JACKSON PROBLEM FRAMES and the Composition Problem

Dr Robin Laney

Open University

Composing Problem Frames

Robin Laney, Leonor Barroca,
Michael Jackson, Bashar Nuseibeh

IEEE Requirements Engineering 04,
Kyoto, Japan

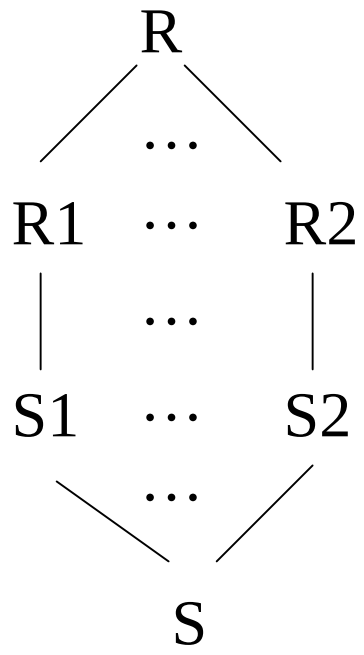
Overview

1. Problem Decomposition
2. Introduce Problem Frames
   - Running Example (Sluice Gate)
3. Introduce Event Calculus
4. Propose approach to Composition Problem

Problem Decomposition

- Make sure we are solving right problem
- Decompose problem rather than solution
- Aim at known sub-problem patterns
  - That we know how to solve
- Recompose solutions to give system
  - Many problems - some addressed here

Decomposition/Recomposition



Decompose

Solve

Compose

S, W => R (Gunter at al Ref Model)

Problems

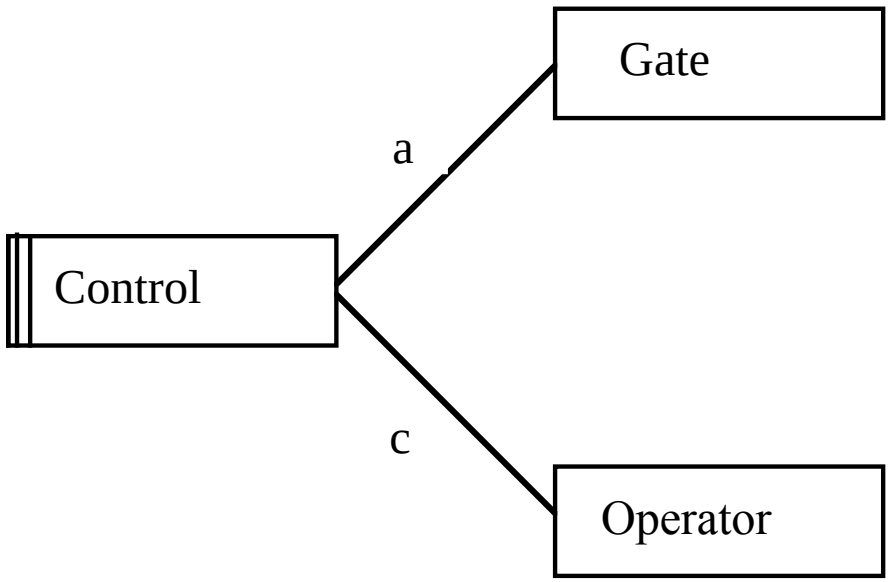
- Inconsistent Requirements
- Inconsistent Specifications
- Inconsistent Models
- Interference Effects on Problem Domains

Sluice Gate Example Requirements

P – The gate should be opened for the first ten minutes of each three-hour period.

OI – The gate should respond to raise, lower and halt commands issued by an operator.

Context Diagram



a: C!{up, down, off}
G!{Top,Bottom}

c:O!{raise,lower,stop}

Sluice Gate Domain (1)

- When stopped, will react to an up event by moving upwards, unless already fully open.
- When stopped, it will react to a down event by moving downwards, unless already fully closed.
- When it receives an off event it will turn its motor off, stopping any motion.

