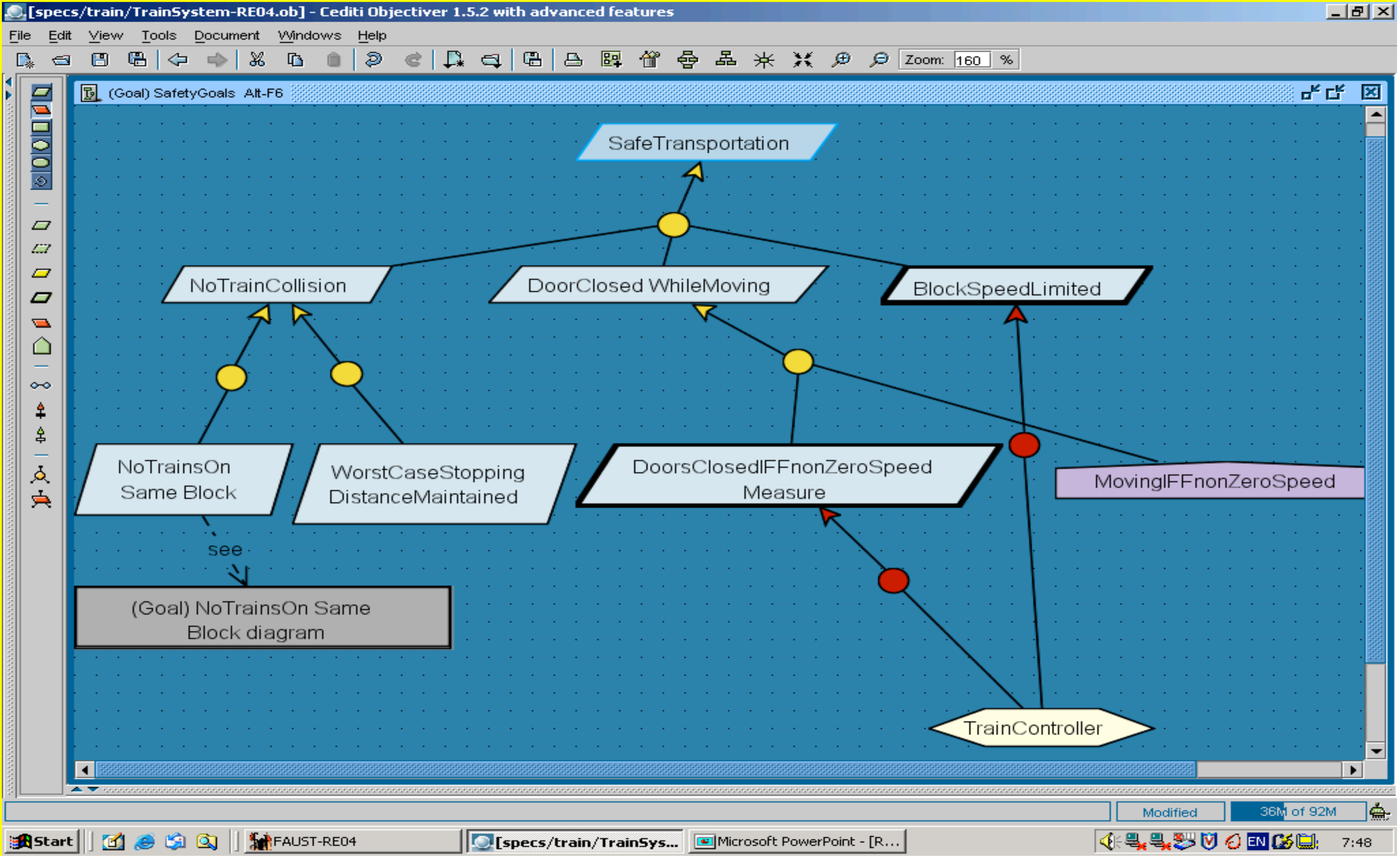
Goal Diagrams, continued

Informatics 113

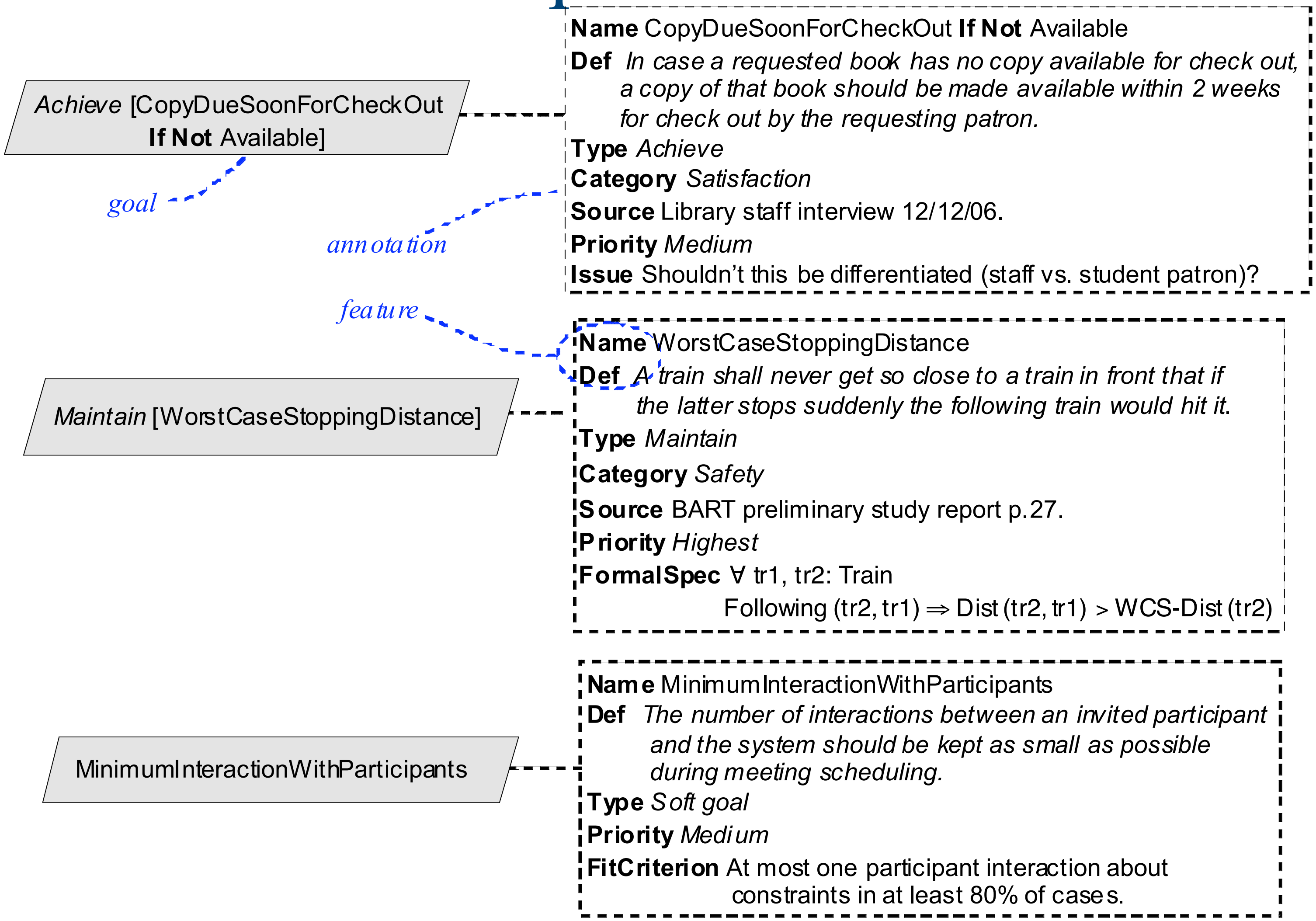
Example Goal Diagram


[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Zoom: 160 %
(Goal) SafetyGoals Alt-F6
SafeTransportation
NoTrainCollision
DoorClosed WhileMoving
BlockSpeedLimited
NoTrainsOn Same Block
WorstCaseStopping DistanceMaintained
DoorsClosedIFFnonZeroSpeed Measure
MovingIFFnonZeroSpeed
see
(Goal) NoTrainsOn Same Block diagram
TrainController
Modified
36M of 92M
Start
FAUST-RE04
[specs/train/TrainSys...
Microsoft PowerPoint - [R...
EN
7:48

Symbology

- Goals: Parallelograms
- “And” nodes: circles
  - Black circles, if the and-refinement is complete
- Or: independent arrows
- Agents: hexagons
  - A stick figure inside, if part of the environment
- Domain assumptions: trapezoid (or “house symbol”)

Details of Goal provided in Annotations



Achieve [CopyDueSoonForCheckOut **If Not** Available]

Name CopyDueSoonForCheckOut **If Not** Available
Def *In case a requested book has no copy available for check out, a copy of that book should be made available within 2 weeks for check out by the requesting patron.*
Type *Achieve*
Category *Satisfaction*
Source Library staff interview 12/12/06.
Priority *Medium*
Issue Shouldn't this be differentiated (staff vs. student patron)?

Maintain [WorstCaseStoppingDistance]

Name WorstCaseStoppingDistance
Def *A train shall never get so close to a train in front that if the latter stops suddenly the following train would hit it.*
Type *Maintain*
Category *Safety*
Source BART preliminary study report p.27.
Priority *Highest*
FormalSpec $\forall$ tr1, tr2: Train
Following (tr2, tr1) $\Rightarrow$ Dist (tr2, tr1) > WCS-Dist (tr2)

MinimumInteractionWithParticipants

Name MinimumInteractionWithParticipants
Def *The number of interactions between an invited participant and the system should be kept as small as possible during meeting scheduling.*
Type *Soft goal*
Priority *Medium*
FitCriterion At most one participant interaction about constraints in at least 80% of cases.

