SOFTWARE ENGINEERING

Design Patterns

The KAOS approach (cont.)

Design patterns

Design patterns

- A design pattern is a way of reusing abstract knowledge about a problem and its solution.
- A pattern is a description of the problem and the essence of its solution.
- It should be sufficiently abstract to be reused in different settings.
- Pattern descriptions usually make use of object-oriented characteristics such as inheritance and polymorphism.

Patterns

- *Patterns and Pattern Languages are ways to describe best practices, good designs, and capture experience in a way that it is possible for others to reuse this experience.*

Pattern elements

Name A meaningful pattern identifier.

Problem description.

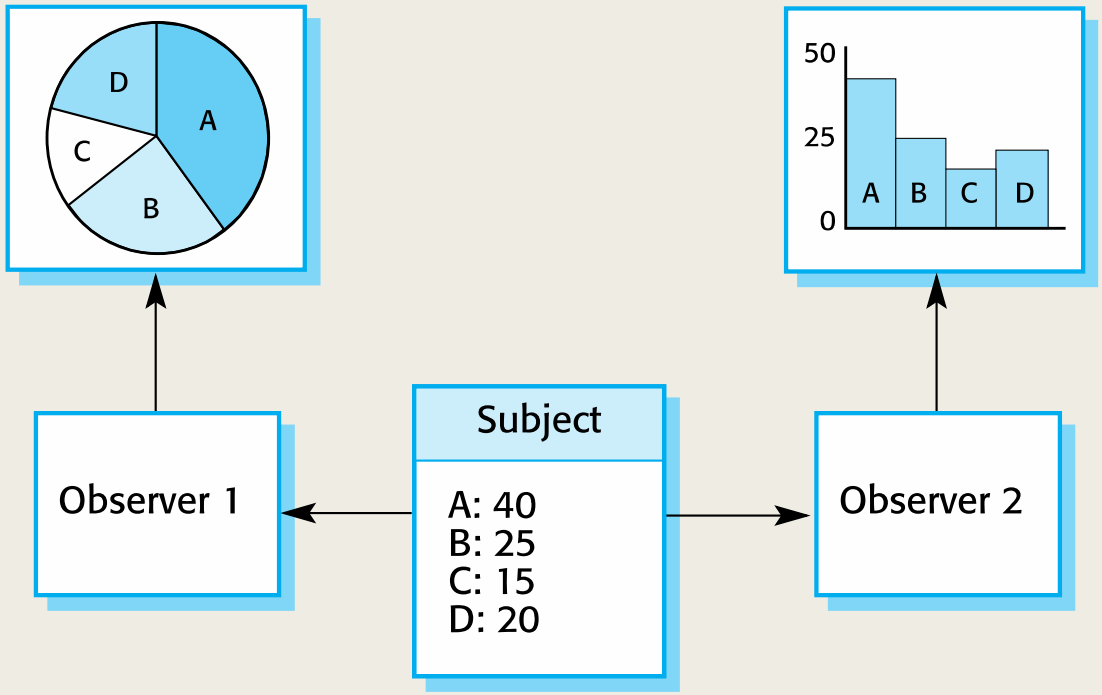
Solution description. Not a concrete design but a template for a design solution that can be instantiated in different ways.

Consequences The results and trade-offs of applying the pattern.

The Observer pattern

- Name
  - *Observer.*
- Description
  - *Separates the display of object state from the object itself.*
- Problem description
  - *Used when multiple displays of state are needed.*
- Solution description
  - *See slide with UML description.*
- Consequences
  - *Optimisations to enhance display performance are impractical.*

Multiple displays using the Observer pattern



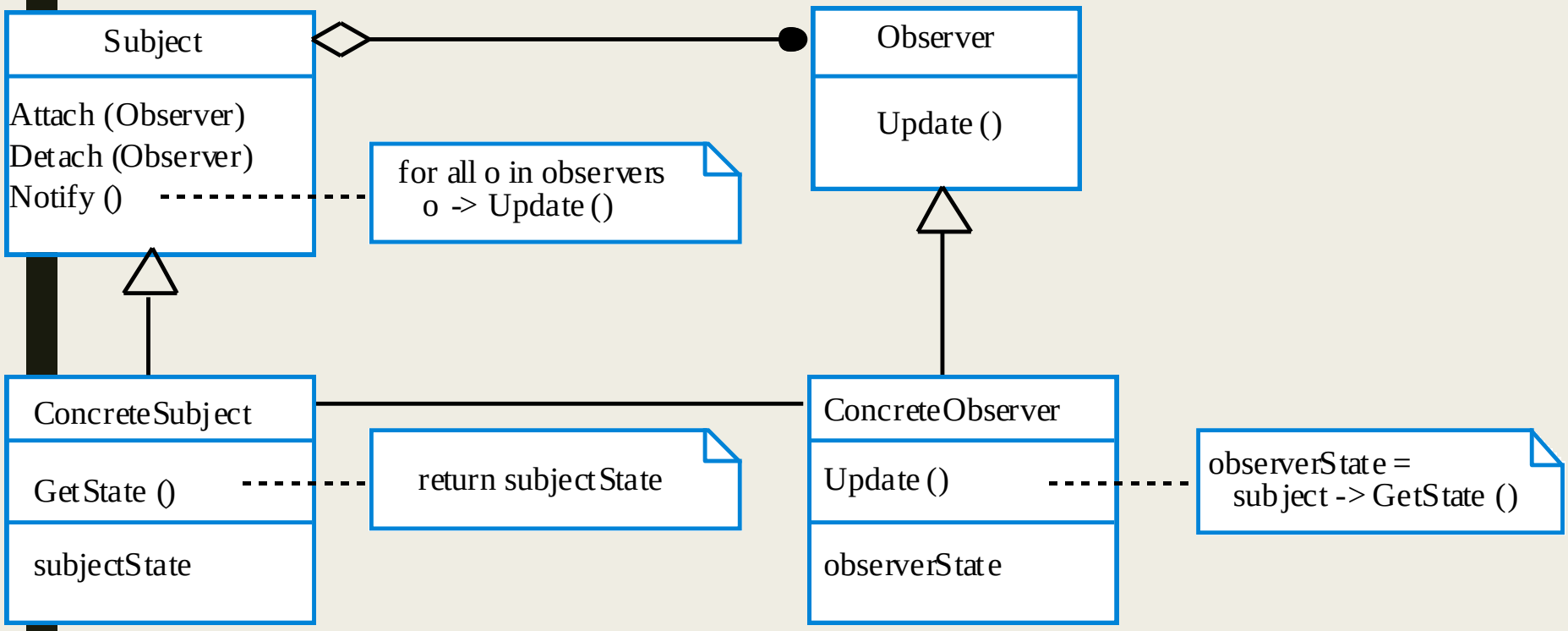
The Observer pattern (1)

Pattern name	Observer
Description	Separates the display of the state of an object from the object itself and allows alternative displays to be provided. When the object state changes, all displays are automatically notified and updated to reflect the change.
Problem description	In many situations, you have to provide multiple displays of state information, such as a graphical display and a tabular display. Not all of these may be known when the information is specified. All alternative presentations should support interaction and, when the state is changed, all displays must be updated. This pattern may be used in all situations where more than one display format for state information is required and where it is not necessary for the object that maintains the state information to know about the specific display formats used.