Sluice Gate Domain (2)

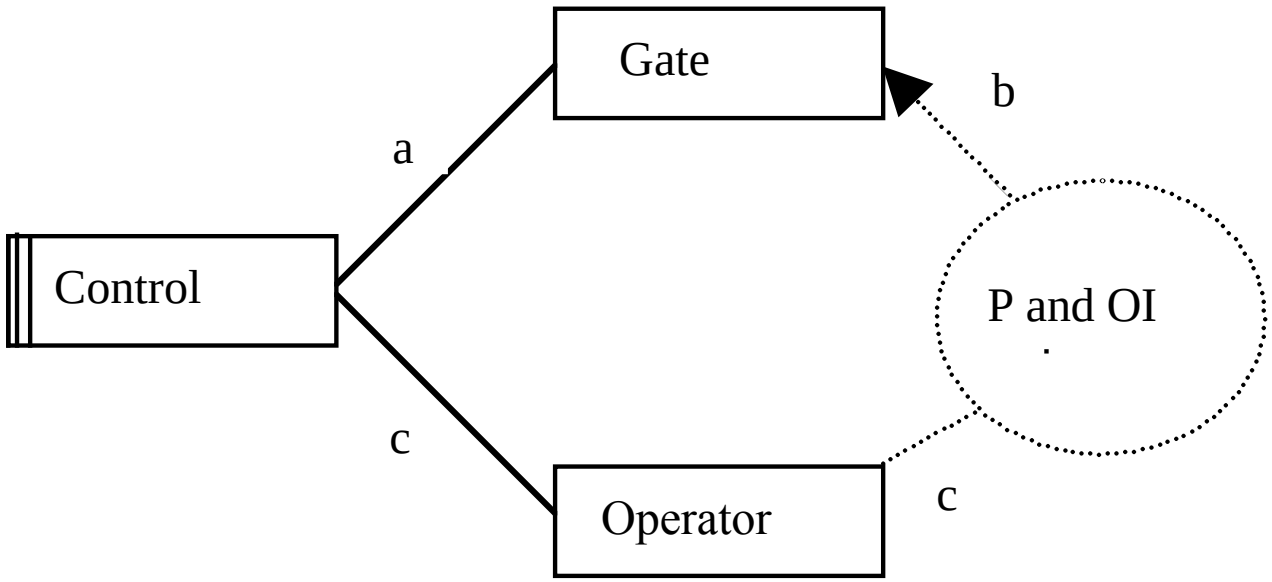
- When it detects that it is fully open it will generate a Top event.

- When it detects that it is fully closed, it will generate a Bottom event.

Operator Domain

- The operator will generate raise commands to request the gate moves upwards,
- lower commands to request the gate moves downwards, and
- halt commands to request the gate stops.

