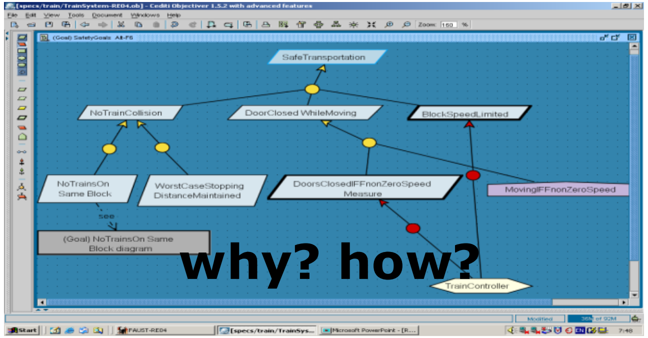
Risk Analysis on Goal Models

Mariano Ceccato

mariano.ceccato@univr.it

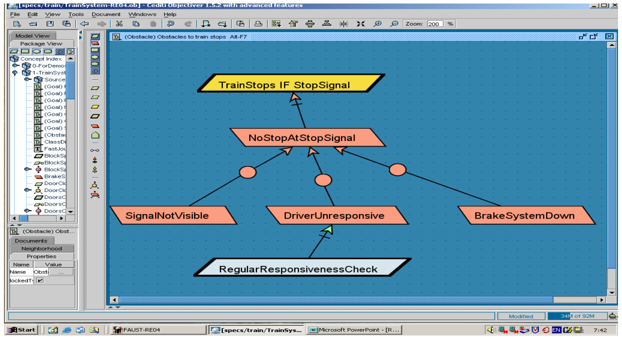
Building models for RE

Goals

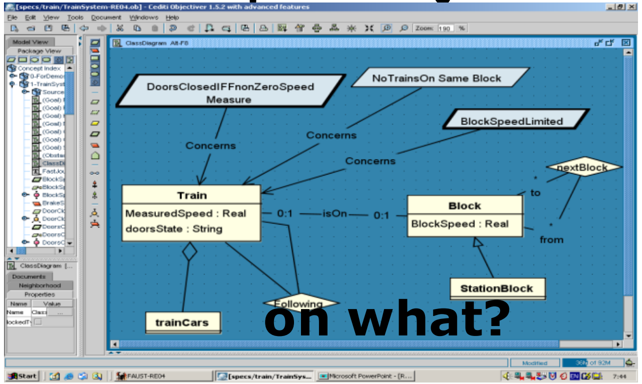

[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Zoom: 150 %
(Goal) SafetyGoals Alt-F6
SafeTransportation
NoTrainCollision
DoorClosed WhileMoving
BlockSpeedLimited
NoTrainsOn Same Block
WorstCaseStopping DistanceMaintained
DoorsClosedIFFnonZeroSpeed Measure
MovingIFFnonZeroSpeed
see
(Goal) NoTrainsOn Same Block diagram
TrainController
Start
FAUST-RE04
[specs/train/TrainSys...
Microsoft PowerPoint - [R...

why? how?

Risks

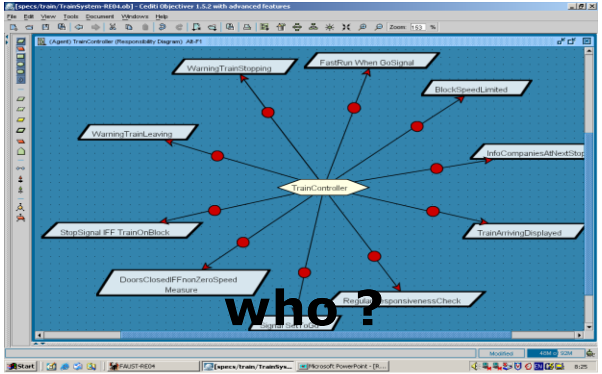

[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Zoom: 200 %
Model View
Package View
Concept Index
(Obstacle) Obstacles to train stops Alt-F7
TrainStops IF StopSignal
NoStopAtStopSignal
SignalNotVisible
DriverUnresponsive
BrakeSystemDown
RegularResponsivenessCheck
Documents
Neighborhood
Properties
Name
Value
Start
FAUST-RE04
[specs/train/TrainSys...
Microsoft PowerPoint - [R...

Conceptual objects



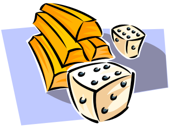
on what?

Agents

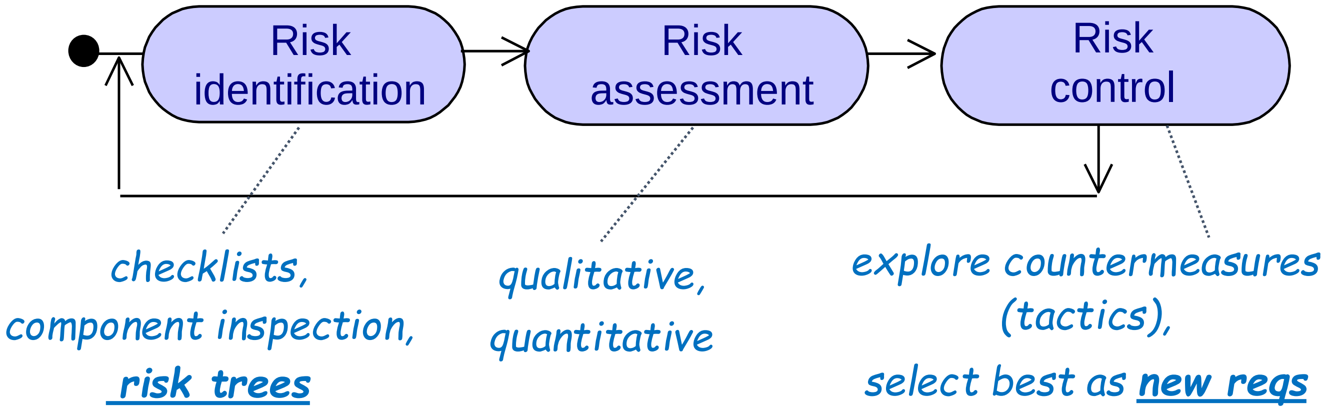

[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Zoom: 153 %
(Agent) TrainController (Responsibility Diagram) Alt-F1
WarningTrainStopping
FastRun When GoSignal
BlockSpeedLimited
WarningTrainLeaving
InfoCompaniesAtNextStop
TrainController
StopSignal IFF TrainOnBlock
TrainArrivingDisplayed
DoorsClosedIFFnonZeroSpeed Measure
RegularResponsivenessCheck
SignalSetToGo
Start
FAUST-RE04
[specs/train/TrainSys...
Microsoft PowerPoint - [R...

who ?

Risk analysis as seen in “Requirements Evaluation” lecture



- **Risk** = uncertain factor whose occurrence may result in loss of satisfaction of corresponding objective
  - has likelihood & consequences (each having likelihood, severity)
- Poor risk management is a major cause of software failure
- Early risk analysis at RE time:



Risk identification

Risk assessment

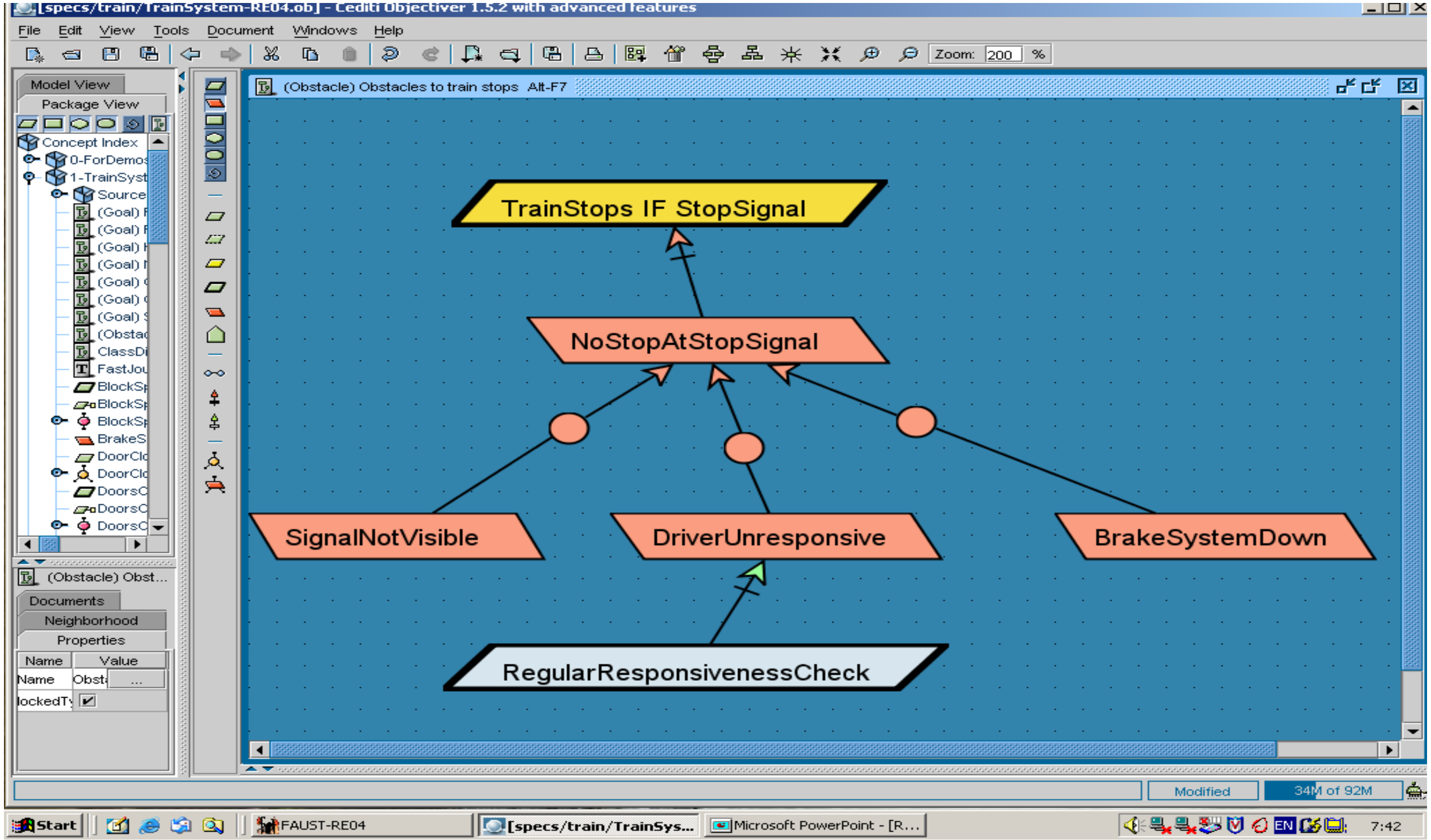
Risk control

*checklists, component inspection, **risk trees***

qualitative, quantitative

*explore countermeasures (tactics), select best as **new reqs***

Risk analysis can be anchored on goal models



Risk analysis on goal models: outline

- Goal obstruction by obstacles
  - What are obstacles?
  - Completeness of a set of obstacles
  - Obstacle categories
- Modeling obstacles
  - Obstacle diagrams
  - Obstacle refinement
  - Bottom-up propagation of obstructions in goal AND-refinements
  - Annotating obstacle diagrams
- Obstacle analysis for a more robust goal model
  - Identifying obstacles
  - Evaluating obstacles
  - Resolving obstacles in a modified goal model

What are obstacles ?

- Motivation: goals in refinement graph are often too ideal, likely to be violated under abnormal conditions
  - (unintentional or intentional agent behaviors)

- **Obstacle** = condition on system for violation of corresponding assertion (generally a goal)
  - Obstruction: {O, Dom } **|=** **not** G
  - Domain consistency: {O, Dom } **|≠** **false**
  - Feasibility: O can be satisfied by some system behavior

  - e.g. G: TrainStoppedAtBlockSignal **If** StopSignal
    Dom: **If** TrainStopsAtStopSignal **then** DriverResponsive
    O: Driver**Un**responsive

- For behavioral goal: existential property capturing unadmissible behavior (negative scenario)

Completeness of a set of obstacles

- Ideally, a set of obstacles to G should be complete
  - Domain completeness: {**not** O_1,..., **not** O_n, Dom } **|=** G
  - e.g.

If not DriverUnresponsive **and not** BrakeSystemDown **and** StopSignal
then TrainStoppedAtBlockSignal ???

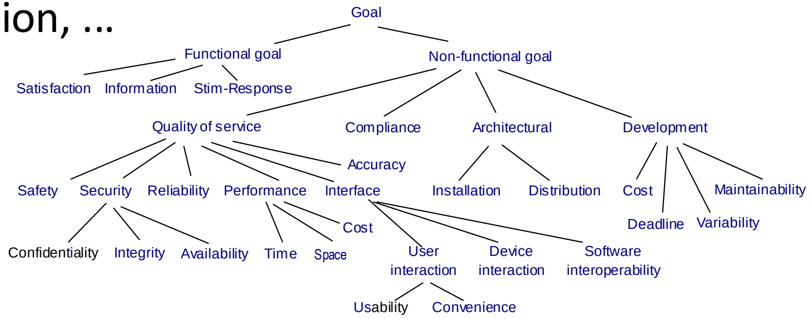
- Completeness is highly desirable for mission-critical goals ...
- ... but bounded by what we know about the domain!
- Obstacle analysis may help elicit relevant domain properties

Obstacle categories for heuristic identification

Correspond to goal categories & their refinement:

- **Hazard** obstacles obstruct <u>Safety</u> goals
- **Threat** obstacles obstruct <u>Security</u> goals
  - Disclosure, Corruption, DenialOfService, ...

- **Inaccuracy** obstacles obstruct <u>Accuracy</u> goals
- **Misinformation** obstacles obstruct <u>Information</u> goals
  - NonInformation, WrongInformation, TooLateInformation, ...
- **Dissatisfaction** obstacles obstruct <u>Satisfaction</u> goals
  - NonSatisfaction, PartialSatisfaction, TooLateSatisfaction, ...
- **Unusability** obstacles obstruct <u>Usability</u> goals
- ...



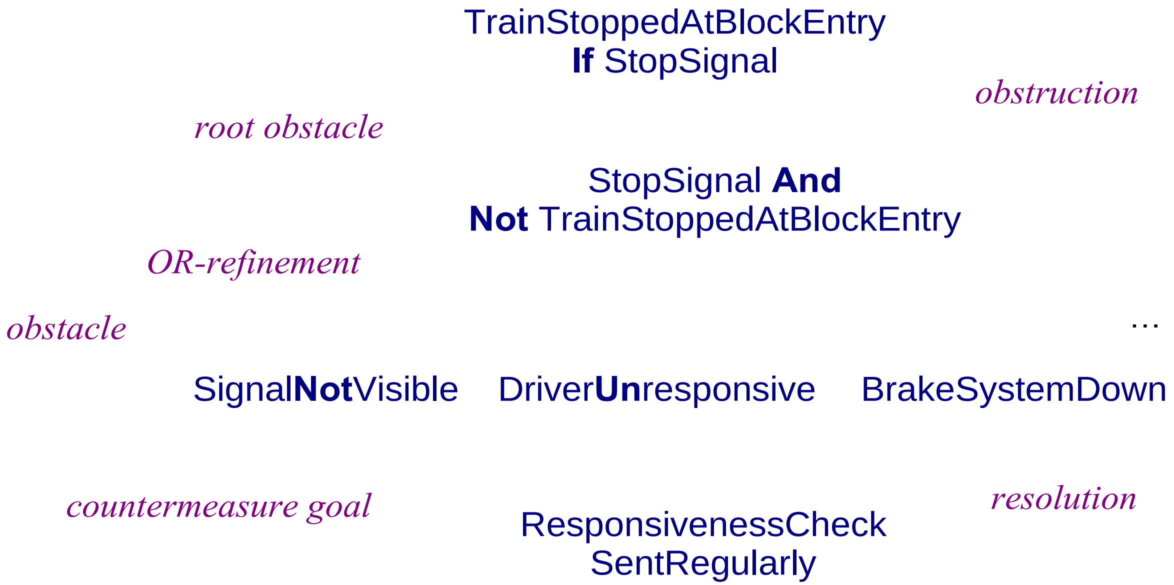
- Goal
  - Functional goal
    - Satisfaction
    - Information
    - Stim-Response
  - Non-functional goal
    - Quality of service
      - Safety
      - Security
        - Confidentiality
        - Integrity
        - Availability
      - Reliability
      - Performance
        - Time
        - Space
        - Cost
      - Interface
        - User interaction
          - Usability
          - Convenience
        - Device interaction
        - Software interoperability
      - Accuracy
    - Compliance
    - Architectural
      - Installation
      - Distribution
    - Development
      - Cost
      - Deadline
      - Variability
      - Maintainability