Figure 8.1 – Goal features as model annotations: examples

Goal Annotations

- Required:
  - – Name
  - – Definition: “must precisely define, in natural language, what the goal prescribes”
    - In other words, the “shall statement”
- Optional
  - – Category, Source, Priority, Stability (how likely is this goal to change?), FitCriterion, FormalSpec, Issues (duh?!)

Goal Refinement

- An AND-refinement of goal G into subgoals $G_1, ..., G_n$ states that G can be satisfied by satisfying $G_1, ..., G_n$

  The set $\{G_1, ..., G_n\}$ is called **refinement** of G

  Subgoal G_i is said to contribute positively to G

- An OR-refinement of goal G into refinements $R_1, ...,$ R_m states that G can be satisfied by satisfying all subgoals from any of the alternative refinements R_i
- Alternative goal refinements yield different system proposals (variants)
  - Different systems
  - Different responsibility assignments (agents)
- Pros/cons to be evaluated against soft goals for selection of best option

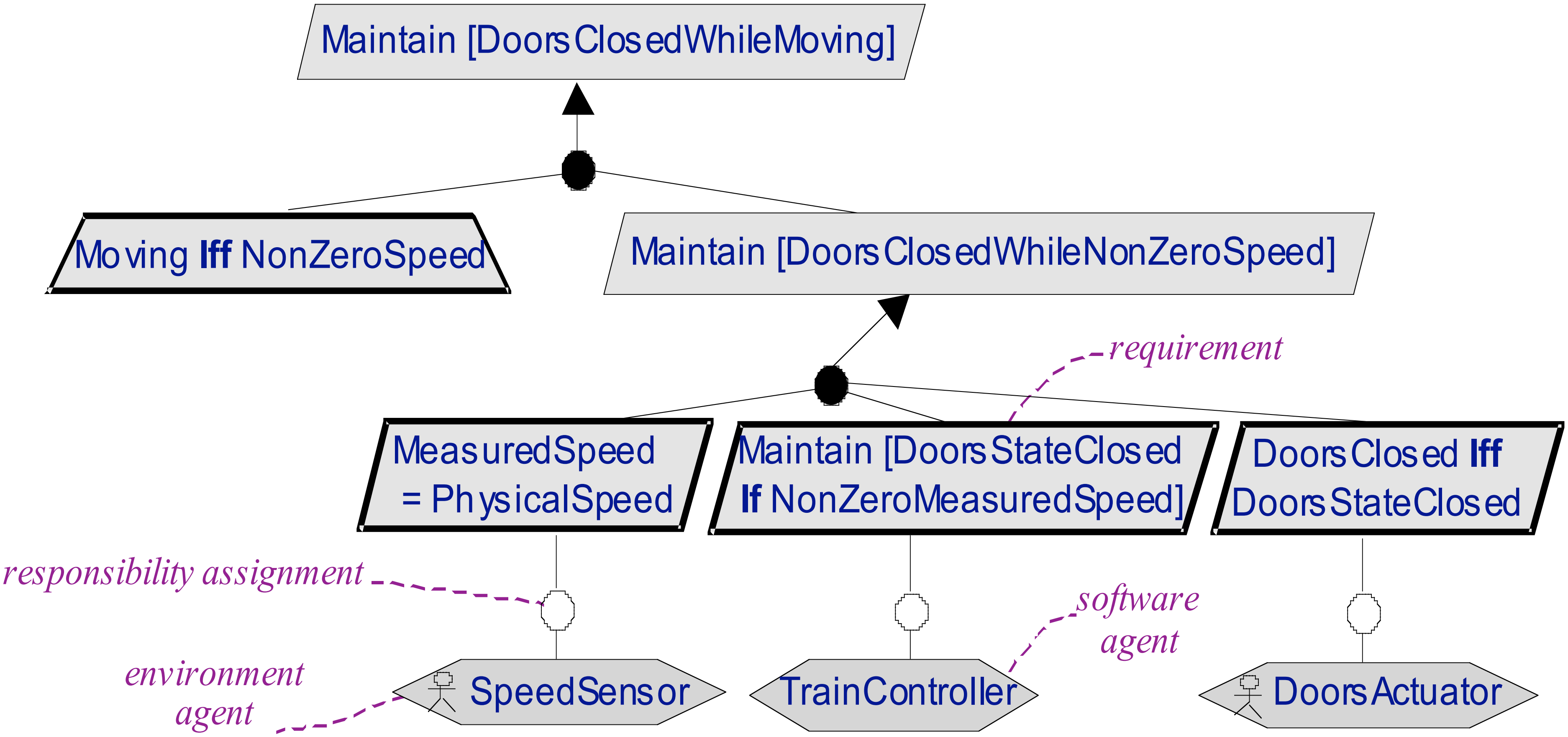
AND-refinements

- Should be complete
  - Sufficient to satisfy in view of known domain properties
    - (is sufficient under the following invariants or hypotheses)
- Should be minimal
  - If you leave one out, no longer a sufficient set
- Should be consistent (duh!)

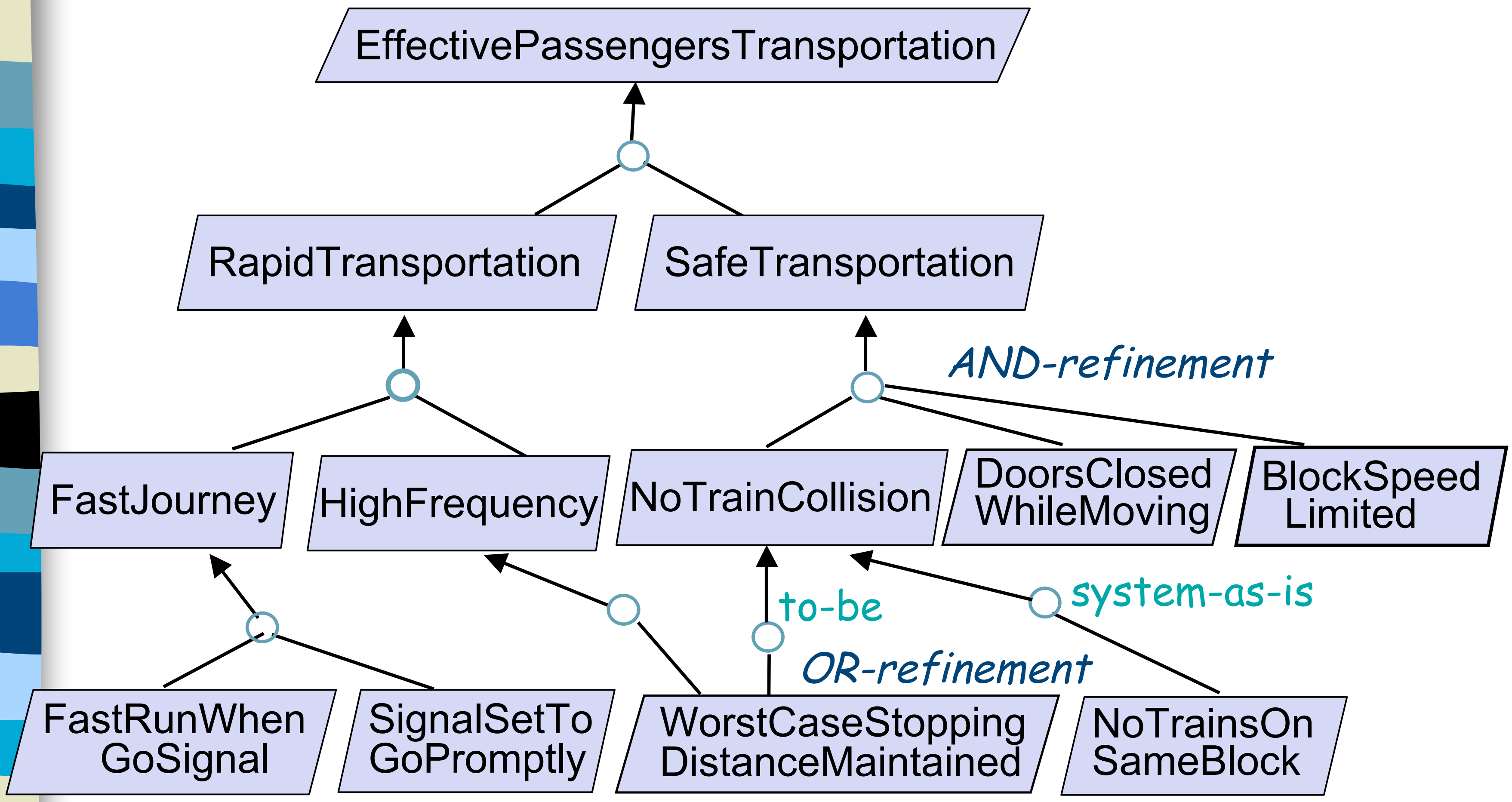
Leaf nodes: How far do you go?

