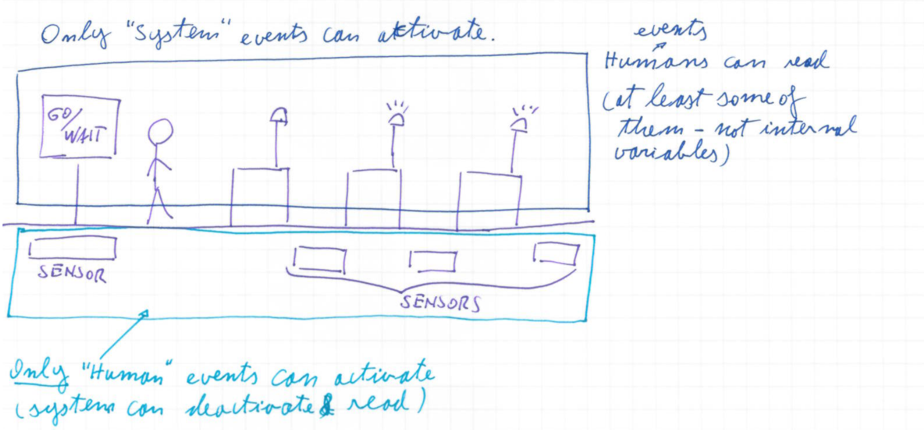
Types of Events



<u>System events</u>

- Read and write system variables.
- Read environment variables.
- Cannot write onto environment variables.

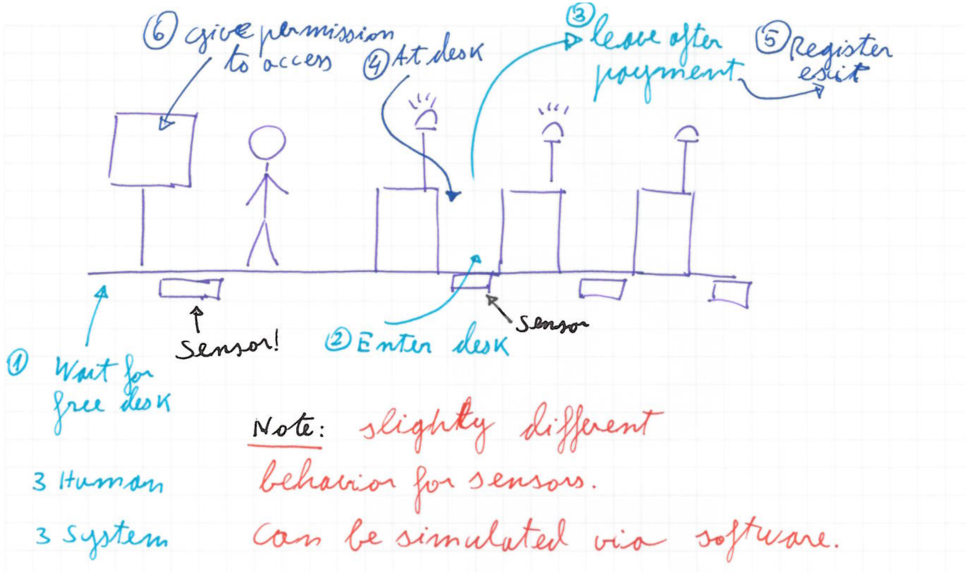
Types of Events



- Environment events reading from internal system variables.
- Environment events writing to system variables.
- System events writing to environment variables altering environment behavior.

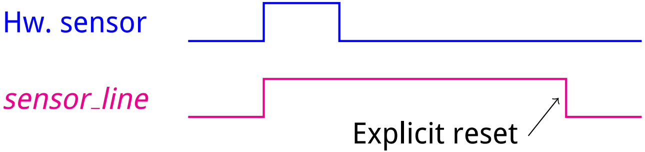
is cheating.

Events in the Model



Understanding Sensors

- Hardware usually on / off when detecting / not detecting.
- Transient behavior: real-time signals may be missed.
  (Unlikely for our case)
- We need to detect clients walking to counter (*sensor_line*).
  - Sensor is *on* as long as clients walking to counter, or
  - Sensor *off* after clients pass, *sensor_line* stays *on* until turned *off*.
    - The latter can be simulated with software.
    - Assume that behavior, do not show code.



- *cashier_sensor*(*k*) is *on* only when client at counter.

Making Sense of Sensor and Screen

		line_sensor False	*line_sensor* True
can_enter	False	No free desk Corridor empty	No free desk Corridor not empty
	True	Free desk Corridor empty	Just entered

- All combinations possible.
- They summarize the immediate previous situation.
- Again, we assume clients wait for corridor to be empty.
- We could force it with a barrier:

$$barrier = OPEN \Leftrightarrow can\_enter = TRUE \land line\_sensor = FALSE$$

Busy Lights and Counter Sensors

- Not in sync.
- That is how it physically happens.
- Every combination has a meaning: captures change over time.

		$busy(k)$ False	$busy(k)$ True
$sensor(k)$	False	Free	Just left
	True	Just arrived	Being served

Some Invariants

If we can enter, there is a free desk

$$can\_enter = TRUE \Rightarrow \exists k \cdot (k \in 1..N\_DESK \land cashier\_busy(k) = FALSE)$$

Note: implication is not causality.

Some Invariants

If we can enter, there is a free desk

$$can\_enter = TRUE \Rightarrow \exists k \cdot (k \in 1..N\_DESK \wedge cashier\_busy(k) = FALSE)$$

Note: implication is not causality.

If we are in the corridor and not yet served, there is a free desk

$$line\_sensor = TRUE \Rightarrow \exists k \cdot (k \in 1..N\_DESK \wedge cashier\_sensor(k) = FALSE)$$

Note: the two above need additional auxiliary invariants (see model).

Some Invariants

If we can enter, there is a free desk

$$can\_enter = TRUE \Rightarrow \exists k \cdot (k \in 1..N\_DESK \land cashier\_busy(k) = FALSE)$$

Note: implication is not causality.

If we are in the corridor and not yet served, there is a free desk

$$line\_sensor = TRUE \Rightarrow \exists k \cdot (k \in 1..N\_DESK \land cashier\_sensor(k) = FALSE)$$

Note: the two above need additional auxiliary invariants (see model).

What about "entering should be disallowed if there is someone in the corridor"?

$$\neg(can\_enter = TRUE \land line\_sensor = TRUE)$$

Remember a previous slide: all combinations *can_enter* and *line_sensor* are legal and have a different meaning.

Remarks

- Absolute time meaningless: cannot rely on relative speed.
- Safety, liveness.
- Auxiliary invariants (remember $n^2 = 1 + 3 + \cdots + (2n+1)$).
- Function initialization.
- Deciding events: observables which change state. Be frugal!
- Modeling physical environment: very often a safe overapproximation.
- Add desk number in screen:
  - Type invariant: $cashier\_number \in 0..N\_DESK$.
  - Gluing invariant: $cashier\_number = 0 \Leftrightarrow can\_enter = FALSE$.
  - $cashier\_number = 0$ means "cannot enter".
  - $cashier\_number :\in \{k | k \in 1..N\_DESK \wedge cashier\_busy(k) = FALSE \wedge cashier\_sensor(k) = FALSE\}$
  - Simple, straighforward refinement.