The Observer pattern (2)

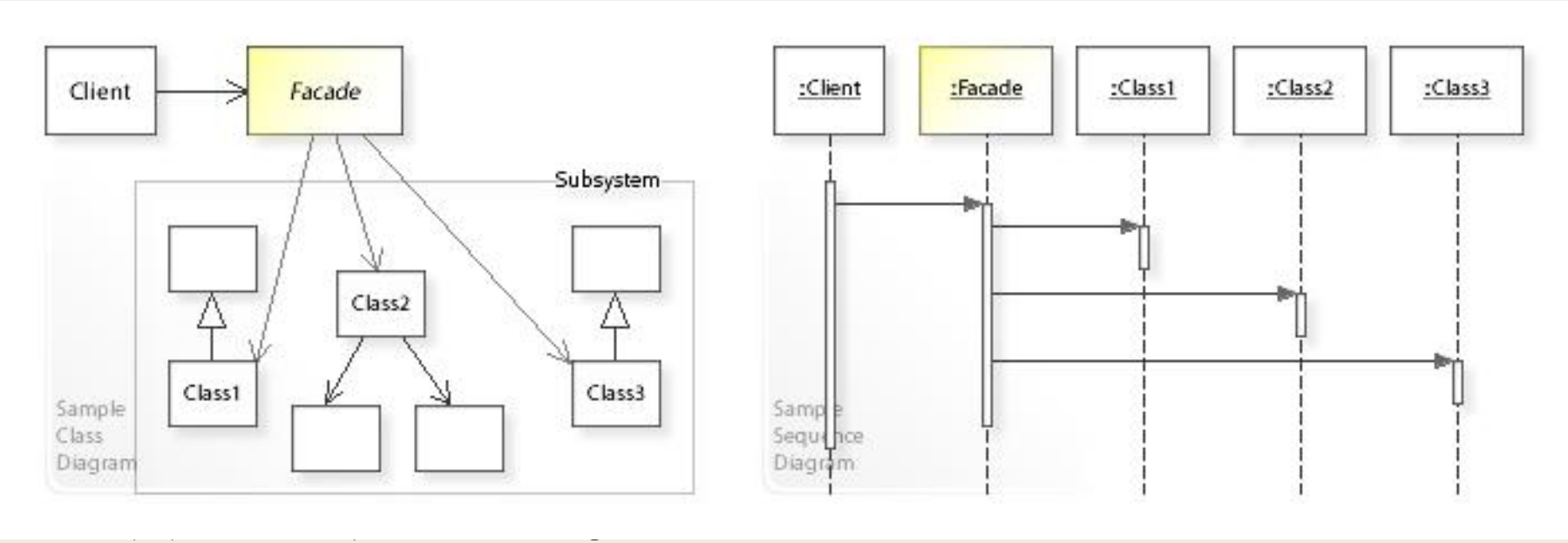
Pattern name	Observer
Solution description	This involves two abstract objects, Subject and Observer, and two concrete objects, ConcreteSubject and ConcreteObject, which inherit the attributes of the related abstract objects. The abstract objects include general operations that are applicable in all situations. The state to be displayed is maintained in ConcreteSubject, which inherits operations from Subject allowing it to add and remove Observers (each observer corresponds to a display) and to issue a notification when the state has changed. The ConcreteObserver maintains a copy of the state of ConcreteSubject and implements the Update() interface of Observer that allows these copies to be kept in step. The ConcreteObserver automatically displays the state and reflects changes whenever the state is updated.
Consequences	The subject only knows the abstract Observer and does not know details of the concrete class. Therefore there is minimal coupling between these objects. Because of this lack of knowledge, optimizations that enhance display performance are impractical. Changes to the subject may cause a set of linked updates to observers to be generated, some of which may not be necessary.

A UML model of the Observer pattern



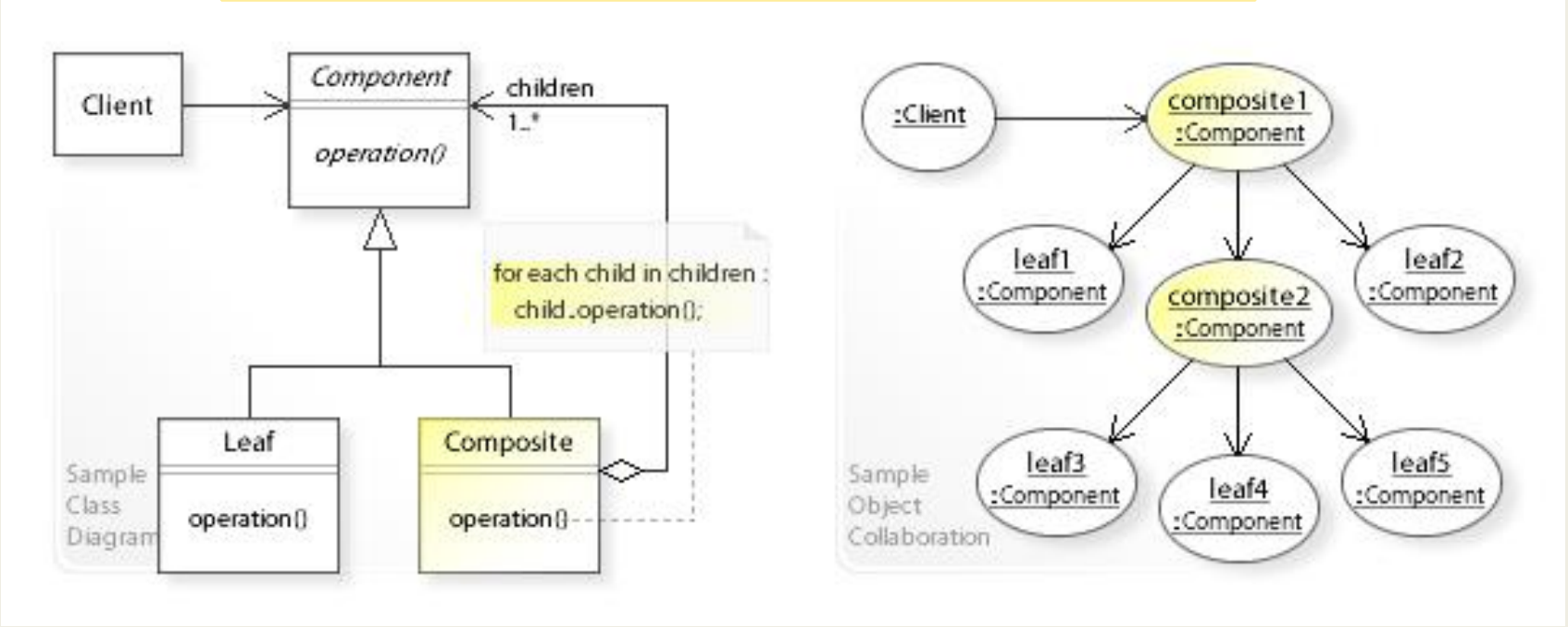
Façade pattern

- The facade pattern is typically used when
  - *a simple interface is required to access a complex system,*
  - *a system is very complex or difficult to understand,*
  - *an entry point is needed to each level of layered software, or*
  - *the abstractions and implementations of a subsystem are tightly coupled.*