- Leaf nodes: no further refinement necessary
  1. Requirements (goals assignable to a single software agent)
  2. Expectations (goals assignable to a single environmental agent)
- “Assignable” == agent is responsible (and capable) of realizing the goal

Example Refinement to Leaf Nodes


Maintain [DoorsClosedWhileMoving]
Moving **Iff** NonZeroSpeed
Maintain [DoorsClosedWhileNonZeroSpeed]
requirement
MeasuredSpeed = PhysicalSpeed
Maintain [DoorsStateClosed **If** NonZeroMeasuredSpeed]
DoorsClosed **Iff** DoorsStateClosed
responsibility assignment
software agent
environment agent
SpeedSensor
TrainController
DoorsActuator

AND/OR Graphs



EffectivePassengersTransportation

RapidTransportation

SafeTransportation

FastJourney

HighFrequency

NoTrainCollision

DoorsClosed WhileMoving

BlockSpeed Limited

FastRunWhen GoSignal

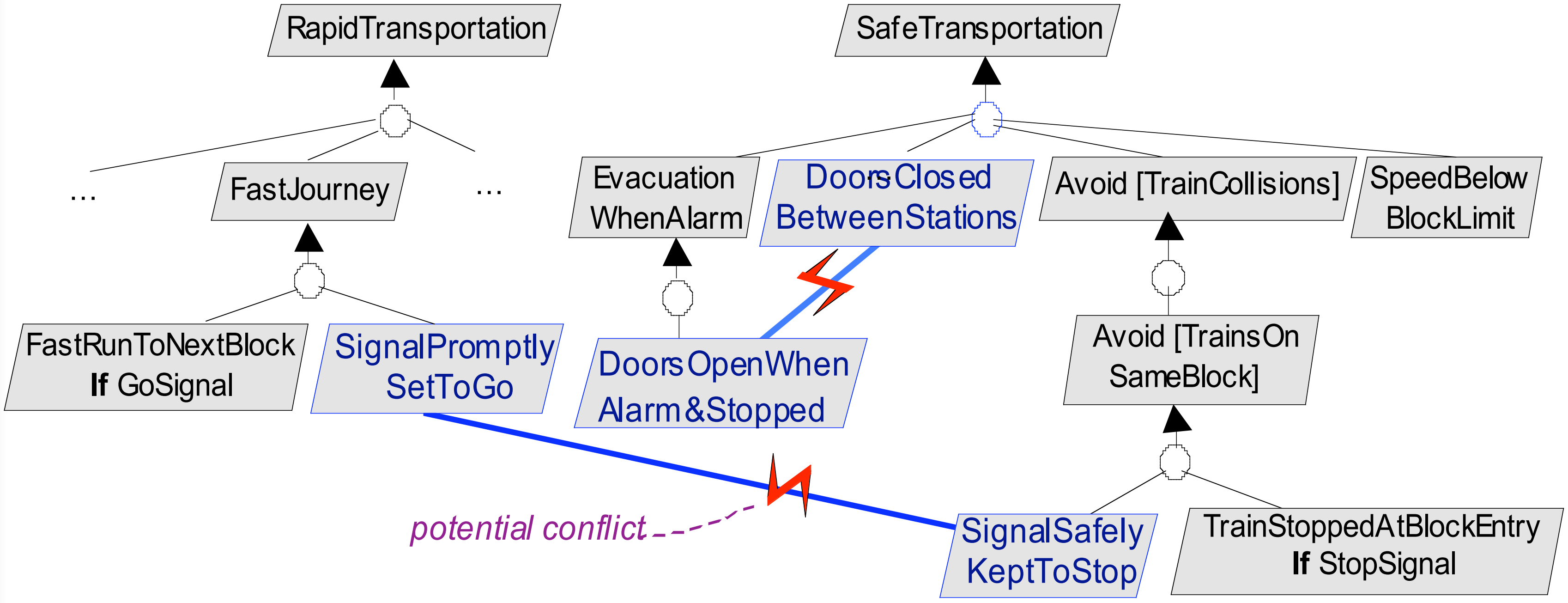
SignalSetTo GoPromptly

WorstCaseStopping DistanceMaintained

NoTrainsOn SameBlock

Conflicts Among goals

- When goals are unsatisfiable together under some condition
- Symbology: lightening bolt between them



Heuristics for finding goals: H1

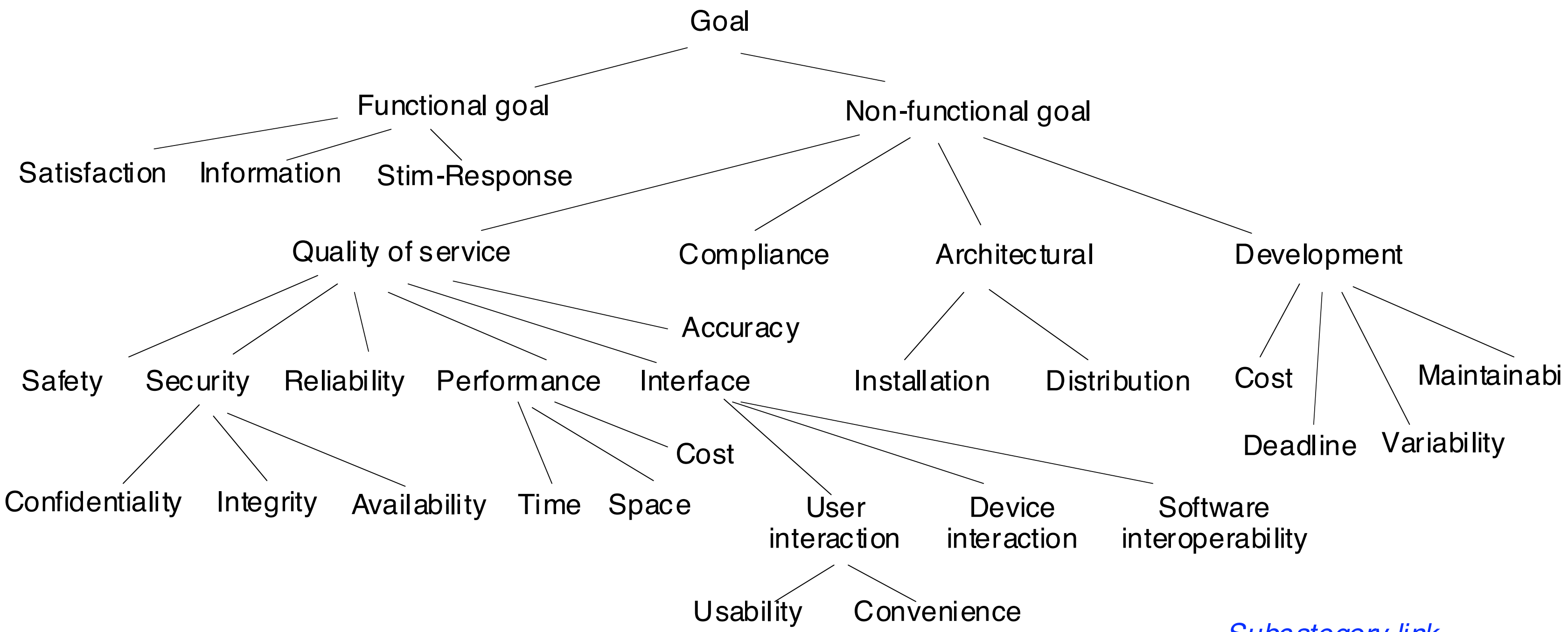
- (H1) Analyze current objectives & problems in system-as-is ...
  - preserve strategic, organization-specific objectives & policies
    - e.g. Effective access to state-of-the-art knowledge
  - preserve application-specific objectives to be found in any system version
    - e.g. Accurate book classification
  - analyze problems & deficiencies in system-as-is
    - Avoid / Reduce / Improve them
      - e.g. Anywhere anytime biblio search

H2: Search for goal-related keywords in elicitation material

- intentional: in order to, so as to, so that, purpose, objective, aim, achieve, maintain, avoid, ensure, guarantee, want, motivate, expect,...
- prescriptive: shall, should, must, has to, to be, may not, may never,...
- amelioration: improve, increase, decrease, reduce, enhance, enable, support, provide, ...