Risk analysis on goal models: outline

- Goal obstruction by obstacles
  - What are obstacles?
  - Completeness of a set of obstacles
  - Obstacle categories
- Modeling obstacles
  - Obstacle diagrams
  - Obstacle refinement
  - Bottom-up propagation of obstructions in goal AND-refinements
  - Annotating obstacle diagrams
- Obstacle analysis for a more robust goal model
  - Identifying obstacles
  - Evaluating obstacles
  - Resolving obstacles in a modified goal model

Obstacle diagrams as AND/OR refinement trees

- Anchored on leafgoals in goal model (unlike risk trees)
  - root = **not** *G*
  - obstacle **AND**-refinement, **OR**-refinement: same semantics as goals
  - **leaf** obstacles: feasibility, likelihood, resolution easier to determine



TrainStoppedAtBlockEntry
If StopSignal

obstruction

root obstacle

StopSignal **And**
Not TrainStoppedAtBlockEntry

OR-refinement

obstacle

…

Signal**Not**Visible

Driver**Un**responsive

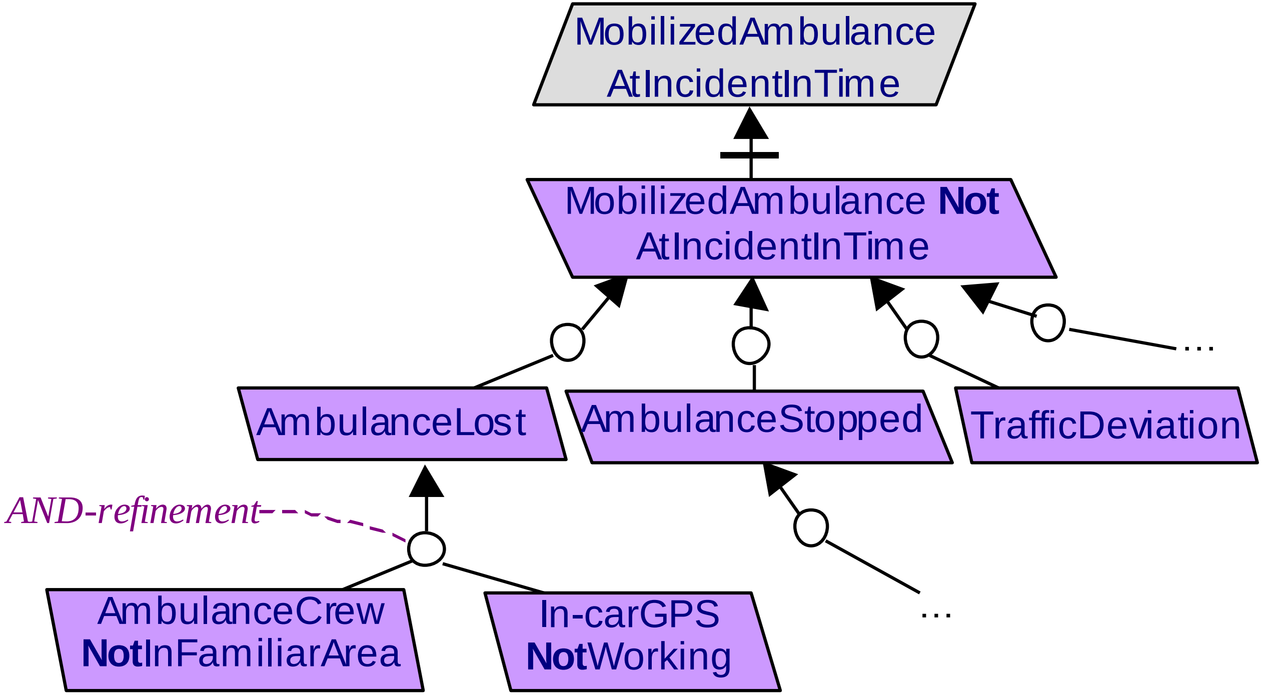
BrakeSystemDown

countermeasure goal

resolution

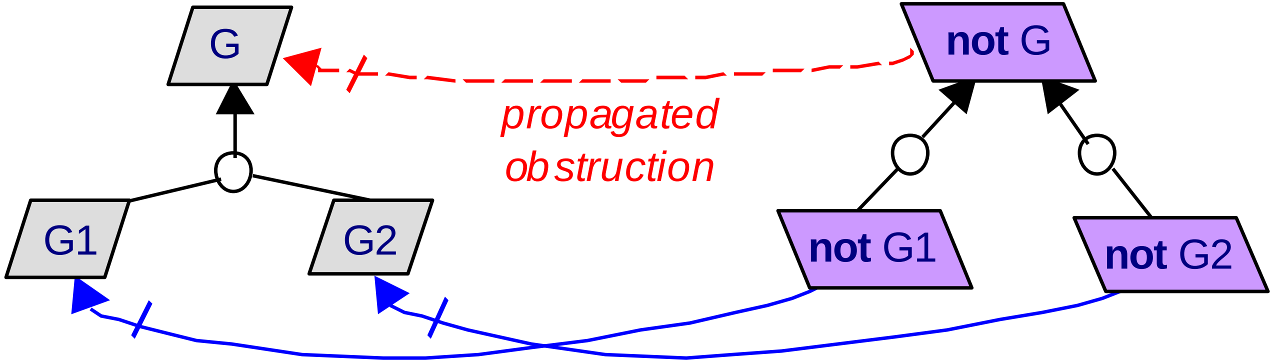
ResponsivenessCheck
SentRegularly

Obstacle diagrams as AND/OR refinement trees


MobilizedAmbulance AtIncidentInTime
MobilizedAmbulance **Not** AtIncidentInTime
AmbulanceLost
AmbulanceStopped
TrafficDeviation
...
AND-refinement
AmbulanceCrew **Not**InFamiliarArea
In-carGPS **Not**Working
...

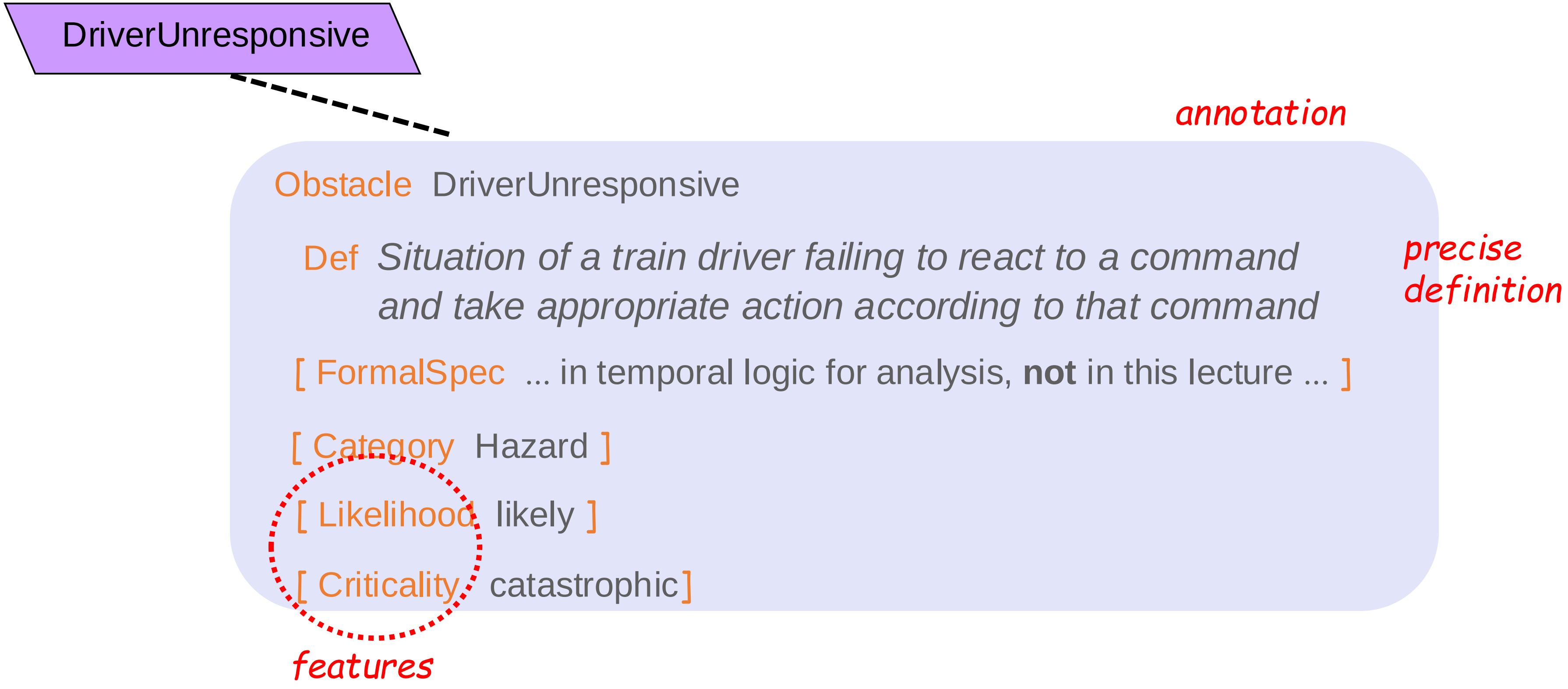
Obstructions propagate bottom-up in goal AND-refinement trees

- De Morgan’s law: not (G1 and G2) equivalent to (not G1) or (not G2)



=> *Severity of **<u>consequences</u>** of an obstacle can be assessed in terms of higher-level goals obstructed*

Annotating obstacle diagrams



annotation

Obstacle DriverUnresponsive

Def *Situation of a train driver failing to react to a command and take appropriate action according to that command*

precise definition

[FormalSpec ... in temporal logic for analysis, **not** in this lecture ...]