Composite pattern

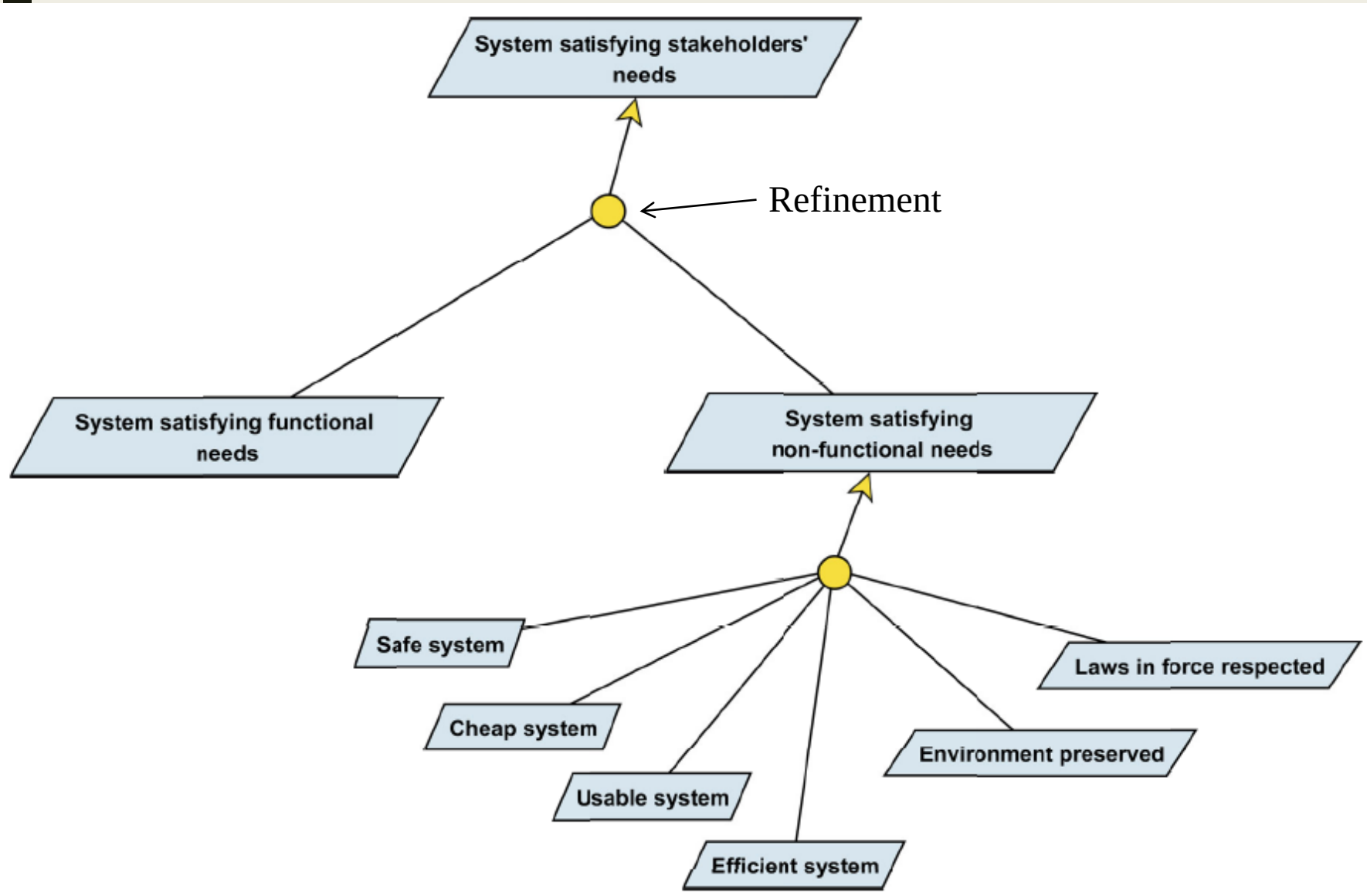
- What problems can the Composite design pattern solve?
  - *A part-whole hierarchy should be represented so that clients can treat part and whole objects uniformly.*



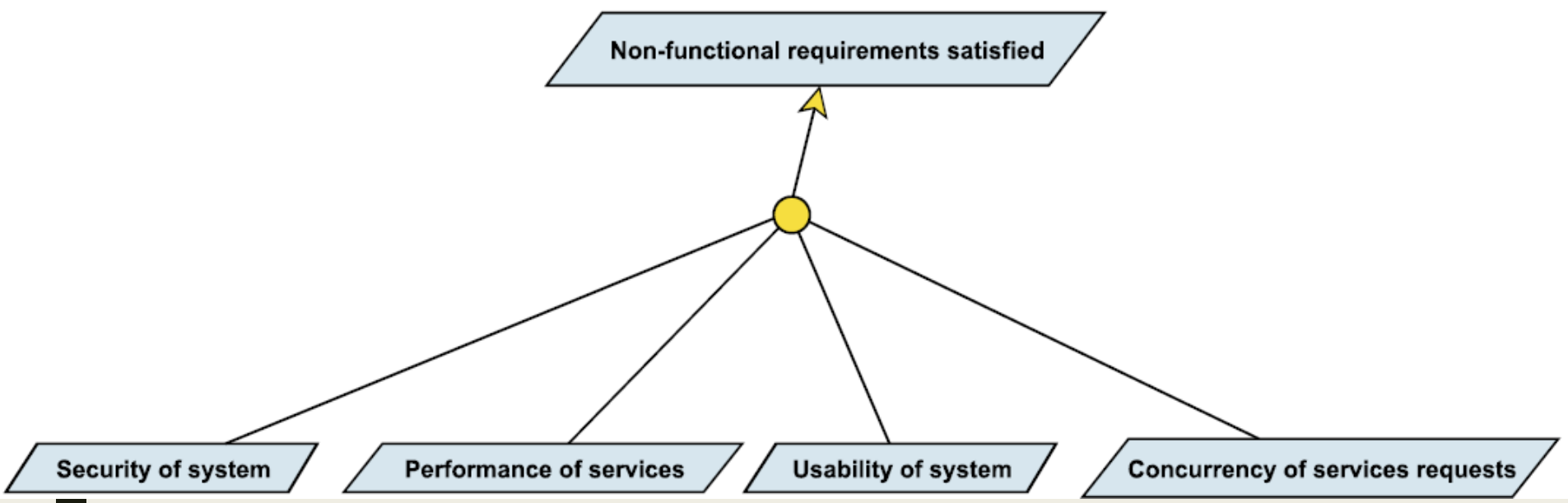
Goal Model and Patterns

- Reqs can be obtained through interviews or docs. Or with the help of patterns
- Requirements patterns
  - *An efficient way is to reuse patterns*
- KAOS consists of modelling generic patterns of reqs.
  - *They are progressively built*