H3: Use the goal categories

- Look for instances of these types



Goal

Functional goal

Non-functional goal

Satisfaction

Information

Stim-Response

Quality of service

Compliance

Architectural

Development

Accuracy

Safety

Security

Reliability

Performance

Interface

Installation

Distribution

Cost

Maintainabi

Cost

Deadline

Variability

Confidentiality

Integrity

Availability

Time

Space

User interaction

Device interaction

Software interoperability

Usability

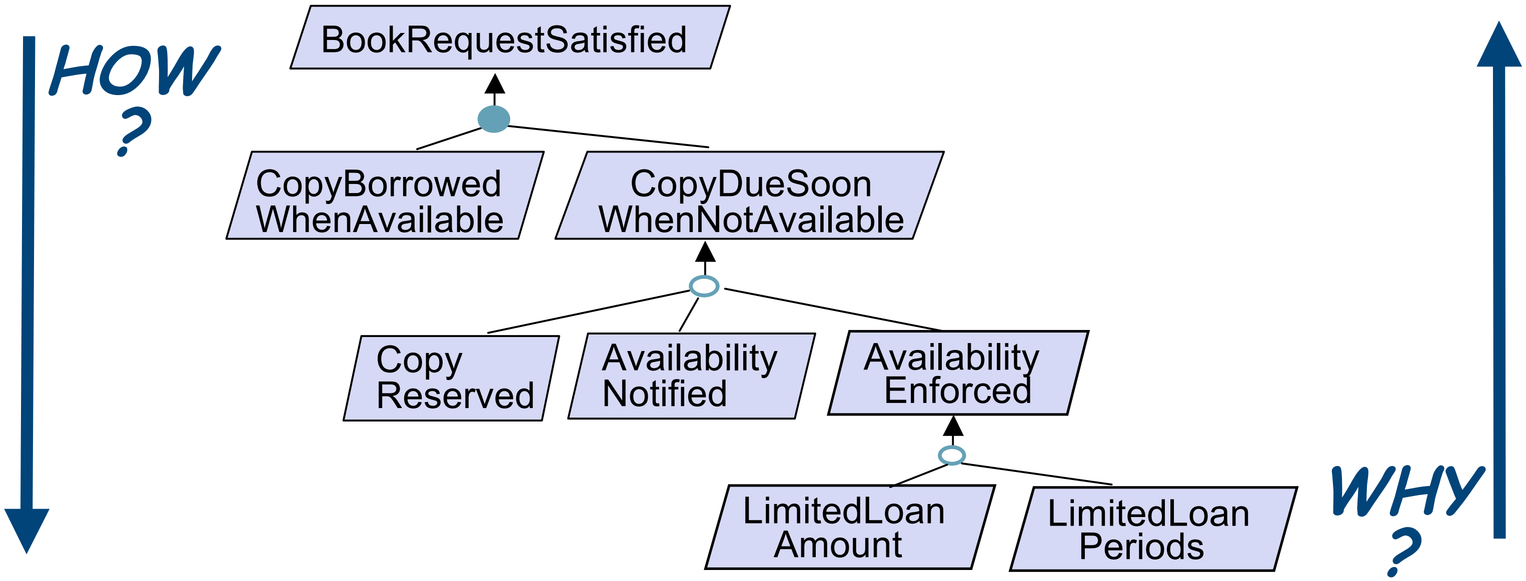
Convenience

Subcategory link

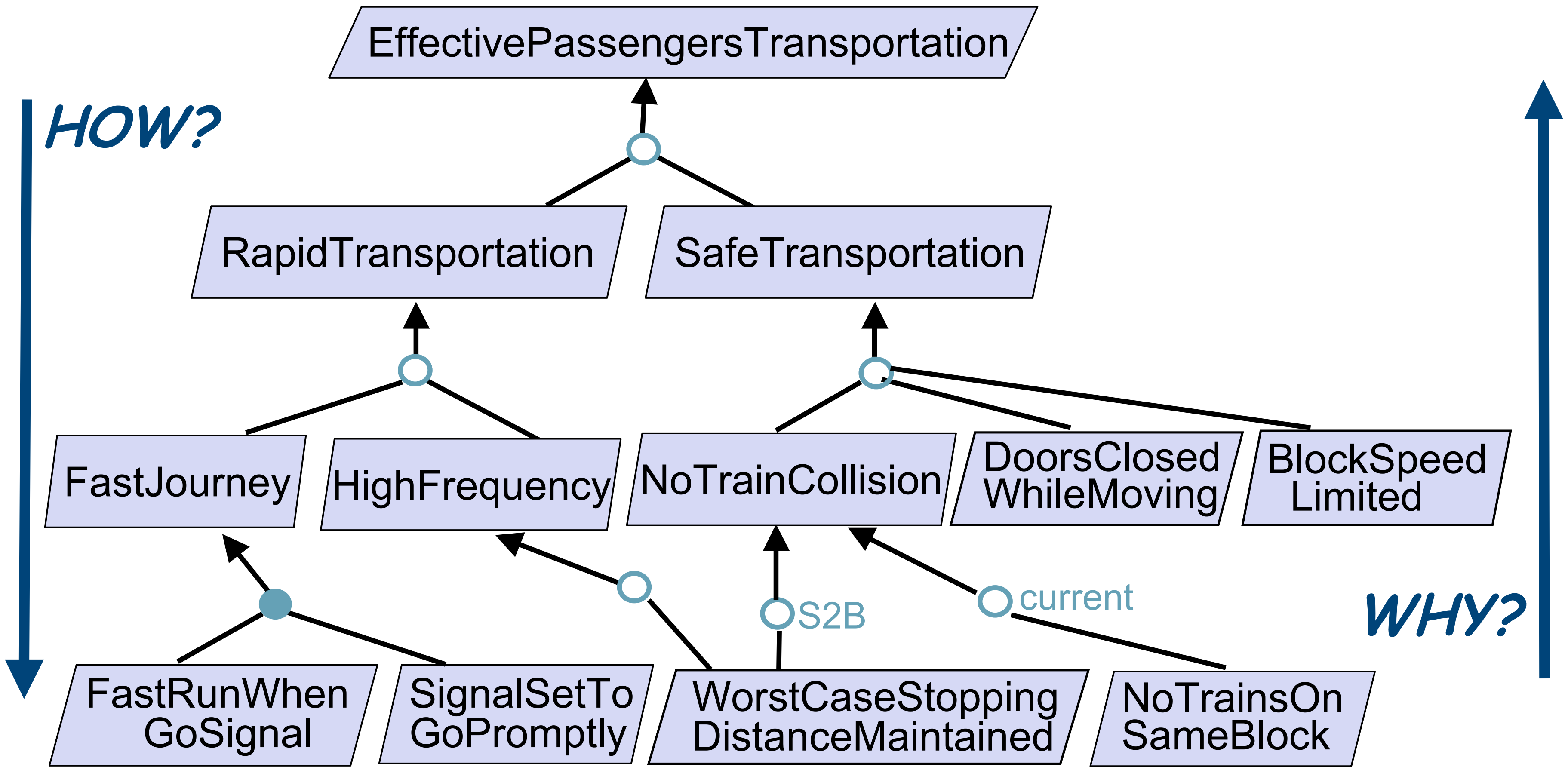
Figure 7.5 – Goal categories

H4: Ask How and Why Questions

- How can G be satisfied? (refinement)
- Why should G be satisfied?(abstraction)

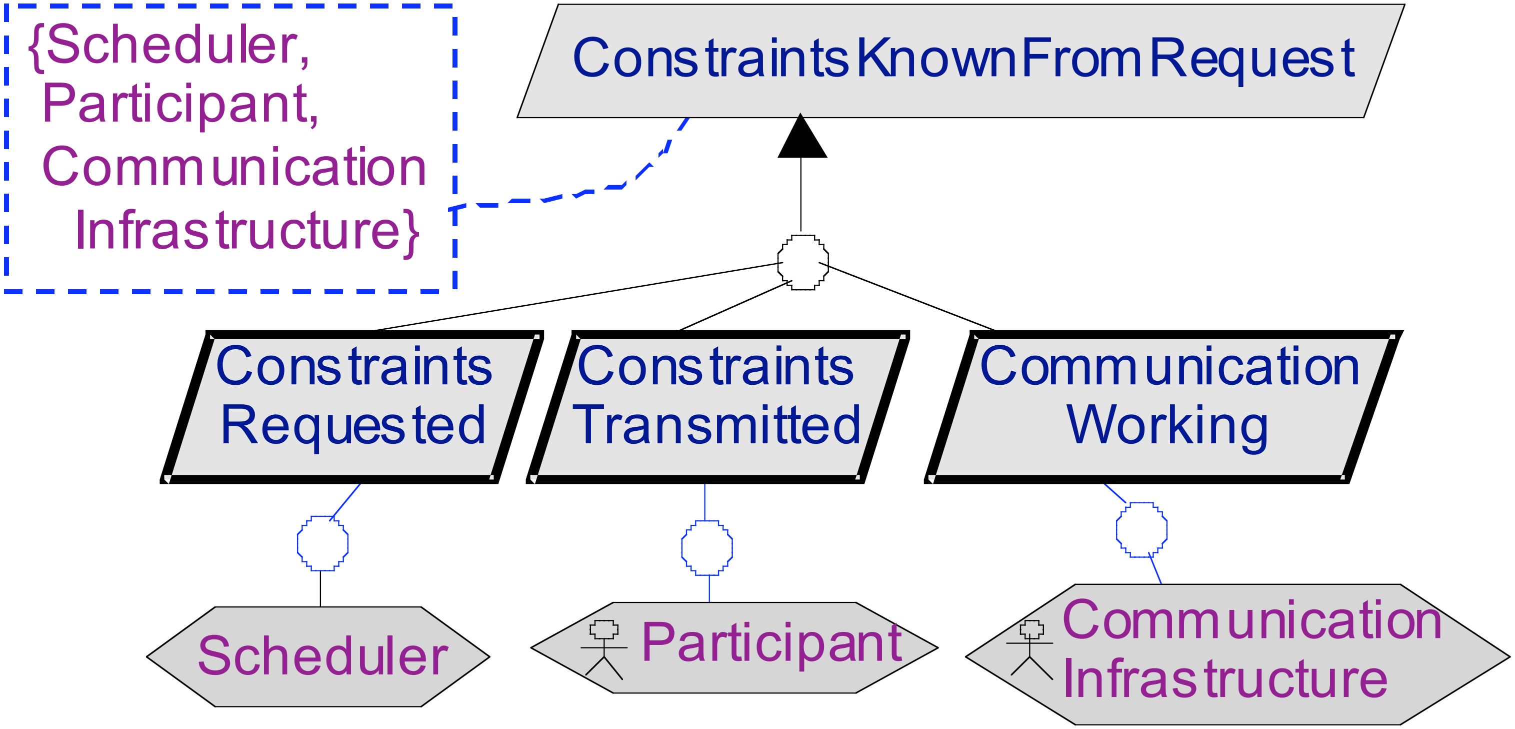


How and Why: System-to-Be and Current



H5: Split Responsibilities

- By examining the contributions of the multiple agents supporting a goal G, you may identify subgoals, each associated with a single agent