[Category Hazard]

[Likelihood likely]

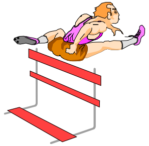
[Criticality catastrophic]

features

Risk analysis on goal models: outline

- Goal obstruction by obstacles
    - What are obstacles?
    - Completeness of a set of obstacles
    - Obstacle categories
- Modeling obstacles
    - Obstacle diagrams
    - Obstacle refinement
    - Bottom-up propagation of obstructions in goal AND-refinements
    - Annotating obstacle diagrams
- Obstacle analysis for a more robust goal model
    - Identifying obstacles
    - Evaluating obstacles
    - Resolving obstacles in a modified goal model

Obstacle analysis for increased system robustness



Anticipate obstacles:

⇒ more realistic goals, new goals as countermeasures to abnormal conditions

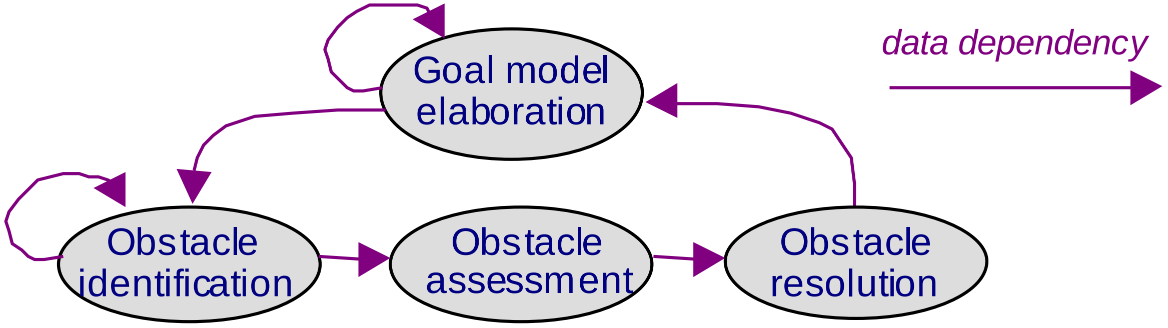
⇒ more complete, realistic goal model

Obstacle analysis:

- For selected goals in the goal model:
  - **identify** as many obstacles to it as possible;
  - **assess** their likelihood & severity;
  - **resolve** them according to likelihood & severity

    => new goals as countermeasures in the goal model

Obstacle analysis and goal model elaboration are intertwined



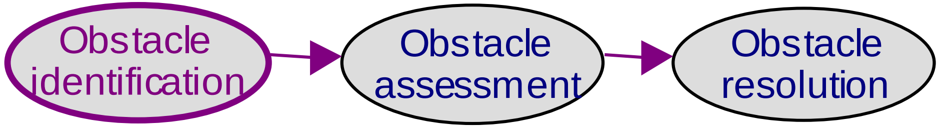
- Goal-obstacle analysis loop terminates when remaining obstacles can be tolerated
  - unlikely or acceptable consequences

- Which goals to consider in the goal model?
  - **leafgoals** (requirements or expectations): easier to refine what is wanted than what is not wanted (+ up-propagation in goal model)
  - based on annotated Priority & Category (Hazard, Security, ...)

Identifying obstacles

- For obstacle to selected assertion *G*
  - *G* can be goal, hypothesis, suspect dom prop ...
- negate *G*; {=> root obstacle}
- find AND/OR refinements of *not G* in view of valid domain properties
  - according to desired extensiveness
- until reaching obstruction preconditions whose feasibility, likelihood, severity, resolvability is easy to assess

= goal-anchored construction of risk-tree



Identifying obstacles: tautology-based refinement

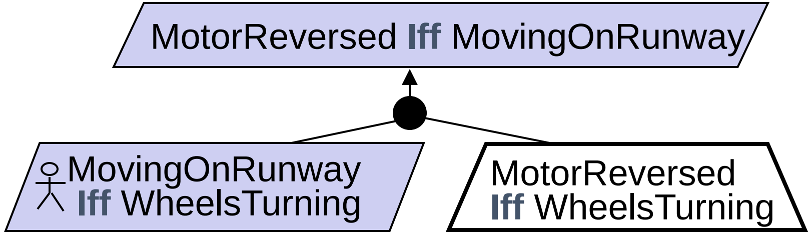
- Goal negation as root => use tautologies to drive refinements

e.g.

- `not(A and B)` amounts to `not A or not B`
- `not(A or B)` amounts to `not A and not B`
- `not(if A then B)` amounts to `A and not B`
- `not(A iff B)` amounts to `(A and not B) or (not A and B)`