Generic Goal Pattern

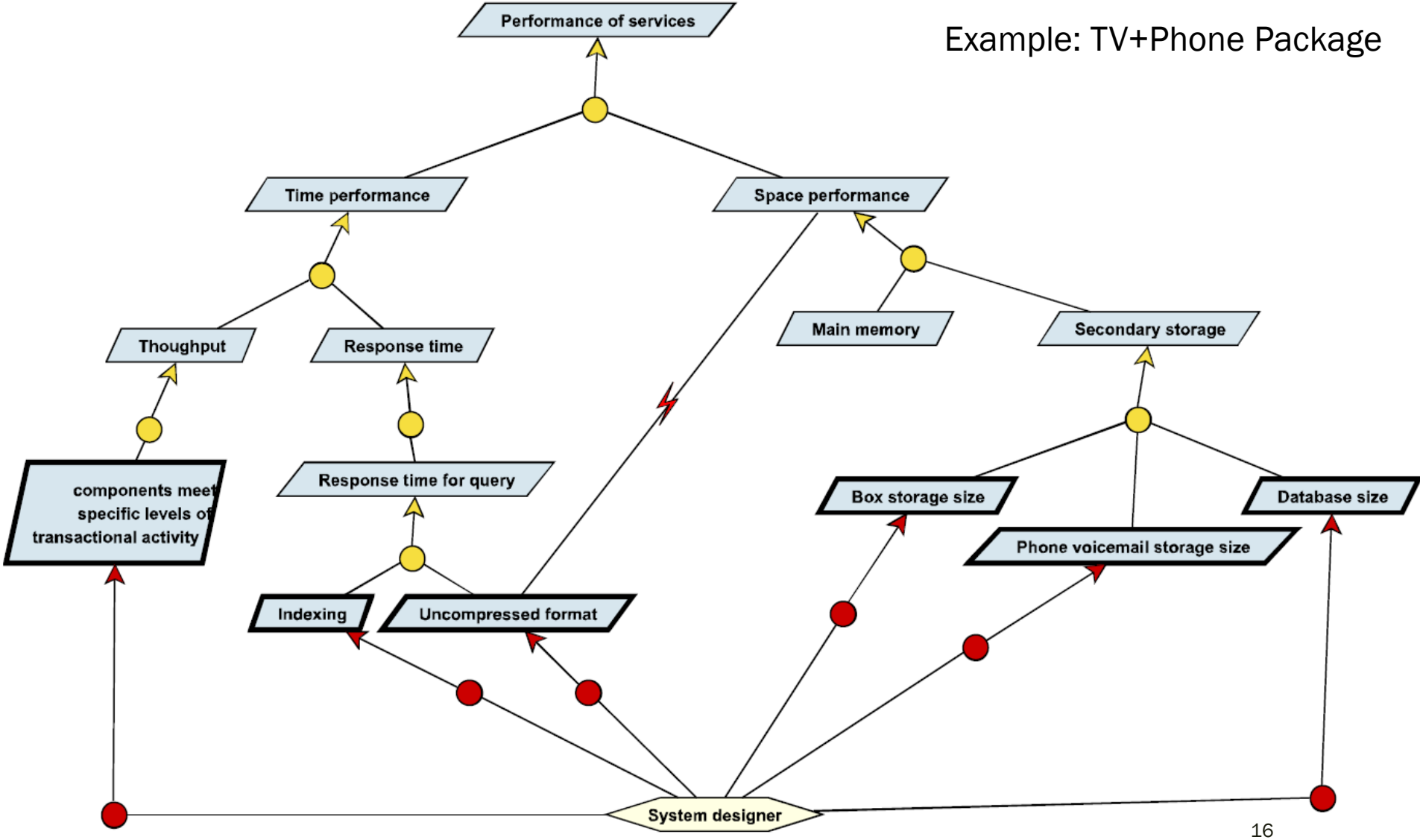


Non-functional requirements

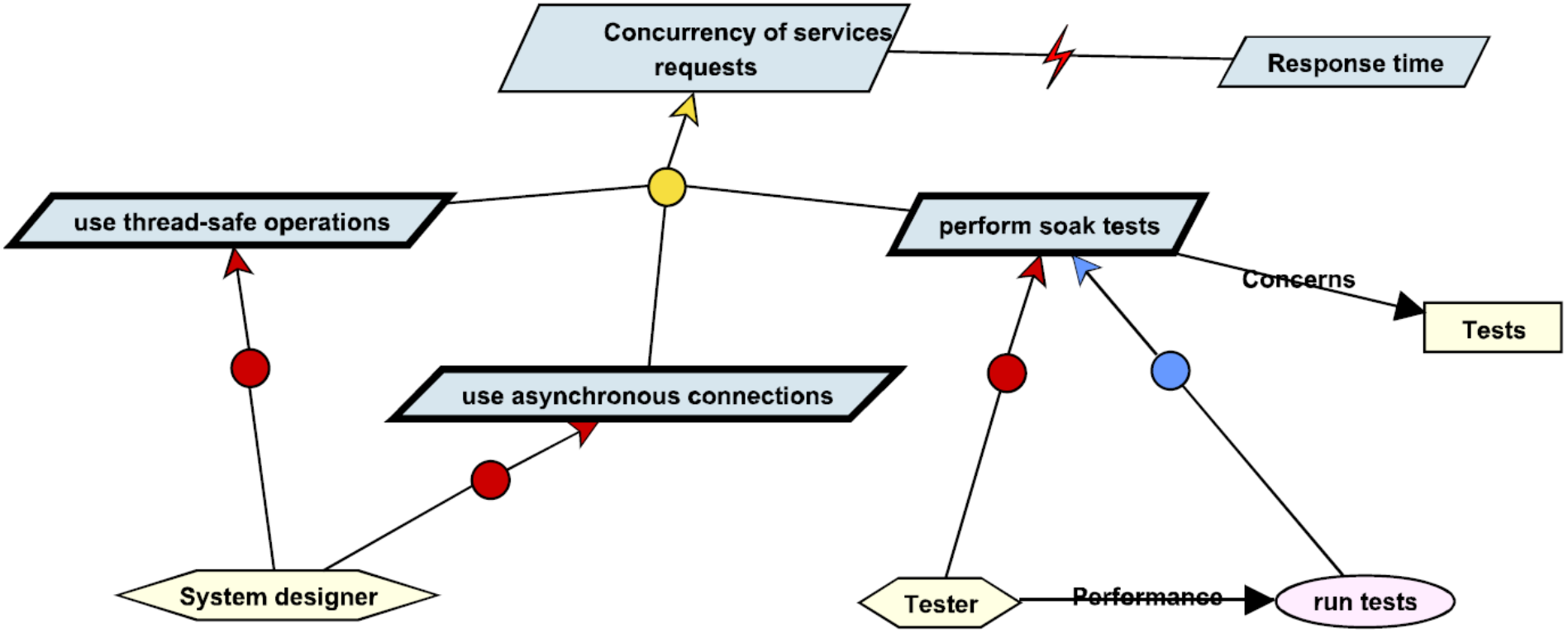


Performance

Example: TV+Phone Package

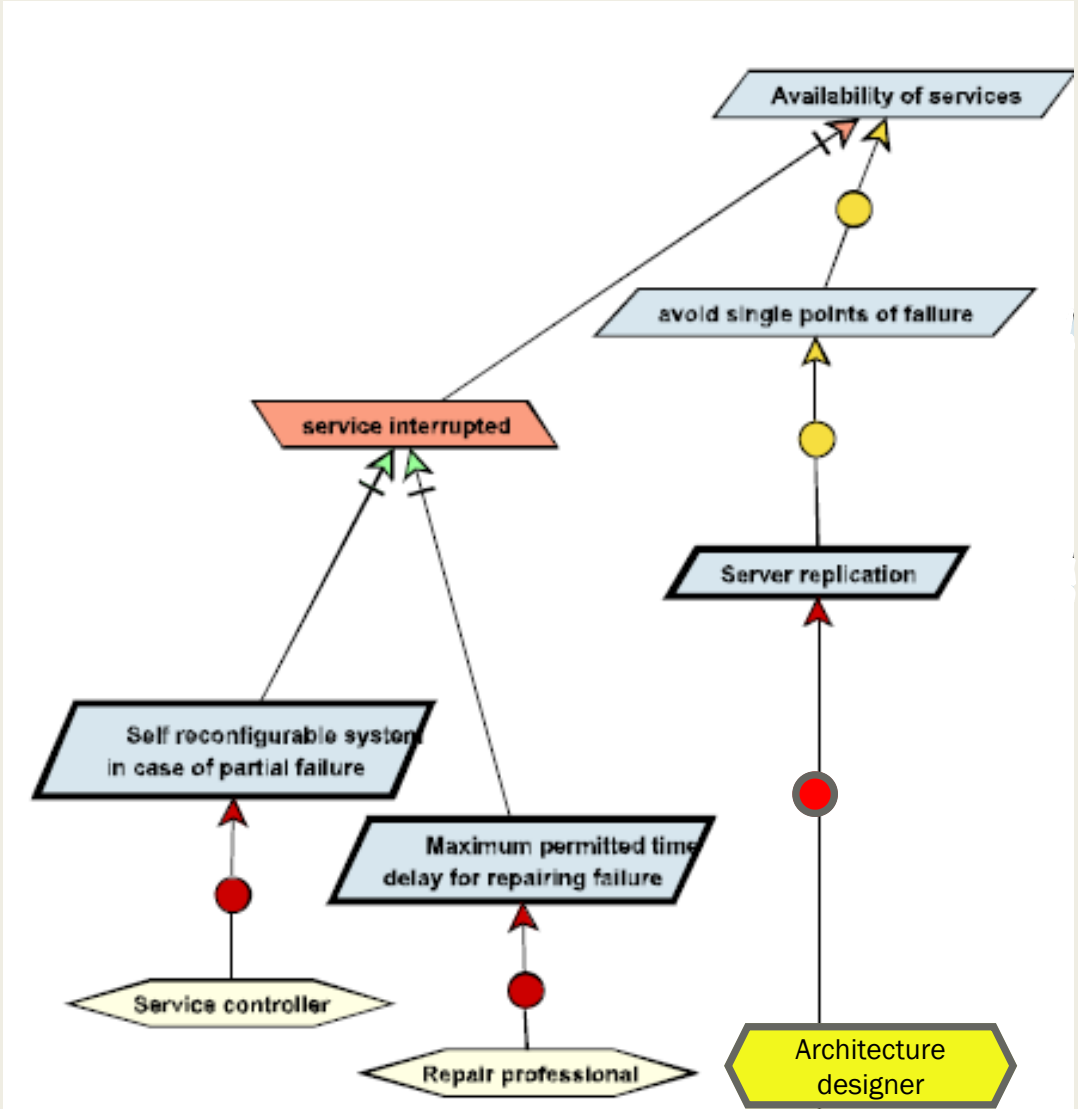