{Scheduler, Participant, Communication Infrastructure}
ConstraintsKnownFromRequest
Constraints Requested
Constraints Transmitted
Communication Working
Scheduler
Participant
Communication Infrastructure

H9: Check the converse of Achieve goals

- Check the converse of Achieve goal for missing Maintain goal
  - Achieve [Target If Condition]:
    - if Condition then sooner-or-later Target
  - ?? Maintain [Target OnlyIf Condition]:
    - always (if Target then Condition)
- Example:
  - Achieve [reverseThrustEnabled If PlaneOnGround]
  - ?? Maintain [reverseThrust OnlyIf PlaneOnGround]

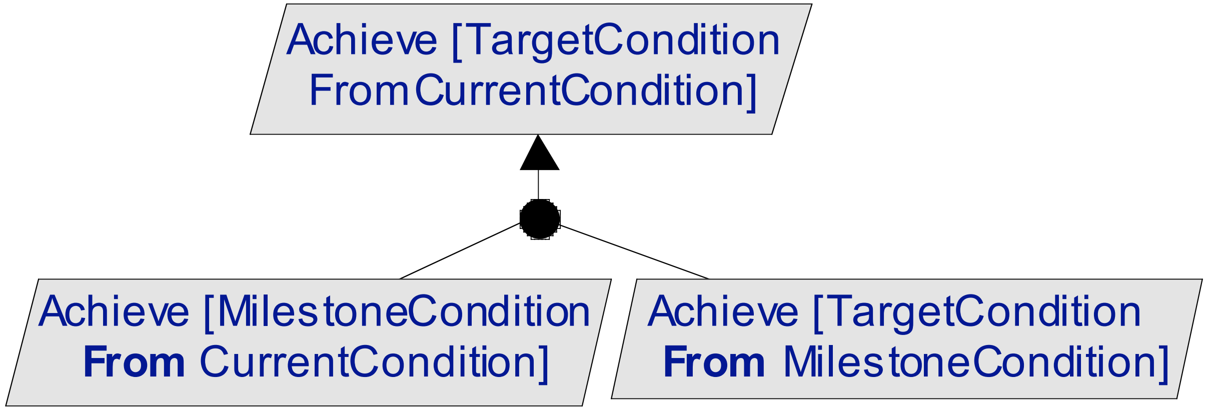
H13: Do not confuse goals with operations

- A goal captures an objective the system should satisfy
- An operation captures a functional service that the system provides to satisfy an objective
- (Watch the verb tense)
  - Goals: past participles (CopyBorrowed)
  - Operation: infinitive (BorrowCopy)
- Goals: entire sequences of states
- Operations: single state transition

Refinement patterns

- Just as with programming patterns, use experience in a particular, careful way to solve frequently recurring problems
- Domain-independent solutions (logic based)
- Domain-specific solutions

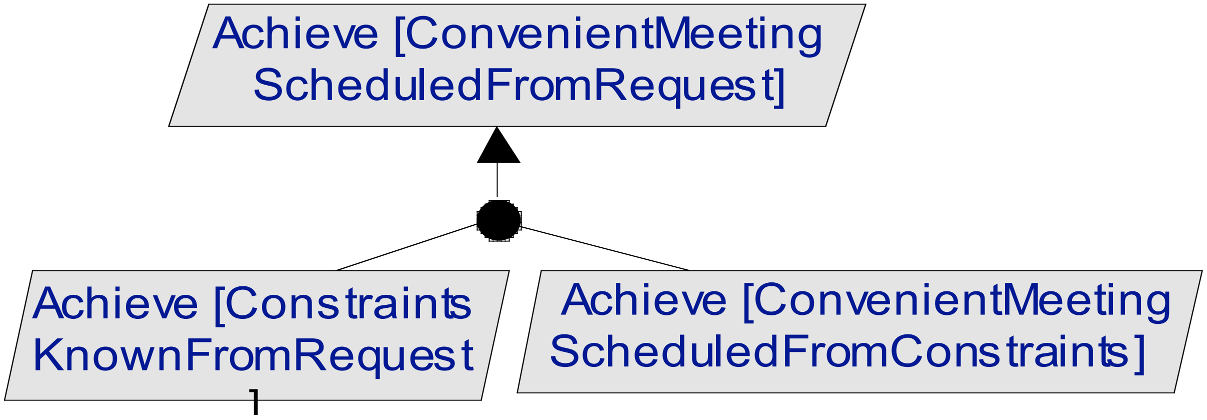
RP1: Baby Steps (“Milestone”)



Achieve [TargetCondition FromCurrentCondition]

Achieve [MilestoneCondition **From** CurrentCondition]

Achieve [TargetCondition **From** MilestoneCondition]

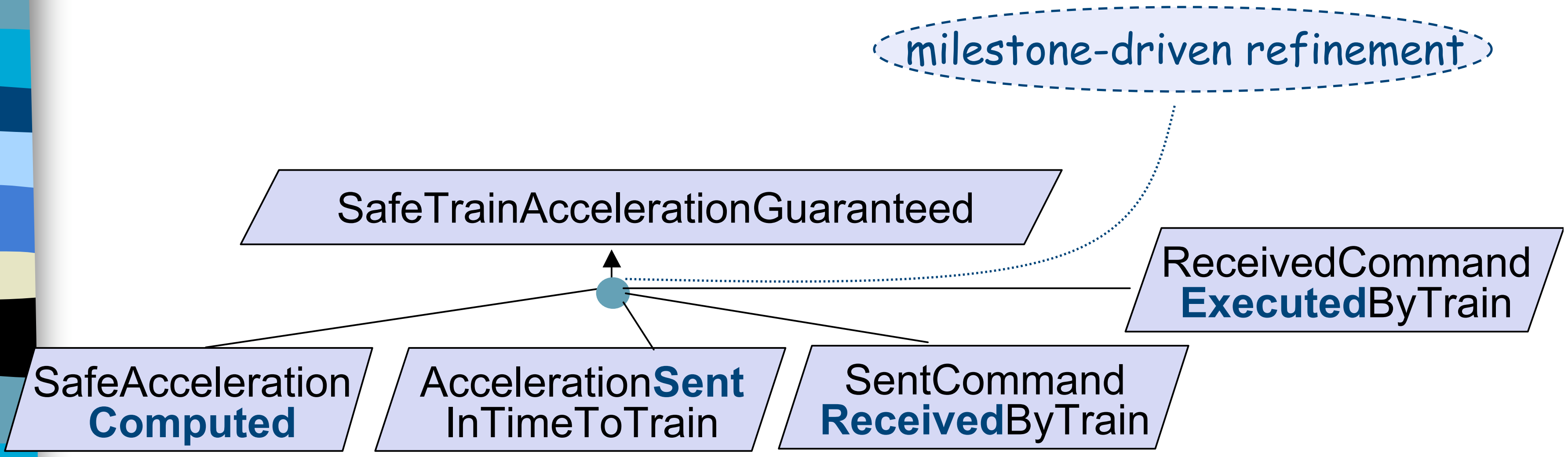


Achieve [ConvenientMeeting ScheduledFromRequest]

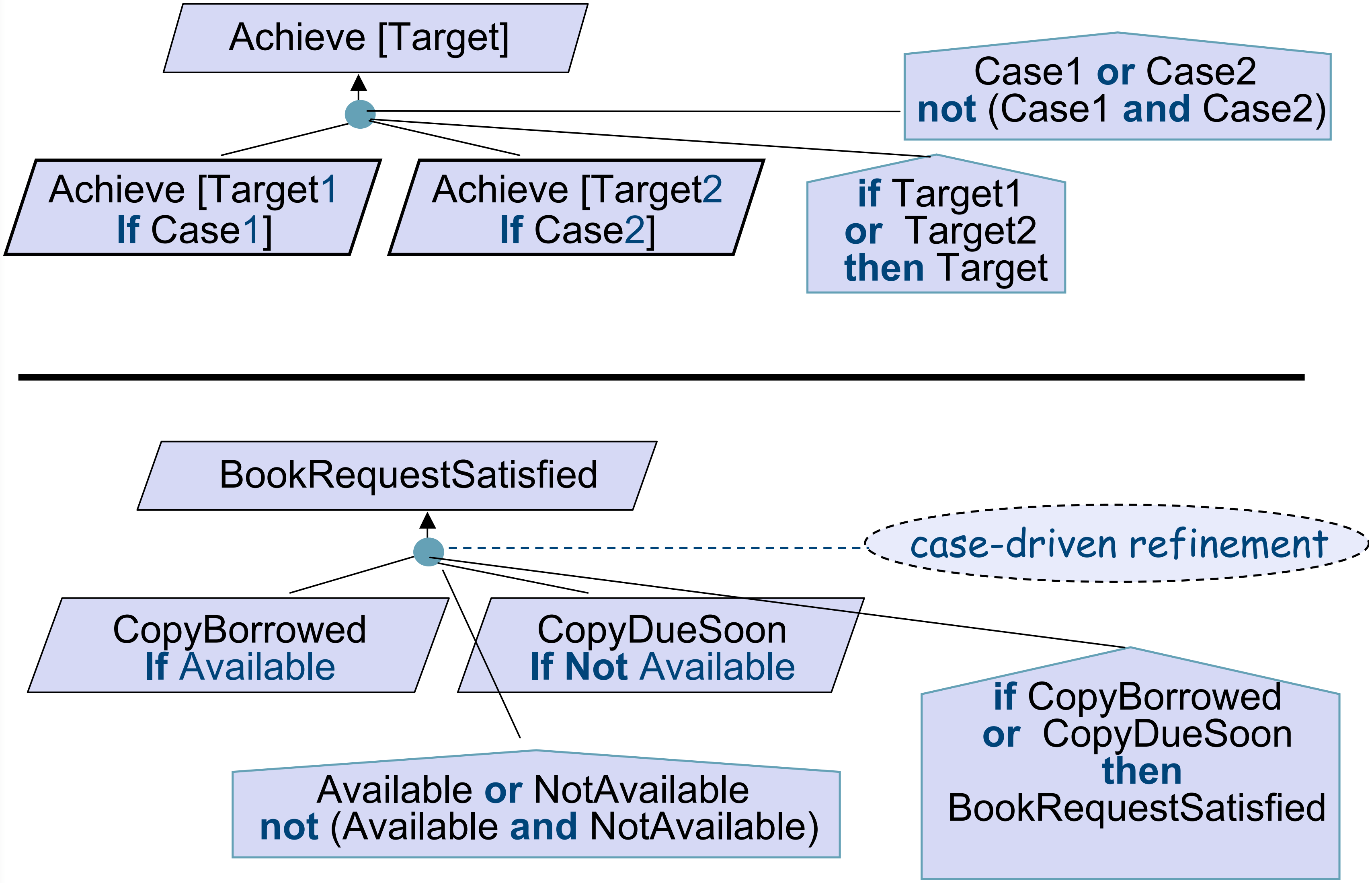
Achieve [Constraints KnownFromRequest]