=> complete OR-refinements when or-connective gets in

Identifying obstacles by tautology-based refinement

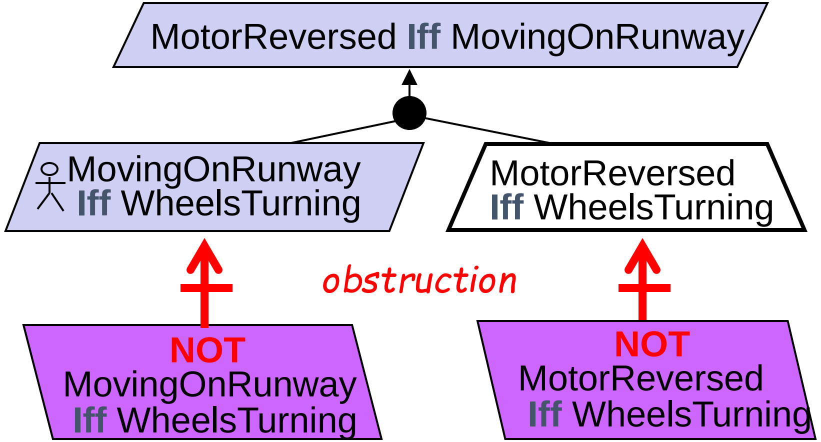


MotorReversed **Iff** MovingOnRunway

MovingOnRunway **Iff** WheelsTurning

MotorReversed **Iff** WheelsTurning

Identifying obstacles by tautology-based refinement



MotorReversed Iff MovingOnRunway

MovingOnRunway Iff WheelsTurning

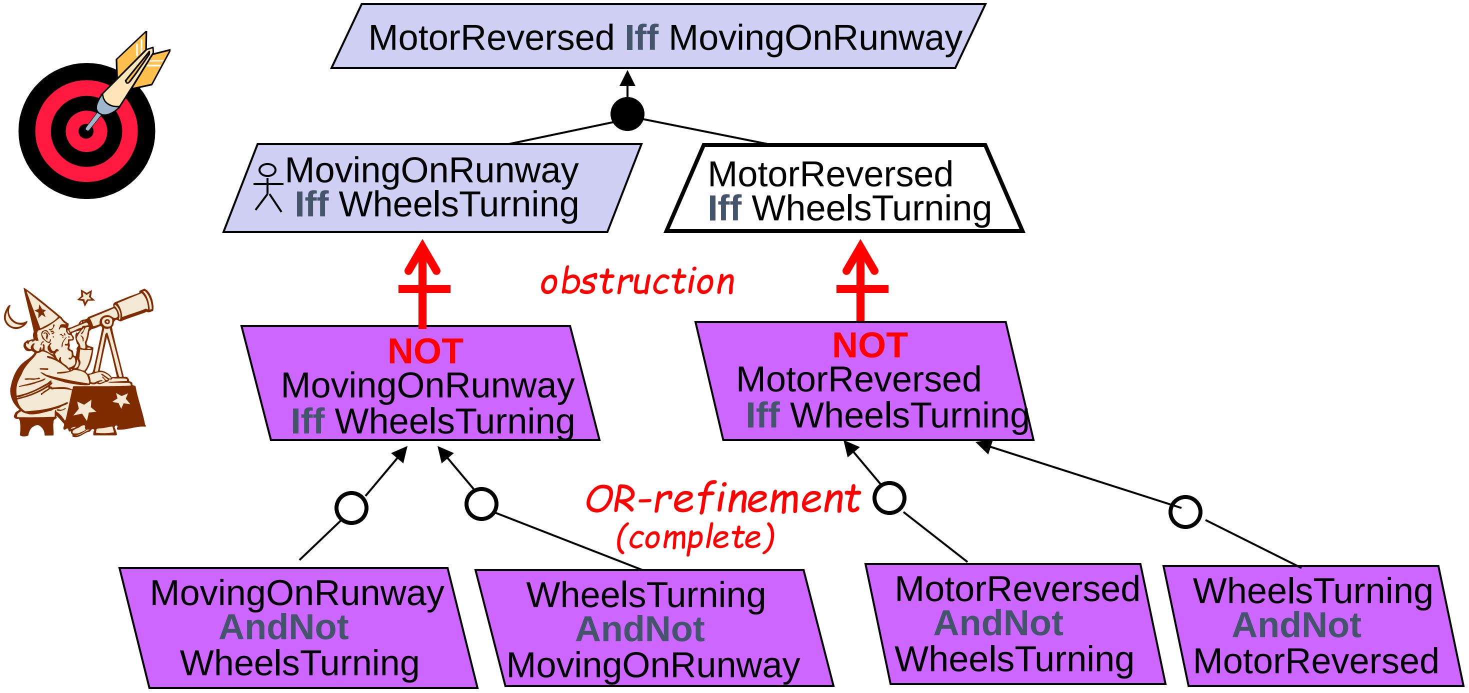
MotorReversed Iff WheelsTurning

obstruction

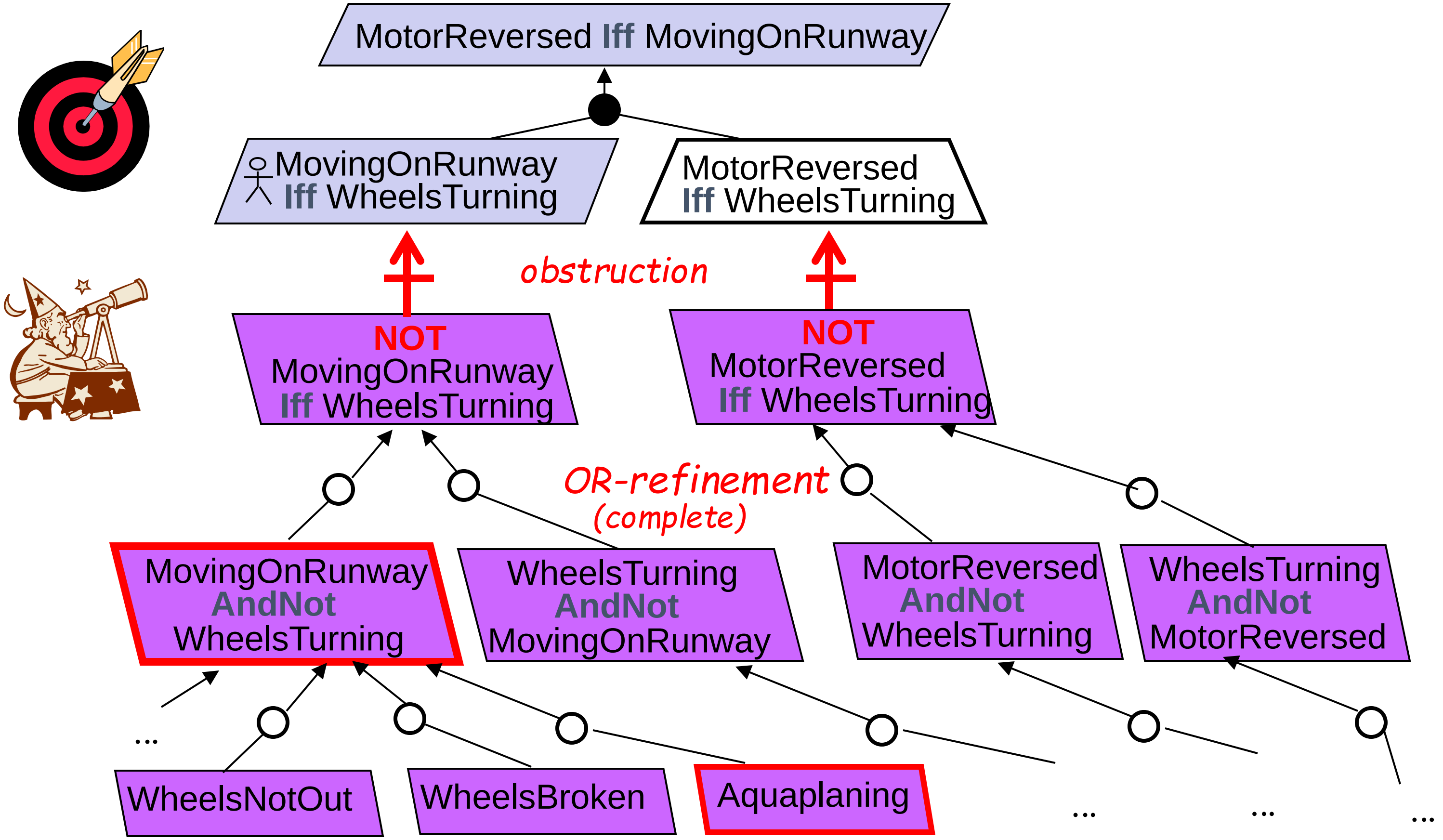
NOT MovingOnRunway Iff WheelsTurning

NOT MotorReversed Iff WheelsTurning

Identifying obstacles by tautology-based refinement


MotorReversed Iff MovingOnRunway
MovingOnRunway Iff WheelsTurning
MotorReversed Iff WheelsTurning
obstruction
NOT MovingOnRunway Iff WheelsTurning
NOT MotorReversed Iff WheelsTurning
OR-refinement (complete)
MovingOnRunway AndNot WheelsTurning
WheelsTurning AndNot MovingOnRunway
MotorReversed AndNot WheelsTurning
WheelsTurning AndNot MotorReversed

Identifying obstacles by tautology-based refinement



MotorReversed Iff MovingOnRunway

MovingOnRunway Iff WheelsTurning

MotorReversed Iff WheelsTurning

obstruction

NOT MovingOnRunway Iff WheelsTurning

NOT MotorReversed Iff WheelsTurning

OR-refinement (complete)

MovingOnRunway AndNot WheelsTurning

WheelsTurning AndNot MovingOnRunway

MotorReversed AndNot WheelsTurning

WheelsTurning AndNot MotorReversed

...

WheelsNotOut

WheelsBroken

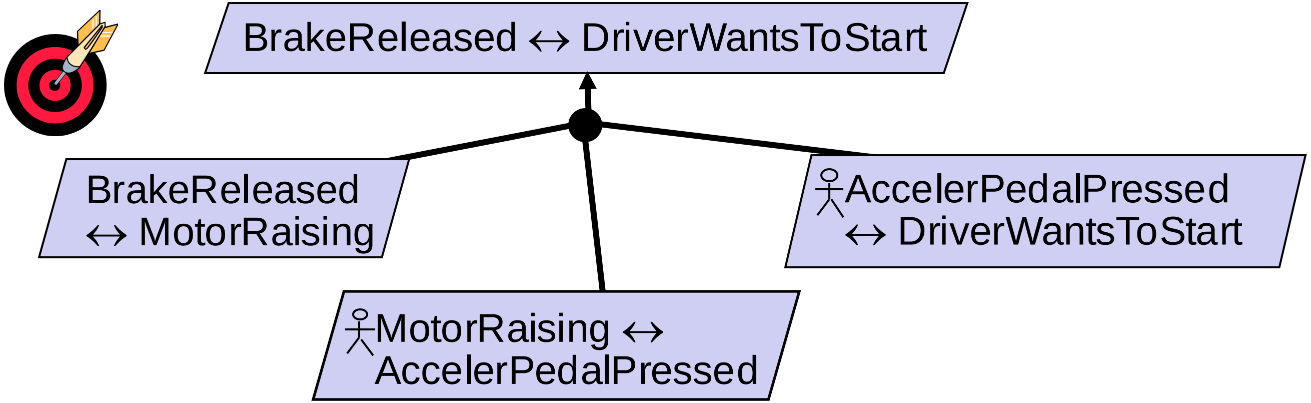
Aquaplaning

...