Concurrency

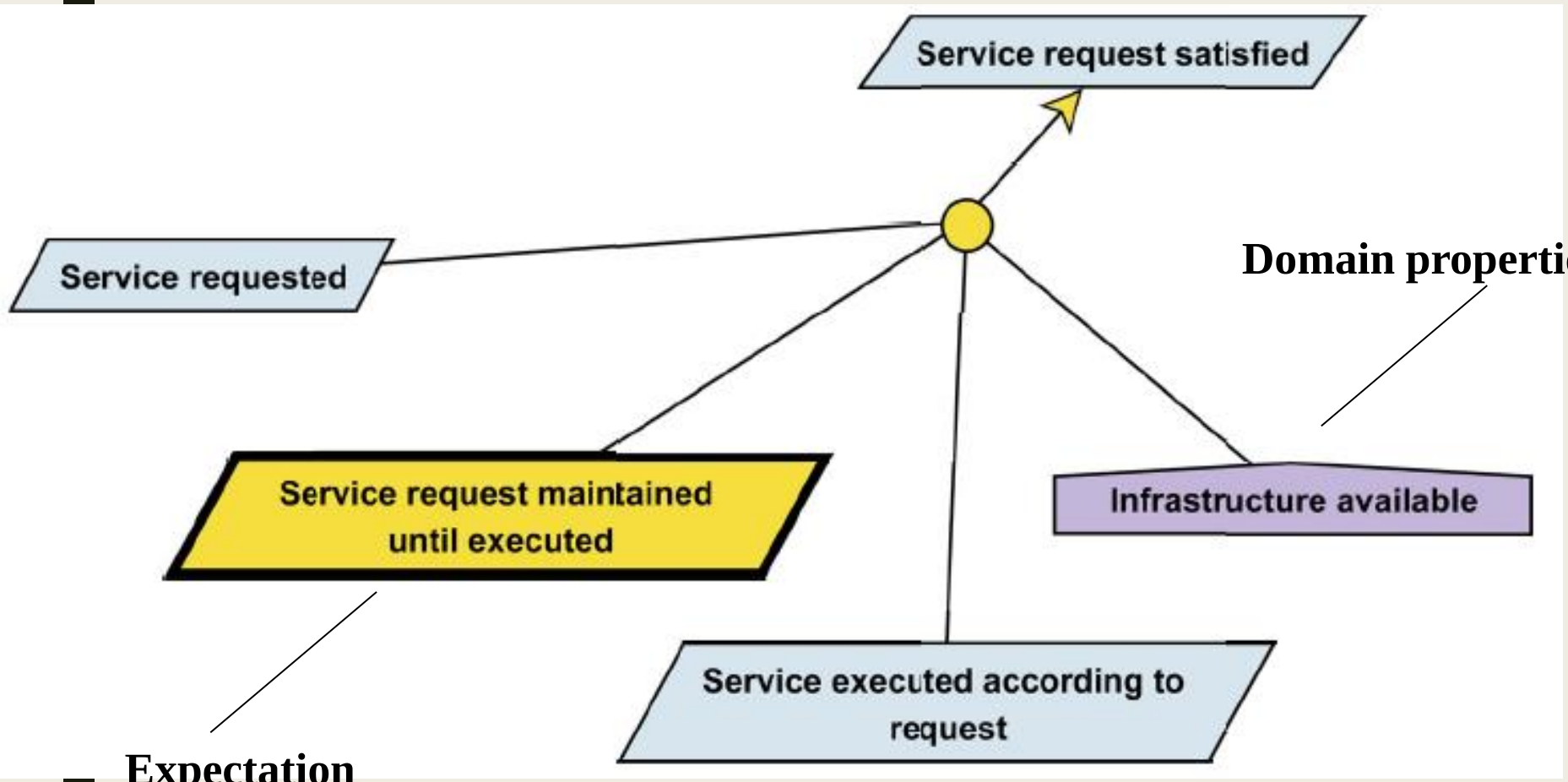


Thread safe: Implementation is guaranteed to be free of race conditions (when an application depends on the sequence or timing of processes or threads for it to