Achieve [ConvenientMeeting ScheduledFromConstraints]

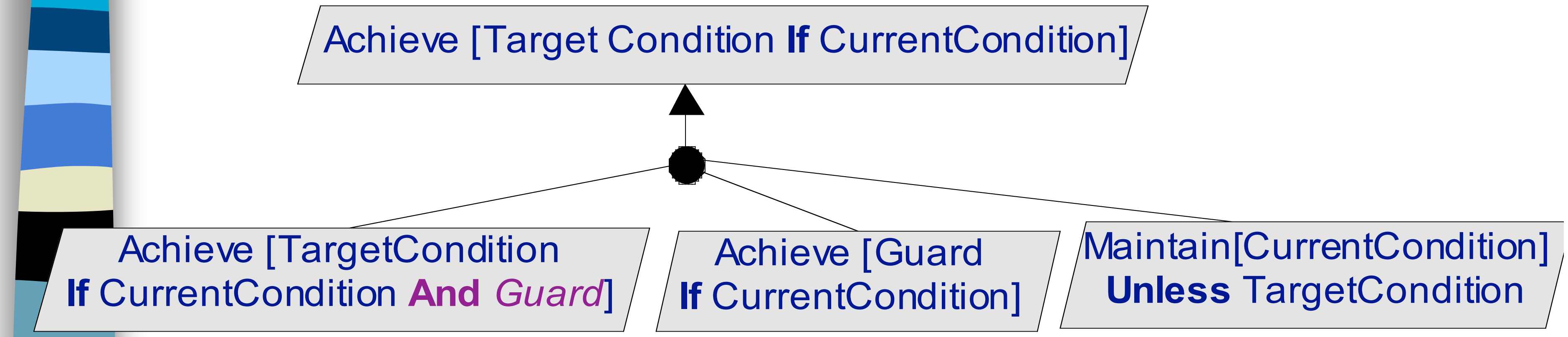
RP1: with many baby steps


milestone-driven refinement
SafeTrainAccelerationGuaranteed
ReceivedCommand **Executed**ByTrain
SafeAcceleration **Computed**
Acceleration**Sent** InTimeToTrain
SentCommand **Received**ByTrain

RP2: Case-based (many allowed)



RP3: Guards are your helpers



Achieve [Target Condition **If** CurrentCondition]

Achieve [TargetCondition
If CurrentCondition **And** *Guard*]

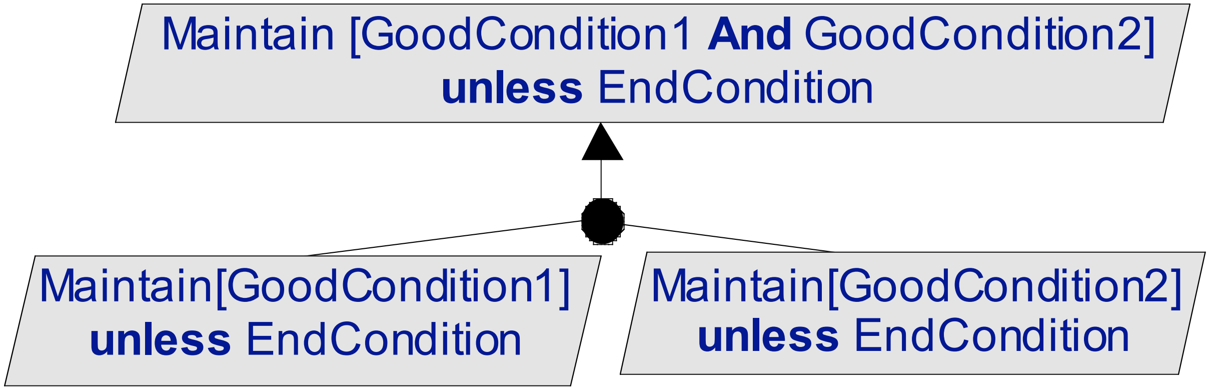
Achieve [Guard
If CurrentCondition]

Maintain[CurrentCondition]
Unless TargetCondition

Get to the intermediate point, where the Guard is true

Stay there long enough that you can reach the Target

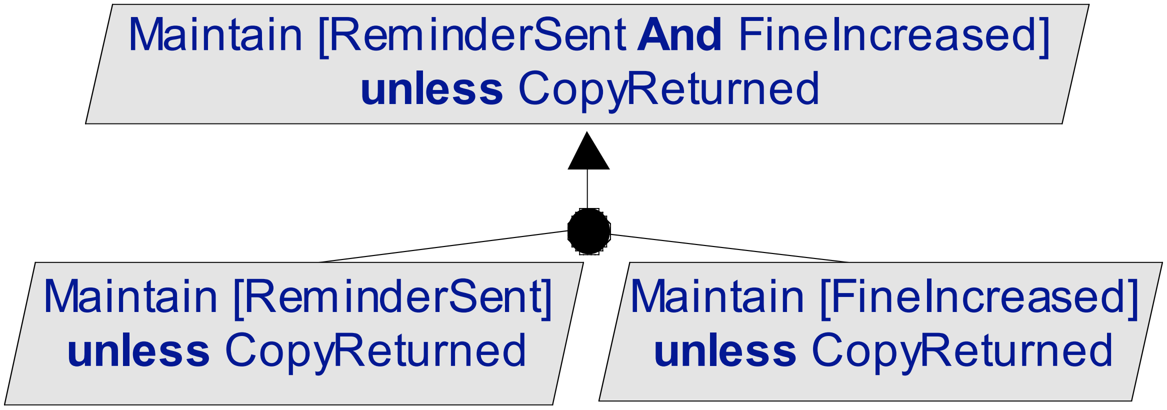
RP4: Simple divide-and-conquor



Maintain [GoodCondition1 **And** GoodCondition2]
unless EndCondition

Maintain[GoodCondition1]
unless EndCondition

Maintain[GoodCondition2]
unless EndCondition



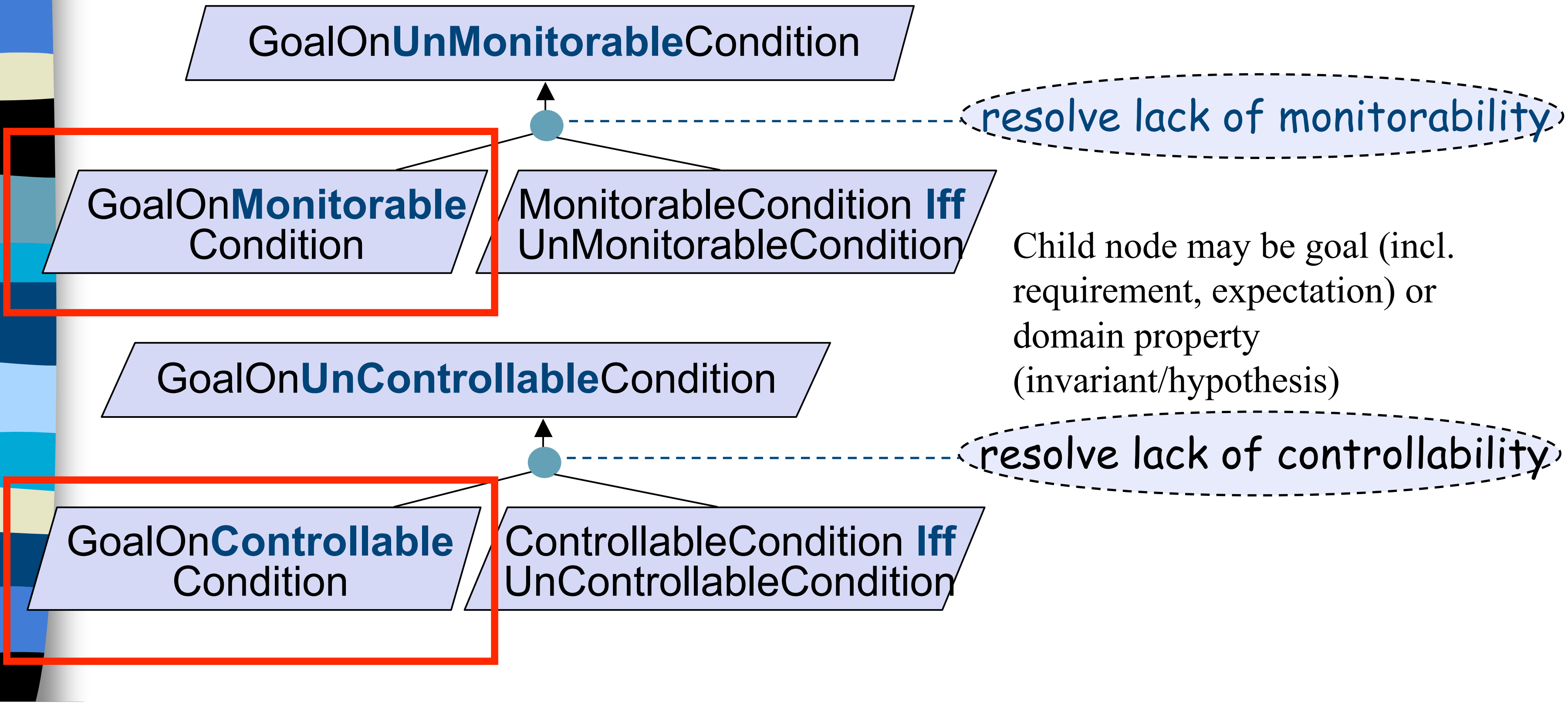
Maintain [ReminderSent **And** FineIncreased]
unless CopyReturned