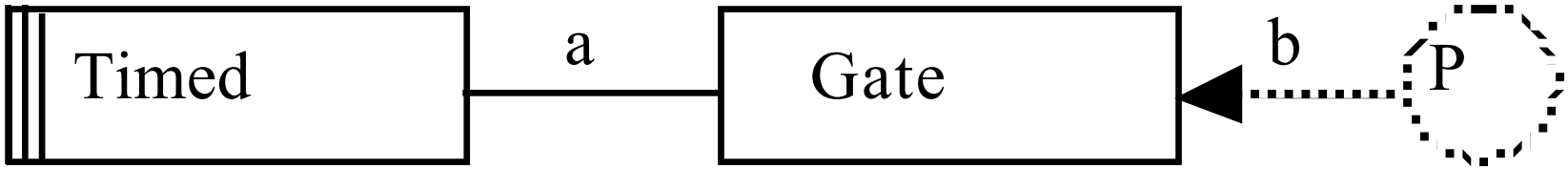
Problem Diagram



a: C!{up, down, off}
G!{Top,Bottom},

b:G!{Open,Shut}
c:O!{raise,lower,stop}

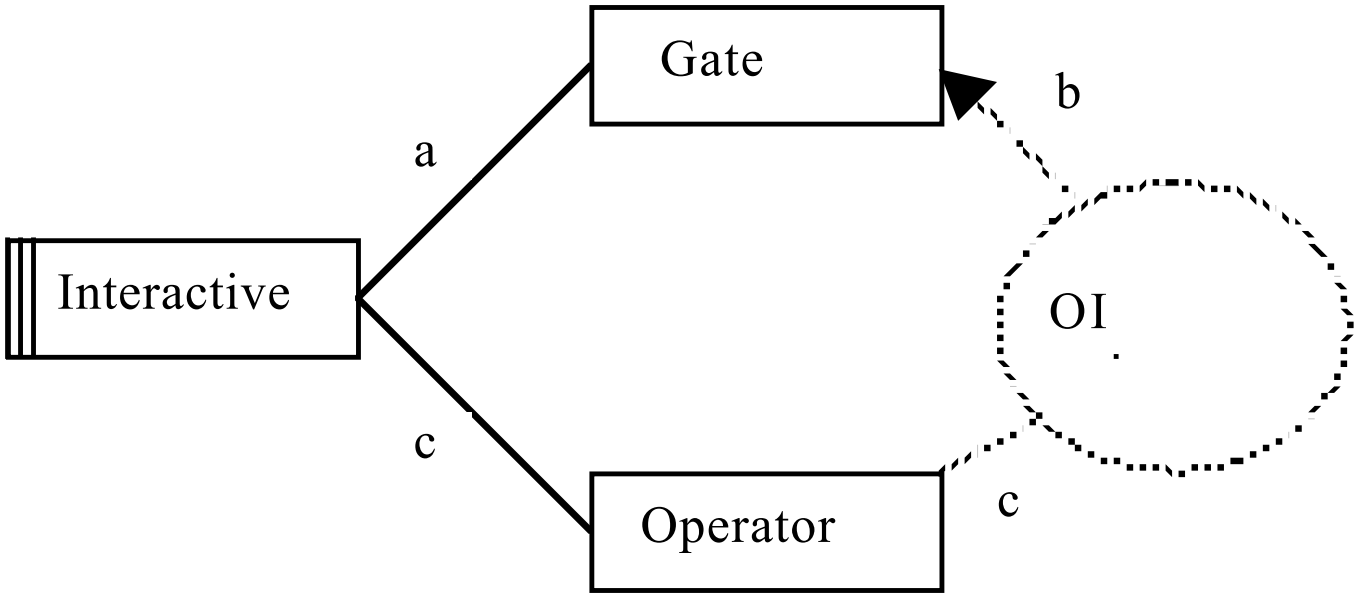
Required Behaviour Frame for Req P



a: T!{up,down,off}
G!{Top,Bottom},

b:G!{Open,Shut}

Commanded Behaviour Frame for Req OI



a: I!{up, down , off}
G!{Top,Bottom},

b:G!{Open,Shut}
c:O!{raise,lower,stop}

Composition Problem

- Problem domains react to machines
- When we compose machines
  - Interference between machines possible
  - Interference experienced by problem domains

Options for Dealing with Interference

1. Allow!
2. Mutual Exclusion
3. Mutual Exclusion with Priorities
4. Mutual Exclusion with Event Priorities

Event Calculus

- Robert Kowalski & Marek Sergot 1986

- Slides Based on:

  The event Calculus Explained

  Murray Shanahan

  Artificial Intelligence Today, ed. M.J.Wooldridge and M.Veloso, Springer Lecture Notes in Artificial Intelligence no. 1600, Springer (1999), pages 409-430.