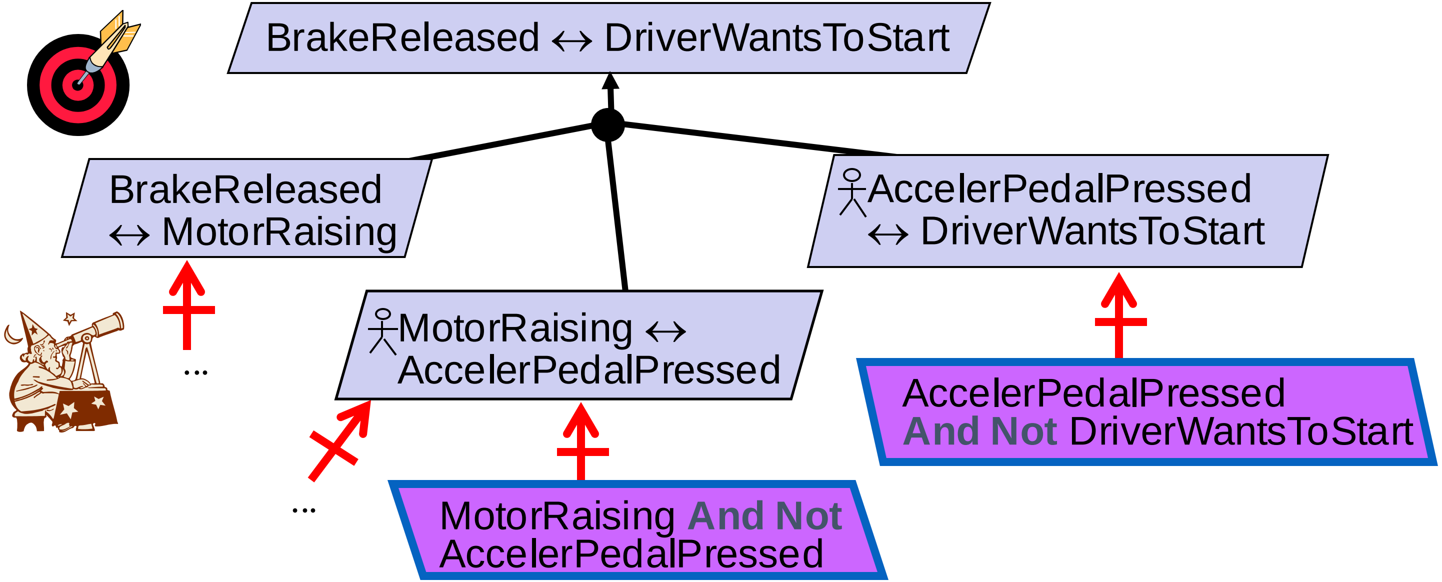
...

...

Obstacle identification: another example


BrakeReleased ↔ DriverWantsToStart
BrakeReleased ↔ MotorRaising
AccelerPedalPressed ↔ DriverWantsToStart
MotorRaising ↔ AccelerPedalPressed

Obstacle identification: another example



BrakeReleased ↔ DriverWantsToStart

BrakeReleased ↔ MotorRaising

MotorRaising ↔ AccelerPedalPressed

AccelerPedalPressed ↔ DriverWantsToStart

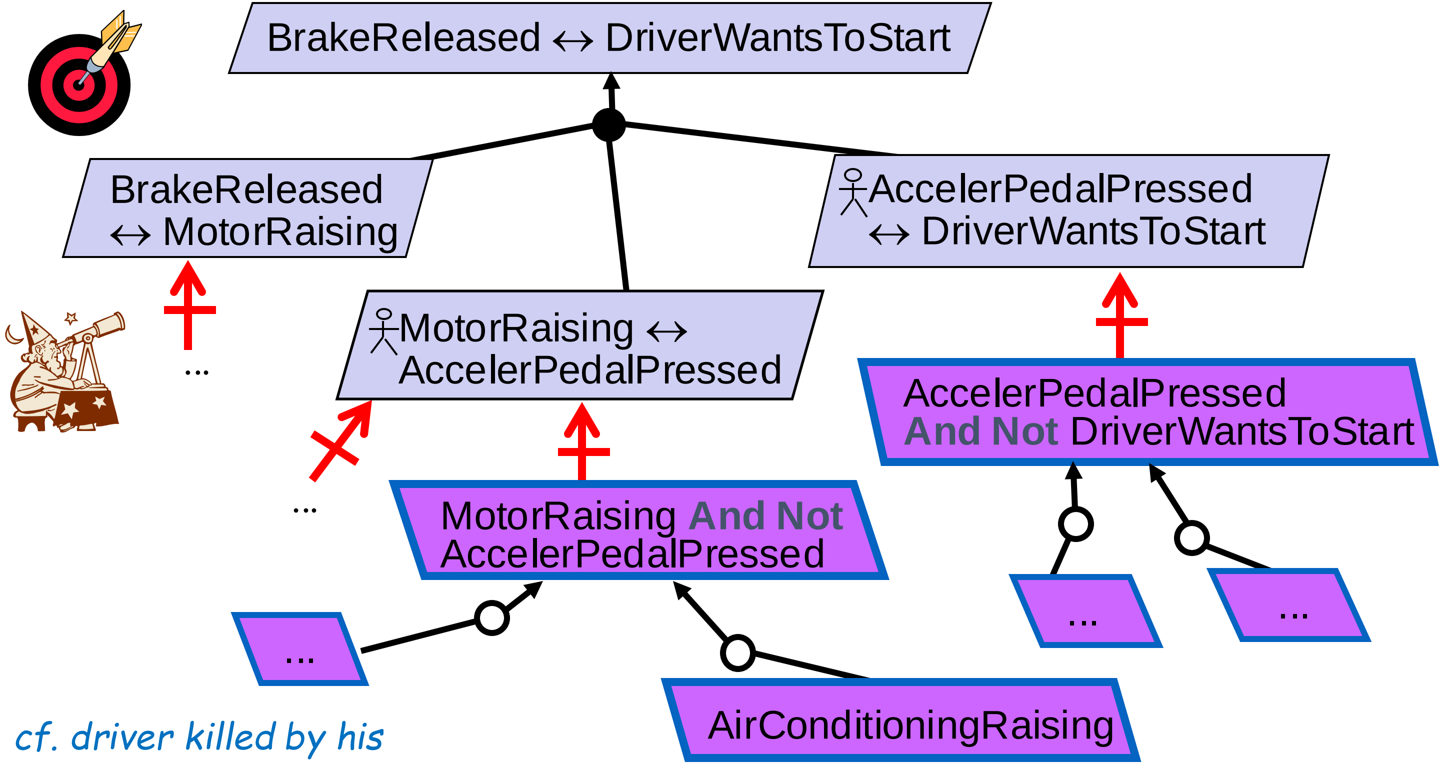
…

…

MotorRaising **And Not** AccelerPedalPressed

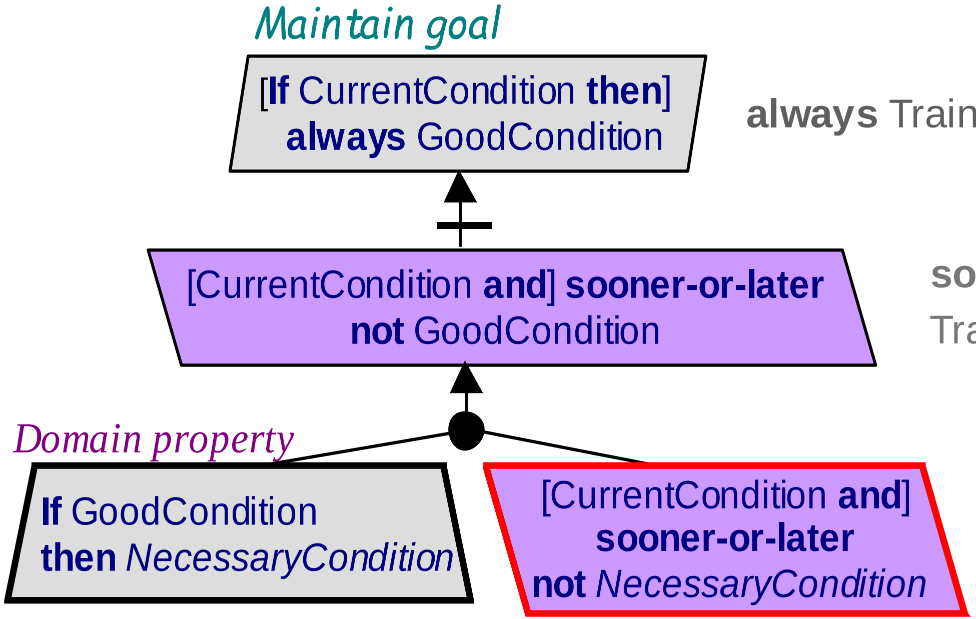
AccelerPedalPressed **And Not** DriverWantsToStart