operate properly)when accessed by multiple threads simultaneously.

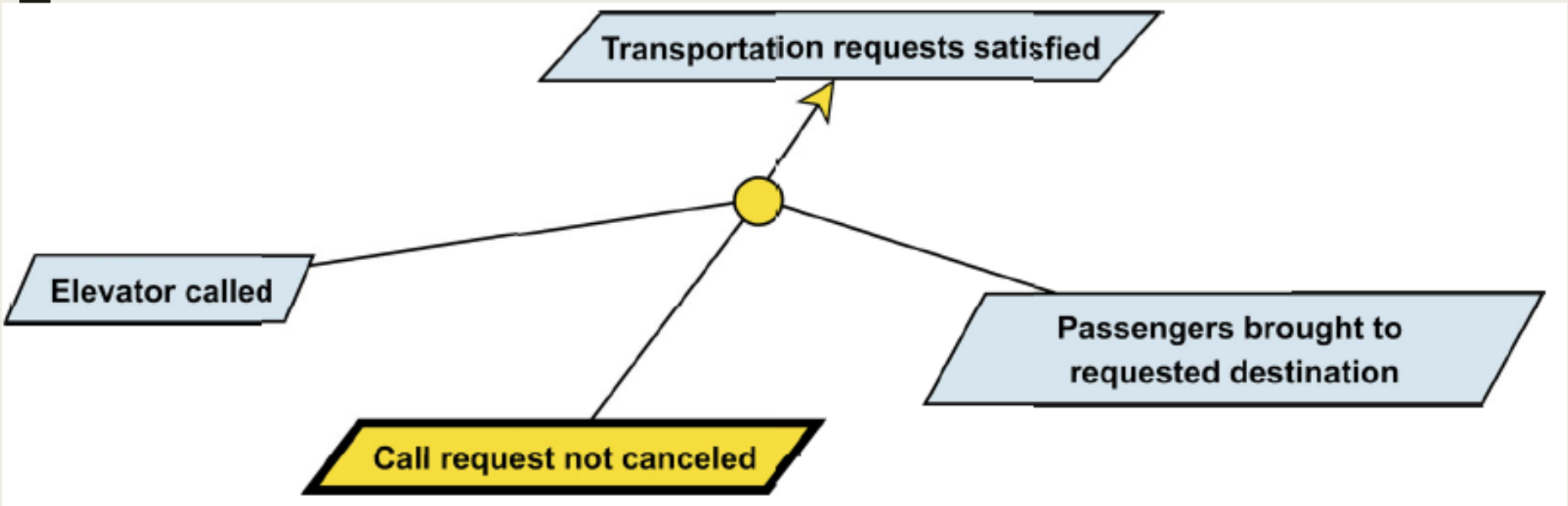
Availability



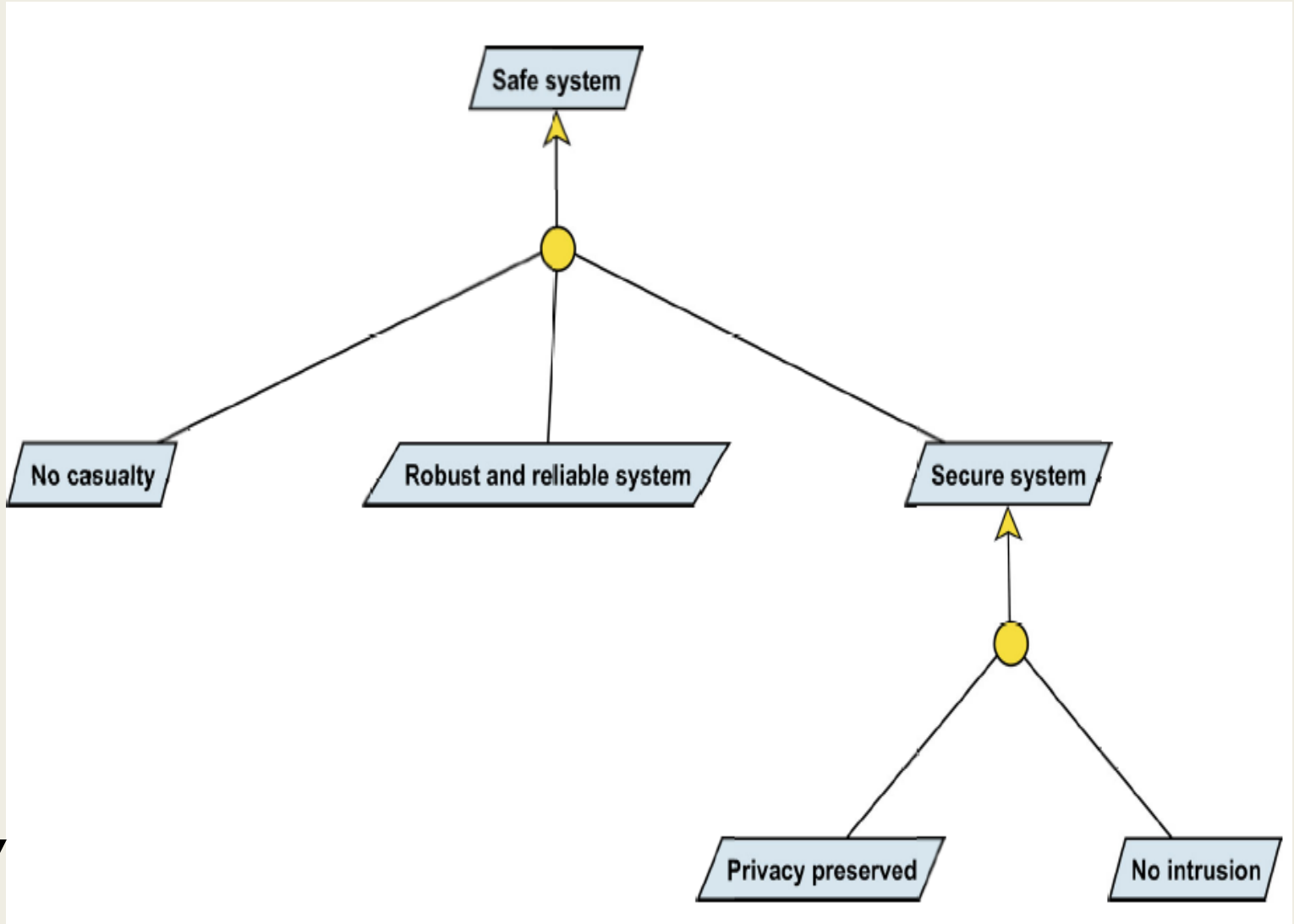
Generic Pattern: Expectations and Domain Properties