Ontology

- Actions or Events (a)
- Fluents (f)
- Time Points (t)

- (Variables Universally quantified by default)

What Happens When

- Initially(f)
- Happens(a,t)
- t1 < t2

What Actions Do

- Initiates (a,f,t)
- Terminates(a,f,t)
- Trajectory(f1,t,f2,delta)

What’s true when

- HoldsAt (f,t)
- Clipped(t1,f,t2)

Event Calculus Axioms(1)

$$\text{HoldsAt}(f, t1) \leftarrow \text{Initially}(f) \wedge \neg\, \text{clipped}(0, f, t1)$$

$$\text{HoldsAt}(f, t2) \leftarrow \text{Happens}(a, t1) \wedge \text{Initiates}(a, f, t1)$$
$$t1 < t2 \wedge$$
$$\neg\, \text{clipped}(t1, f, t2)$$

Event Calculus Axioms (2)

$$\begin{aligned}
&\mathrm{HoldsAt}(f2,t3) \leftarrow \\
&\mathrm{Happens}(a,t1) \wedge \mathrm{Initiates}(a,f1,t1) \wedge \\
&\mathrm{Trajectory}(f1,t1,f2,d) \wedge \\
&t2 = t1 + d \wedge t1 < t2 < t3 \wedge \\
&\neg\, \mathrm{clipped}(t1,f1,t2) \wedge \neg\, \mathrm{clipped}(t2,f2,t3)
\end{aligned}$$

Event Calculus Axioms(3)

$$\text{Clipped}(t1, f, t2) \leftrightarrow \exists\, a, t$$
$$[\text{Happens}(a, t) \wedge t1 < t < t2 \wedge \text{Terminates}(a, f, t)]$$

Sluice Gate Phenomena

- Actions

  up,down, off [Machine]

  raise, lower, stop [Operator]

- Events
  - Top, Bottom [Gate]
- Fluents
  - MovingUp, MovingDown, [New!]
  - Stopped
  - Open, Shut [Requirement]

Sluice Gate Description

Initiates(up,MovingUp,t) ← HoldsAt(Stopped,t) (G1)
Initiates(down,MovingDown,t) ← HoldsAt(Stopped,t) (G2)
Initiates (off,Stopped,t) (G3)
Initiates (top,Open,t) (G4)
Initiates (bottom,Shut,t) (G5)

Sluice Gate : Common Sense?

Trajectory(MovingUp,t,Open,sufftime)	(G6)
Trajectory(MovingDown,t,Shut,sufftime)	(G7)
Terminates(down,Open,t)	(G8)
Terminates(up,Shut,t)	(G9)
Terminates(off,MovingUp,t)	(G10)
Terminates(off,MovingDown,t)	(G11)

Requirement (P)

Making Sure the Gate is Open at Start of three hour period and

Keep Gate open for first ten minutes of three hour period

$$HoldsAt(Open,t1) \wedge \neg\ clipped(t1,Open,t1+10) \wedge t1 \bmod 180 = 0$$

Prohibiting Events

Formula	Meaning
prohibit($\alpha, \tau 1, \tau 2$)	Event α should not occur between times $\tau 1$ and $\tau 2$

$$\text{prohibit}(a, t1, t2) \Leftrightarrow \neg \exists\, a, t\ (\text{Happens}(a, t) \wedge\ t1 < t < t2)$$

Specification (Timed)

((Initially(Open) ∧ prohibit(down,0,t1) ∧ t1 mod 180 = 0)

OR

(Happens(Top,t) ∧ prohibit(down,t,t2) ∧

t2 mod 180 = 0 ∧ t < t2)

OR …

Specification Continued (Timed)

$$(\text{Happens}(up, t1) \wedge \text{Happens}(off, t1-1) \wedge$$

$$t2 = t1 + sufftime \wedge t1 < t2 < t \wedge$$

$$\text{prohibit}(off, t1, t2) \wedge \neg \text{prohibit}(down, t2, t)$$

$$\wedge\ t \bmod 180 = 0))$$

AND

$$(\text{prohibit}(t1, down, t1+10) \wedge t1 \bmod 180 = 0)$$

Requirement OI

∃ d . Happens(raise,t) ->HoldsAt(t+d,MovingUp) /\

∃ d . Happens(lower,t) ->HoldsAt(t+d,MovingDown) /\

∃ d . Happens(halt,t) -> HoldsAt(t+d,Stopped)