Obstacle identification: another example


BrakeReleased ↔ DriverWantsToStart
BrakeReleased ↔ MotorRaising
…
MotorRaising ↔ AccelerPedalPressed
AccelerPedalPressed ↔ DriverWantsToStart
…
MotorRaising And Not AccelerPedalPressed
AccelerPedalPressed And Not DriverWantsToStart
…
AirConditioningRaising
…
…

cf. driver killed by his luxurious car on a hot summerday

Identifying obstacles from necessary conditions for obstructed target



Maintain goal
[If CurrentCondition then] always GoodCondition
[CurrentCondition and] sooner-or-later not GoodCondition
Domain property
If GoodCondition then *NecessaryCondition*
[CurrentCondition and] sooner-or-later not *NecessaryCondition*

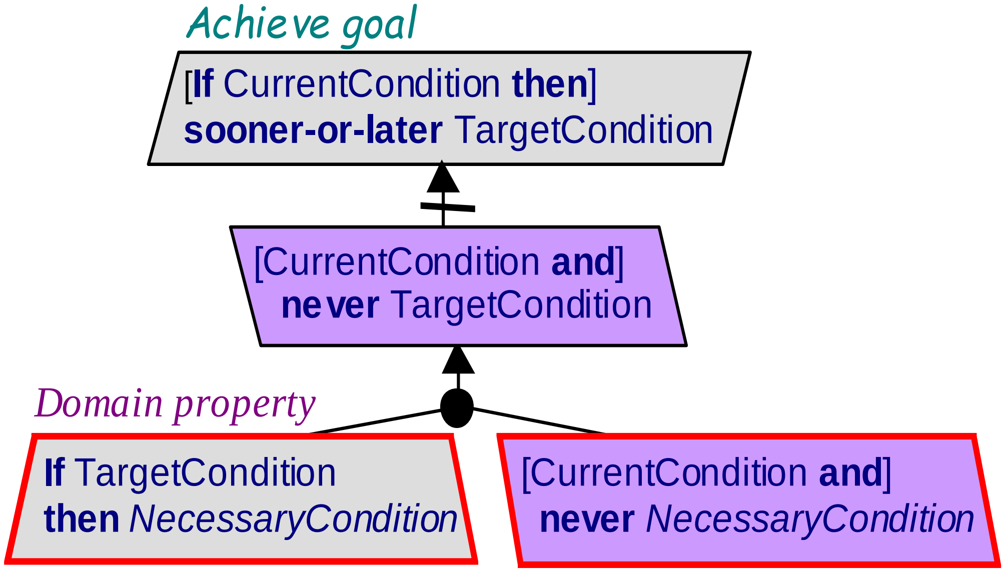
always TrainStopped**If**StopSignal

sooner-or-later not TrainStopped**If**StopSignal

If TrainStopped**If**StopSignal **then** *DriverResponsive*

sooner-or-later not *DriverResponsive*

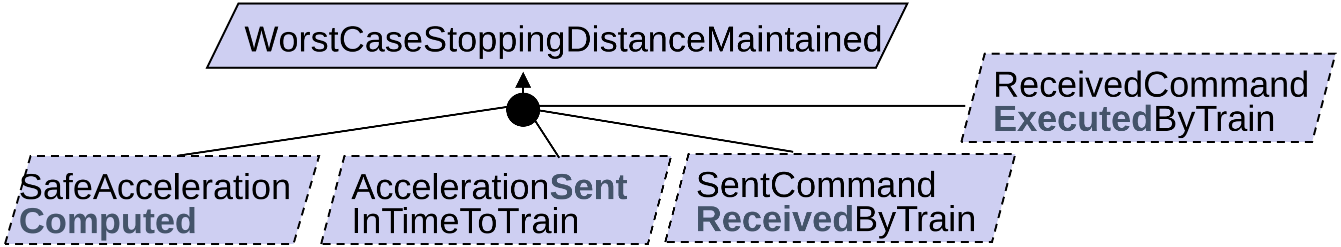
Identifying obstacles from necessary conditions for obstructed target


Achieve goal
[**If** CurrentCondition **then**]
sooner-or-later TargetCondition
[CurrentCondition **and**]
never TargetCondition
Domain property
If TargetCondition
then *NecessaryCondition*
[CurrentCondition **and**]
never *NecessaryCondition*

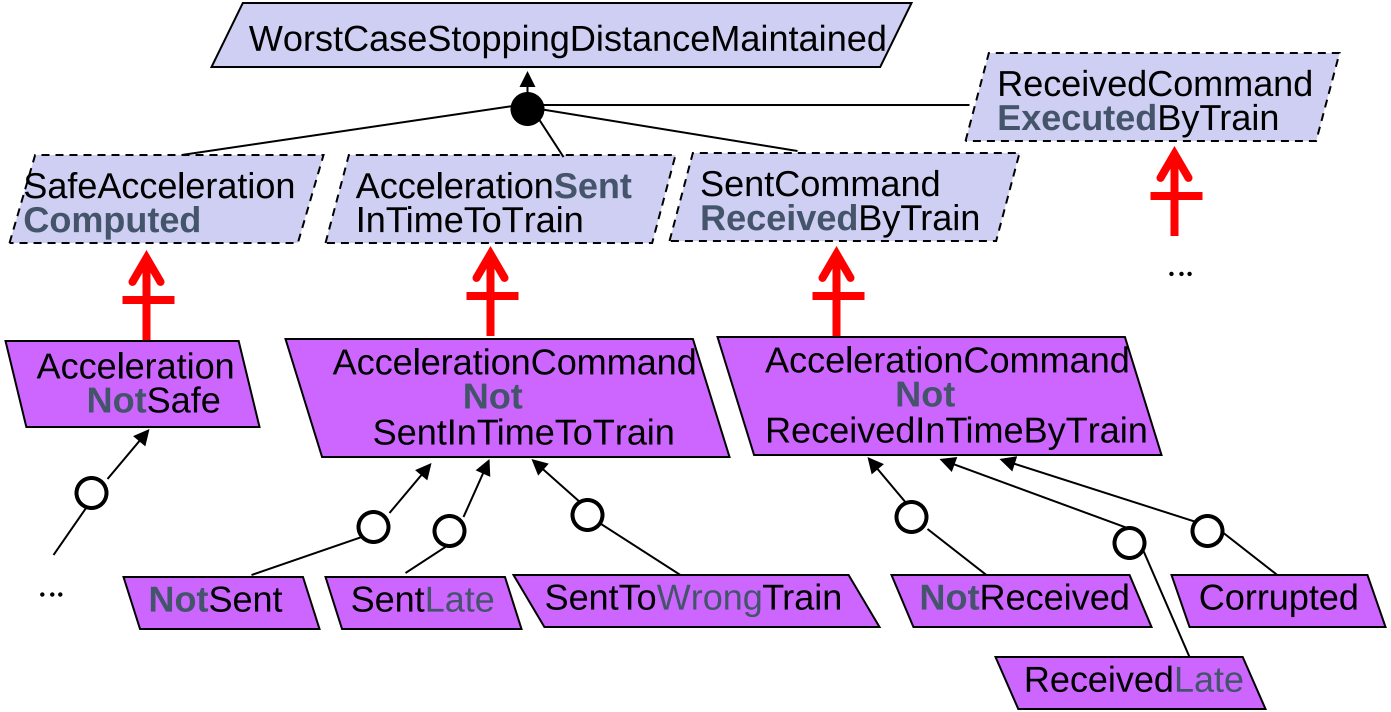
Can also be used for eliciting relevant domain properties

- "what are necessary conditions for TargetCondition?"