Maintain [ReminderSent]
unless CopyReturned

Maintain [FineIncreased]
unless CopyReturned

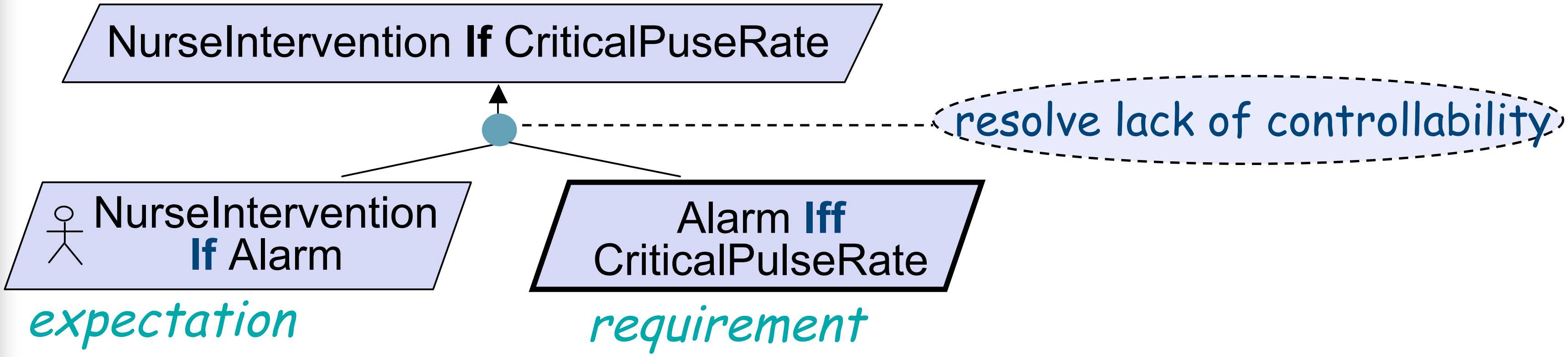
RP5&6: Monitor/Control

- Applicable when the goal refers to quantities not monitorable or not controllable by candidate agent


GoalOnUnMonitorableCondition
resolve lack of monitorability
GoalOnMonitorable Condition
MonitorableCondition Iff UnMonitorableCondition
GoalOnUnControllableCondition
resolve lack of controllability
GoalOnControllable Condition
ControllableCondition Iff UnControllableCondition

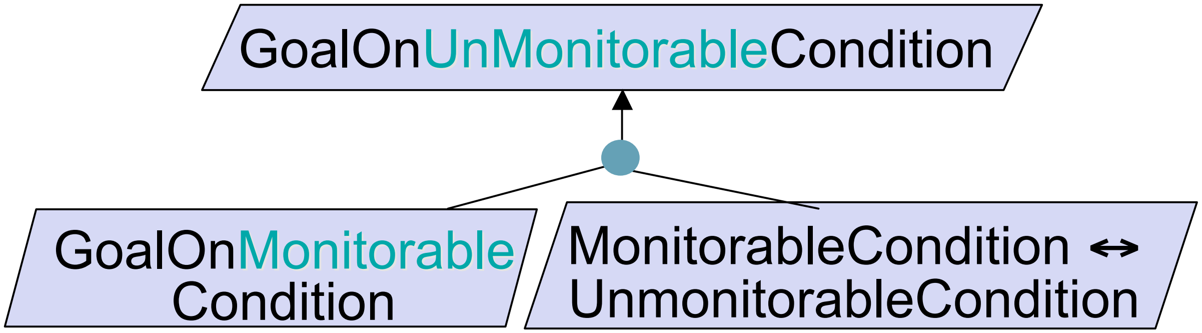
Child node may be goal (incl. requirement, expectation) or domain property (invariant/hypothesis)

RP6: Controlling Example


NurseIntervention **If** CriticalPuseRate
resolve lack of controllability
NurseIntervention **If** Alarm
expectation
Alarm **Iff** CriticalPulseRate
requirement

Setting the Alarm is something the software **can** control

RP5: Monitoring Example

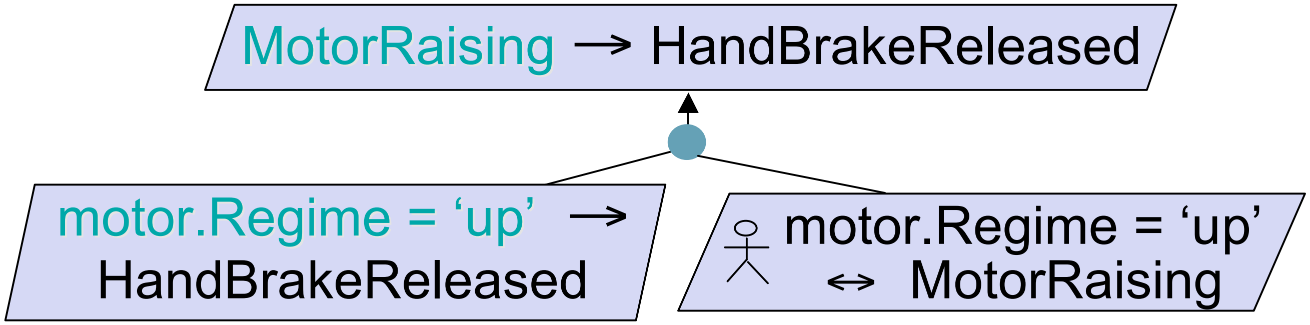


GoalOnUnMonitorableCondition

GoalOnMonitorable Condition

MonitorableCondition ↔ UnmonitorableCondition

The issue is "how can the software know if the engine revs are increasing?



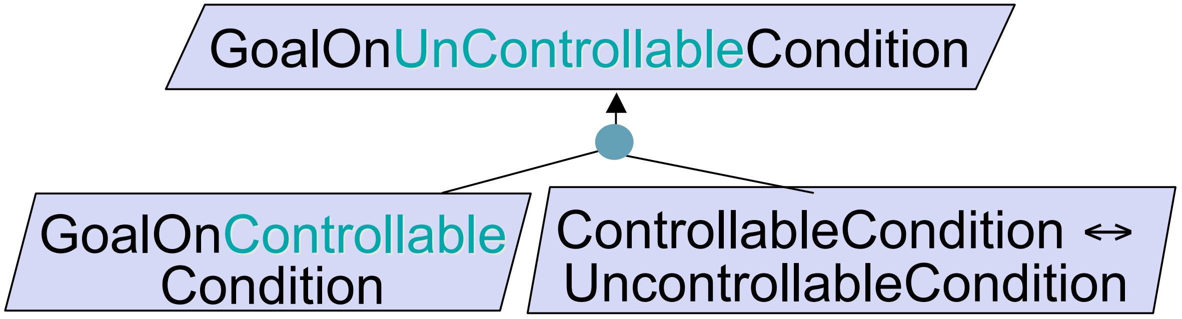
MotorRaising → HandBrakeReleased

motor.Regime = 'up' → HandBrakeReleased

motor.Regime = 'up' ↔ MotorRaising

Answer: introduce a variable that the software can monitor, and make an agent responsible for setting that value under the appropriate circumstances