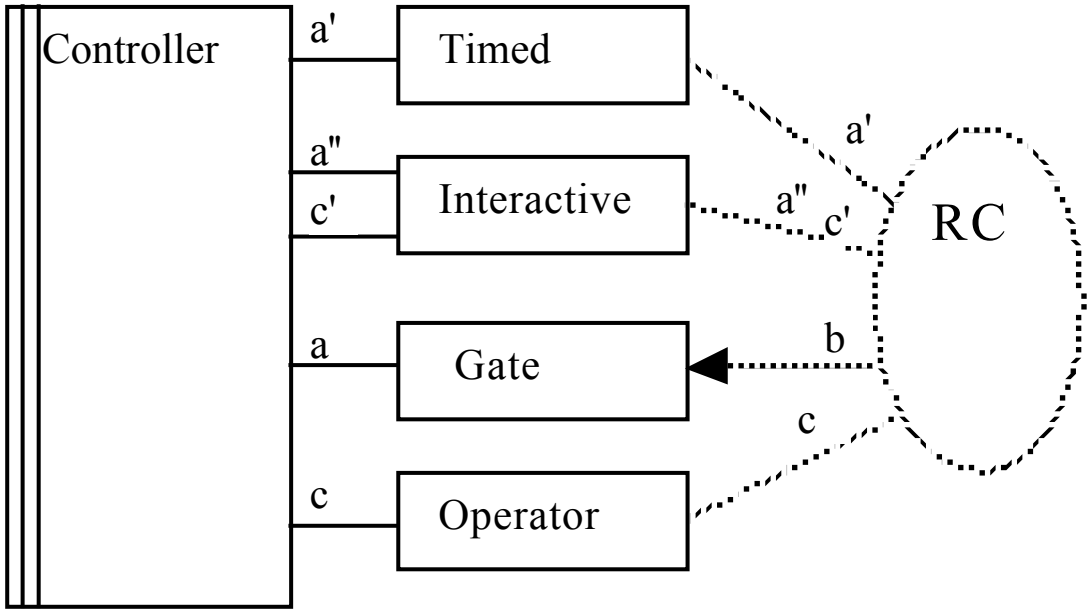
Specification (Operated)

(Happens(raise,t) -> Happens(off,t+1) /\
Happens(up,t+2)) /\

(Happens(lower,t) -> Happens(off,t+1) /\
Happens(down,t+2)) /\

(Happens(halt,t) -> Happens(off,t+1))

Composition Frame



a: C!{up, down, off}
G!{Top,Bottom},
a'': I!{up, down, off,prohibit(…)}
C!{Top,Bottom}
b:G!{Open,Shut}

a': T!{up, down, off,prohibit(…)}
C!{Top,Bottom},
c:O!{raise,Lower,stop}
c':C!{raise,lower,stop}

Related Work

- Inconsistency Management
  - Robinson et al.
- Managing Conflicts
  - Van Lamsweerde et al.
- Feature Interaction
  - Zave
- Architectural Connectors
  - Allen and Garlan
- Abductive Approaches and Events
  - Garcez, Russo, Miller, Nuseibeh, Kramer

Frame Problem

Scenario adapted from Hanks & Mcdermott 87

- Filling a Bucket and Dipping a Cloth
  - HappensAt(fill,1) (fill initiates Full)
  - HappensAt(sneeze,2)
  - HappensAt(dip,3) (dip and Full initiates Wet)
  - HoldsAt(Wet,4) ???
- Solution
  - Inertial Assumption (fluents persist…)
  - Uniqueness of names