Obstacle models as goal-anchored fault trees

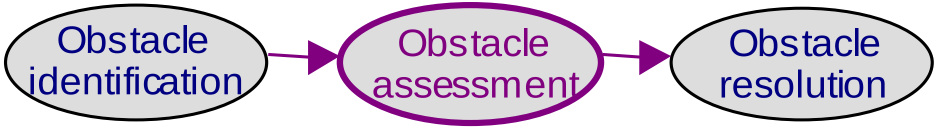

WorstCaseStoppingDistanceMaintained
ReceivedCommand **Executed**ByTrain
SafeAcceleration **Computed**
Acceleration**Sent** InTimeToTrain
SentCommand **Received**ByTrain

Obstacle models as goal-anchored fault trees

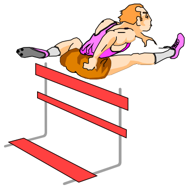

WorstCaseStoppingDistanceMaintained
ReceivedCommand ExecutedByTrain
SafeAcceleration Computed
AccelerationSent InTimeToTrain
SentCommand ReceivedByTrain
...
Acceleration NotSafe
AccelerationCommand Not SentInTimeToTrain
AccelerationCommand Not ReceivedInTimeByTrain
...
NotSent
SentLate
SentToWrongTrain
NotReceived
Corrupted
ReceivedLate

Evaluating obstacles

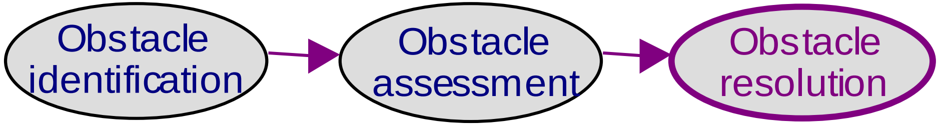
- Check *domain-consistency* & *feasibility* conditions
  - satisfying scenario ?
- Assess *Likelihood* and *Criticality*
  - with domain experts
  - rough estimates can be obtained from propagation rules:
    - Likelihood (O) = **min$_i$** (Likelihood (sO$_i$)) if *O* is **AND**-refined to *sO$_i$*
    - Likelihood (*O*) = **max$_i$** (Likelihood (*sO$_i$*)) if *O* is **OR**-refined to *sO$_i$*
  - severity of consequences can be estimated from *number* & *Priority* of higher-level goals obstructed by up-propagation in goal trees



Resolving obstacles



- Resolution through countermeasures
  - new or modified goals in goal model
  - often to be refined
- For every identified obstacle ...
  - explore alternative resolutions
  - select "best" resolution based on ...
    - likelihood/severity of obstacle
    - non-functional/quality goals in goal model

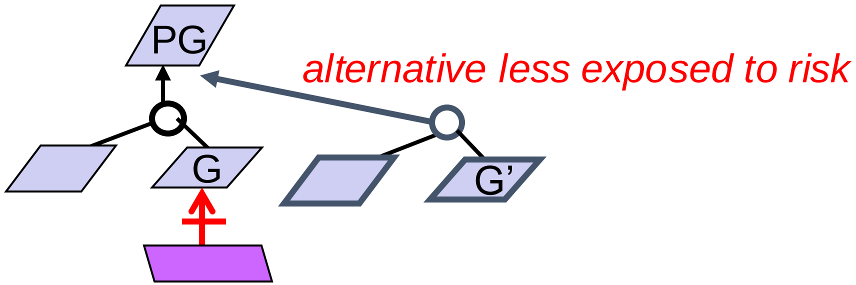


Exploring alternative countermeasures

By use of **model transformation operators**

- encode resolution tactics

- **Goal substitution**: consider alternative refinement of parent goal to avoid obstruction of child goal



e.g. ~~MotorReversed **Iff** WheelsTurning~~

→ MotorReversed **Iff** PlaneWeightSensed

Exploring alternative countermeasures

- **Agent substitution**: consider alternative responsibilities for obstructed goal so as to make obstacle unfeasible

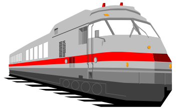
G *G*

O **ag** **ag'**

e.g. Maintain [SafeAccelerationComputed]

obstructed by ComputedAccelerationNotSafe

~~OnBoardTrainController~~ → VitalStationComputer



Exploring alternative countermeasures

- **Goal weakening**: weaken the obstructed goal ’s formulation so that it no longer gets obstructed
  - for **if-then** goal specs: add conjunct in **if**-part
    or disjunct in **then**-part

  e.g. Maintain [TrafficControllerOnDutyOnSector]
  obstructed by NoSectorControllerOnDuty
  → goal weakening:
  TrafficControllerOnDutyOnSector **or** WarningToNextSector

Exploring alternative countermeasures

- **Obstacle prevention**: introduce new goal *Avoid [obstacle]*

  e.g. AccelerationCommandCorrupted

  → Avoid [AccelerationCommandCorrupted]

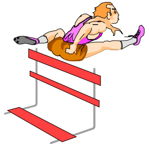
  - to be further refined
    - standard resolution tactics for security threats

- **Goal restoration**: enforce target condition as obstacle occurs
  - => new goal: **if** O **then sooner-or-later** TargetCondition

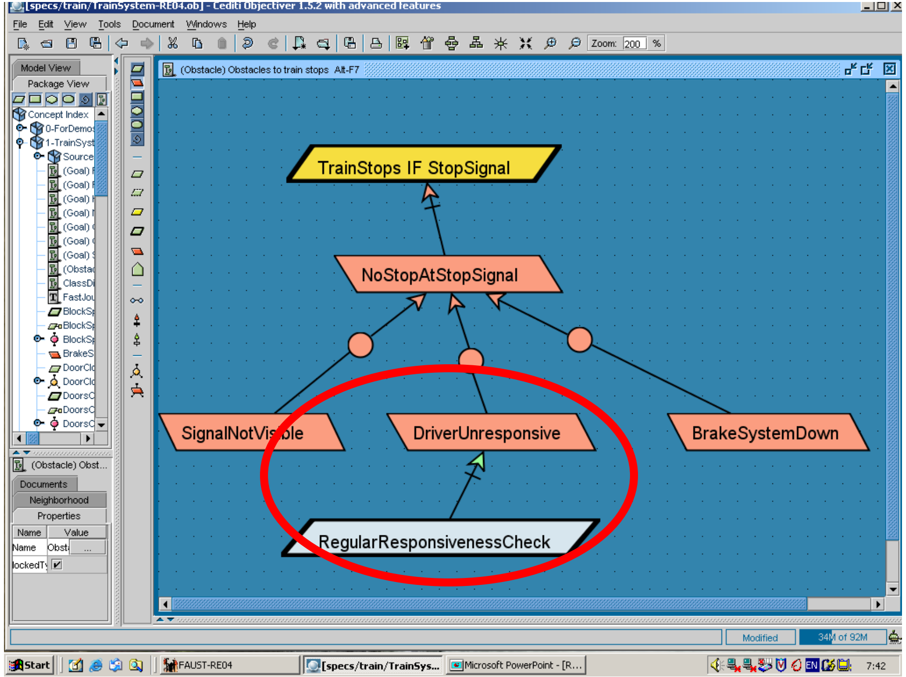
    e.g. ResourceNotReturnedInTime → ReminderSent

    WheelsNotOut → WheelsAlarmGenerated

Exploring alternative countermeasures



- **Obstacle reduction**: reduce obstacle likelihood by ad-hoc countermeasure



Exploring alternative countermeasures

Obstacle mitigation: introduce new goal to mitigate consequences of obstacle:

- **Weak mitigation**: new goal ensures <u>weaker version</u> of goal when obstructed

  e.g. Achieve [AttendanceIfInformedAndMeetingConvenient]
  → Achieve [ImpedimentNotified]

- **Strong mitigation**: new goal <u>ensures parent</u> of goal when obstructed

  e.g. OutdatedSpeed/PositionEstimates
  → Avoid [TrainCollisionWhenOutDatedTrainInfo]

- Resolution goals must then be further refined in the goal model

Selecting best resolution

- Evaluation criteria for comparing alternative resolutions:
  - number of obstacles resolved by the alternative
  - their likelihood & criticality
  - the resolution’s contribution to soft goals
  - its cost

- If obstacle not eliminated, multiple alternatives may be taken

  e.g. FineCharged + ReminderSent (for book copies not returned in time)

- Selected alternative => new/weakened goal in goal model
  - resolution link to obstacle for traceability
  - weakening may need to be propagated in goal model
  - to be refined & checked for conflicts & new obstacles (identify-assess-resolve cycle)