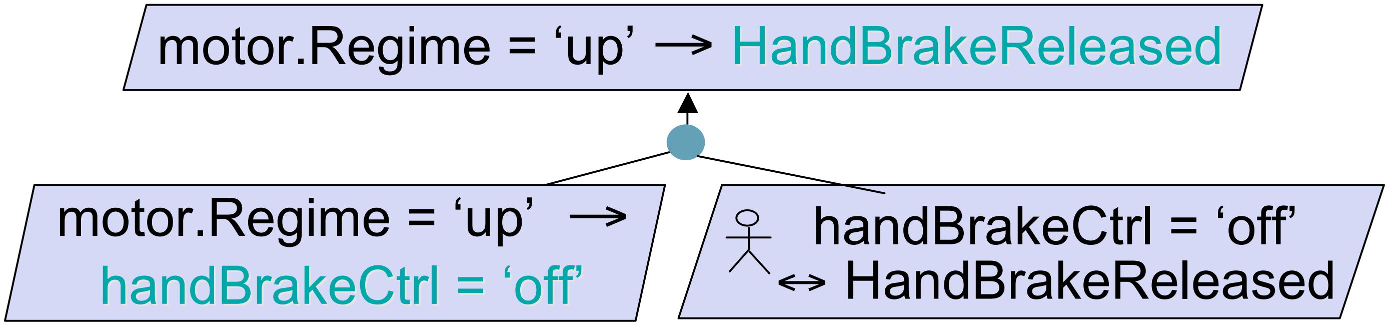
And apply it again…



GoalOnUnControllableCondition

GoalOnControllable Condition

ControllableCondition ↔ UncontrollableCondition

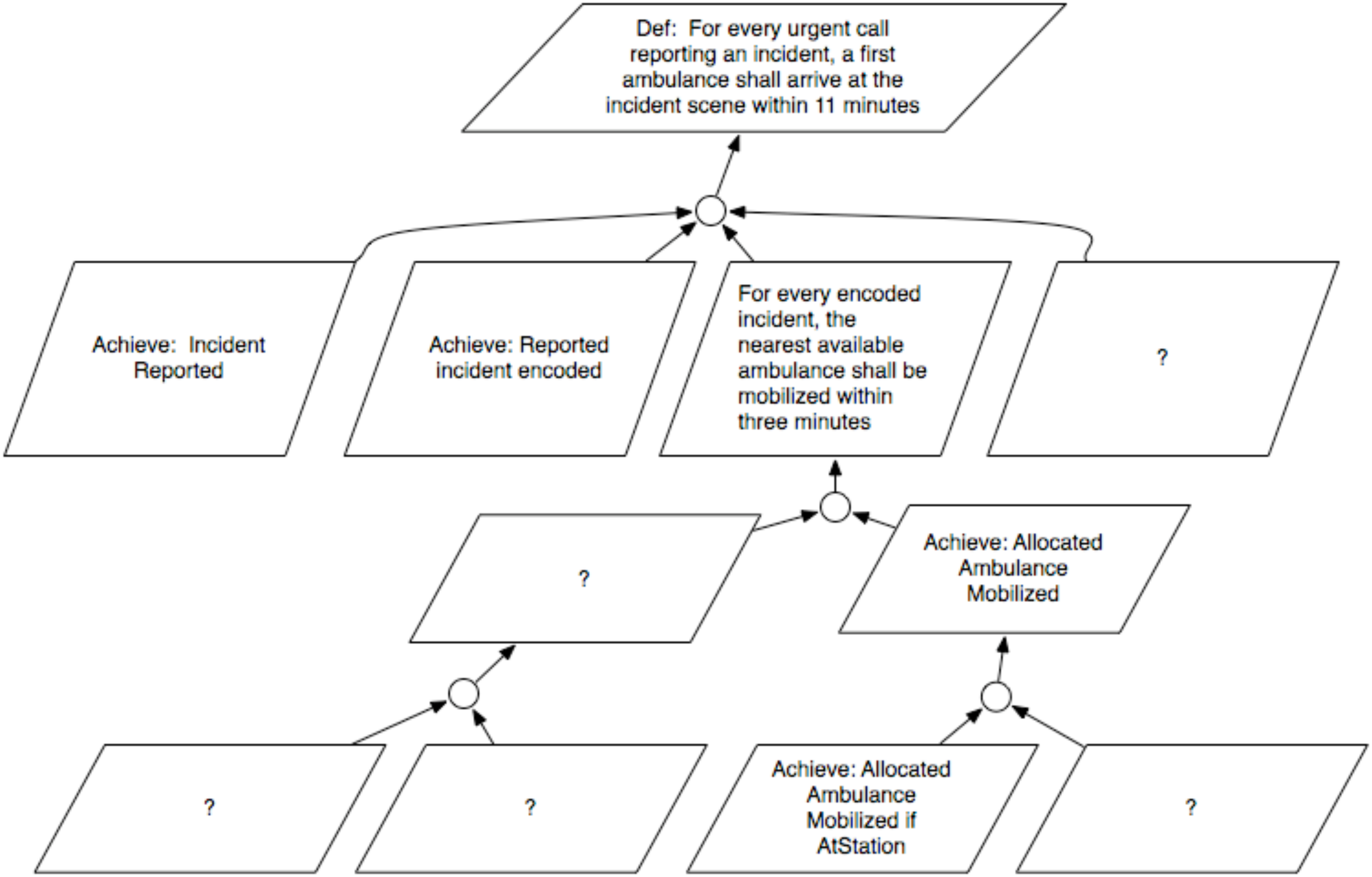


motor.Regime = 'up' → HandBrakeReleased

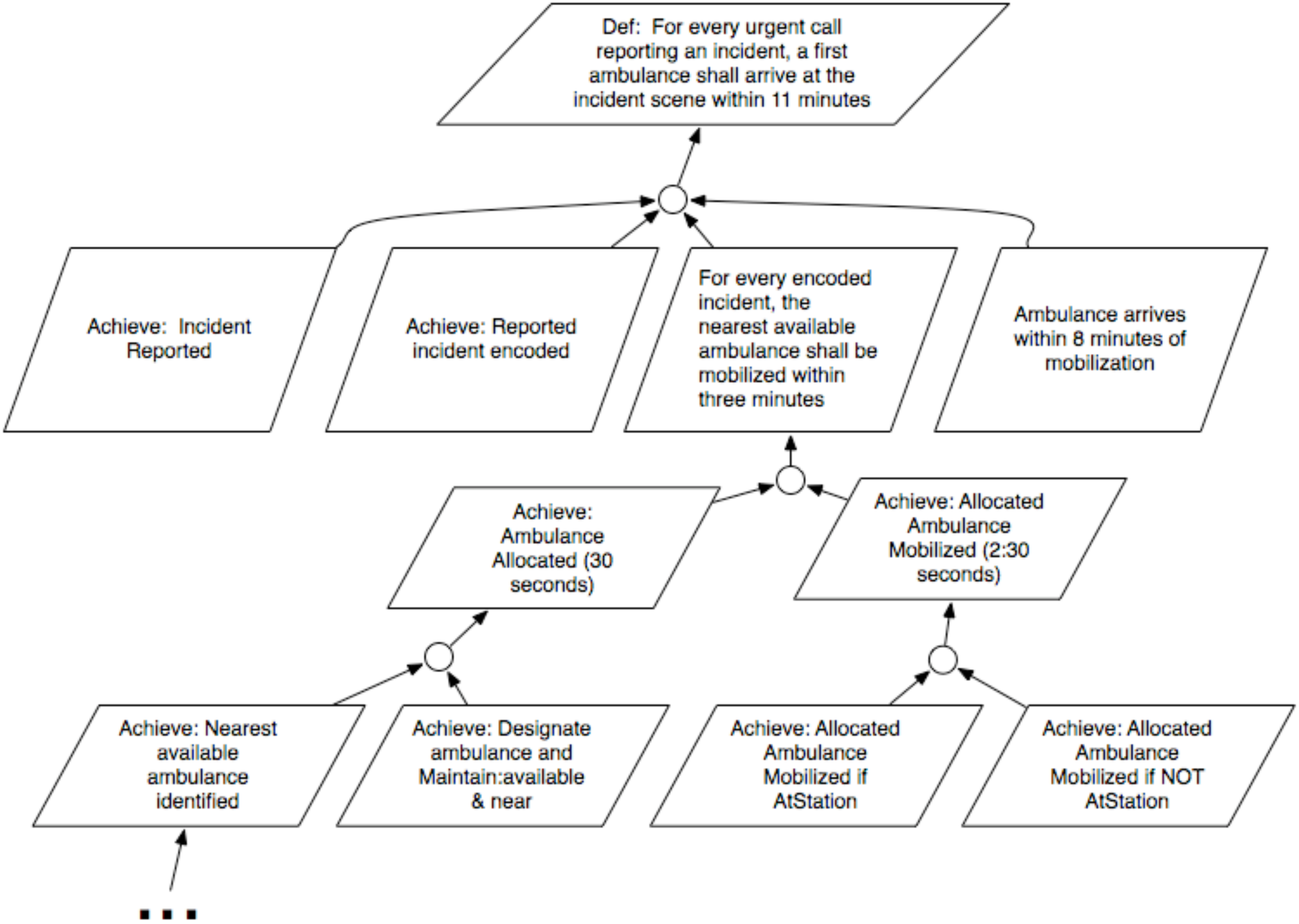
motor.Regime = 'up' → handBrakeCtrl = 'off'

handBrakeCtrl = 'off' ↔ HandBrakeReleased

req



Def: For every urgent call reporting an incident, a first ambulance shall arrive at the incident scene within 11 minutes
Achieve: Incident Reported
Achieve: Reported incident encoded
For every encoded incident, the nearest available ambulance shall be mobilized within three minutes
?
?
Achieve: Allocated Ambulance Mobilized
?
?
Achieve: Allocated Ambulance Mobilized if AtStation
?


Def: For every urgent call reporting an incident, a first ambulance shall arrive at the incident scene within 11 minutes
Achieve: Incident Reported
Achieve: Reported incident encoded
For every encoded incident, the nearest available ambulance shall be mobilized within three minutes
Ambulance arrives within 8 minutes of mobilization
Achieve: Ambulance Allocated (30 seconds)
Achieve: Allocated Ambulance Mobilized (2:30 seconds)
Achieve: Nearest available ambulance identified
Achieve: Designate ambulance and Maintain:available & near
Achieve: Allocated Ambulance Mobilized if AtStation
Achieve: Allocated Ambulance Mobilized if NOT AtStation
...