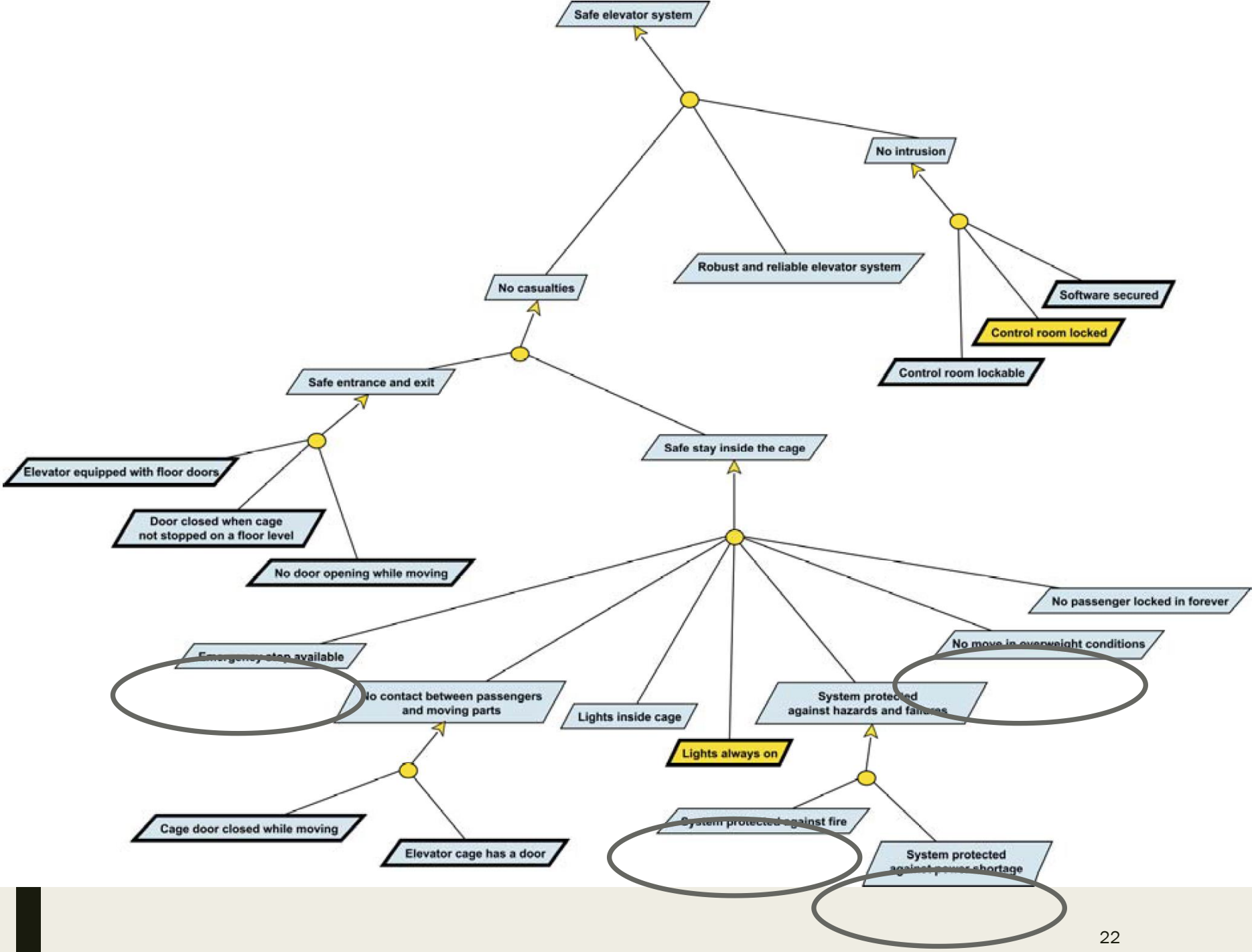


Application of the pattern to the elevator system

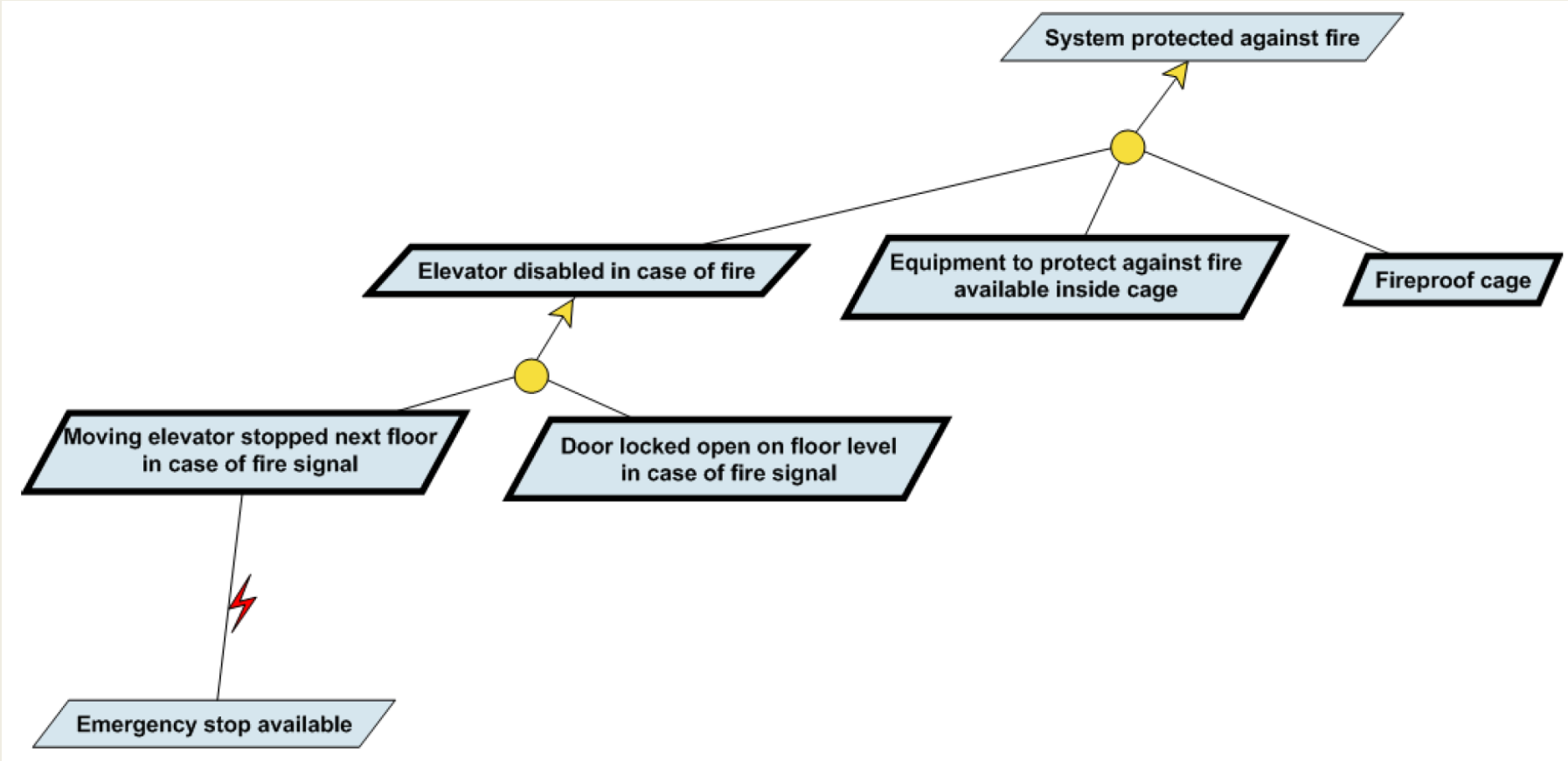


Generic NFR *Goal* for a Safe System

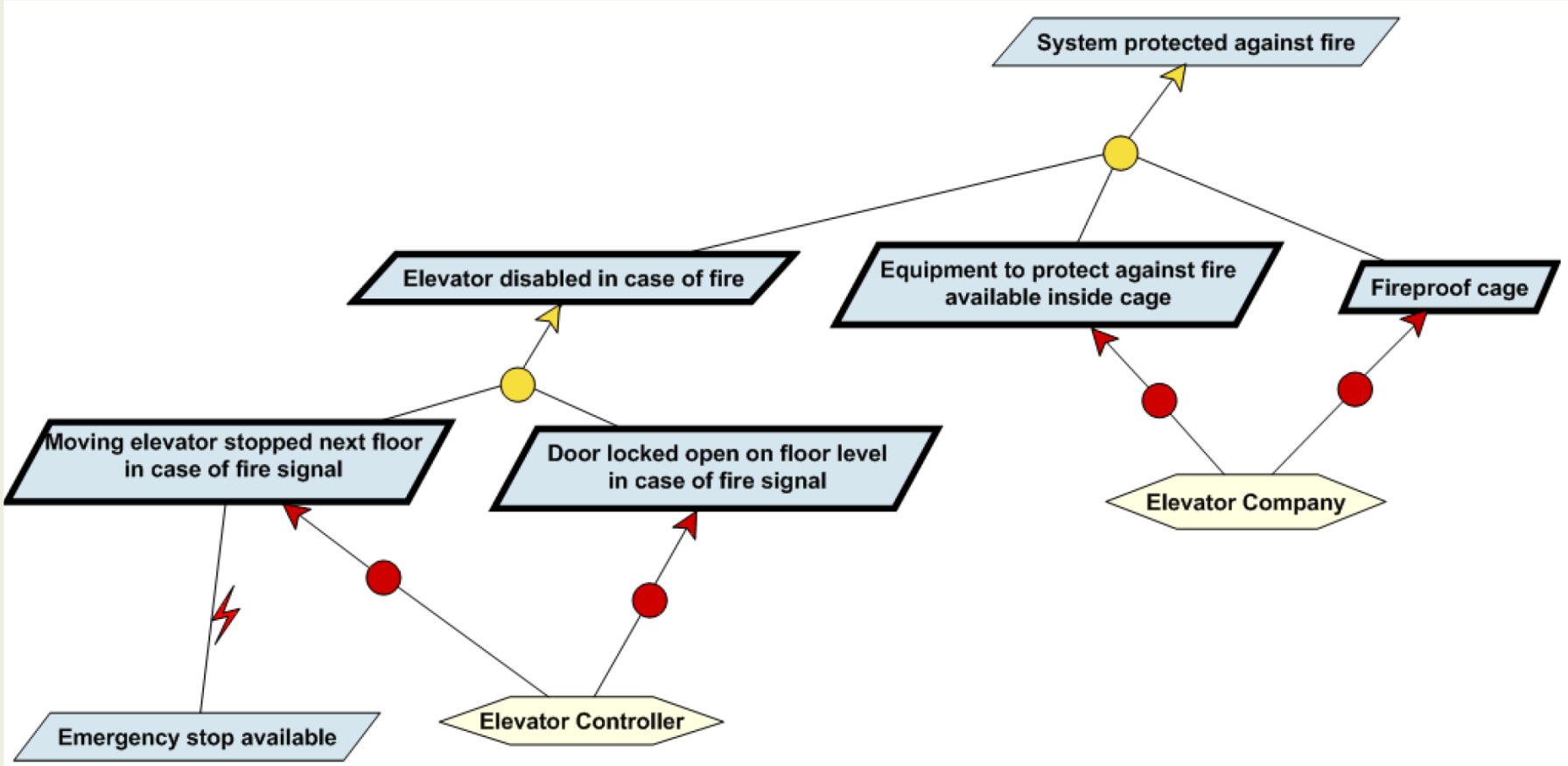



Safe elevator system
No intrusion
Robust and reliable elevator system
No casualties
Software secured
Control room locked
Control room lockable
Safe entrance and exit
Safe stay inside the cage
Elevator equipped with floor doors
Door closed when cage not stopped on a floor level
No door opening while moving
No passenger locked in forever
No move in overweight conditions
Emergency stop available
No contact between passengers and moving parts
Lights inside cage
System protected against hazards and failures
Lights always on
Cage door closed while moving
Elevator cage has a door
System protected against fire
System protected against power shortage

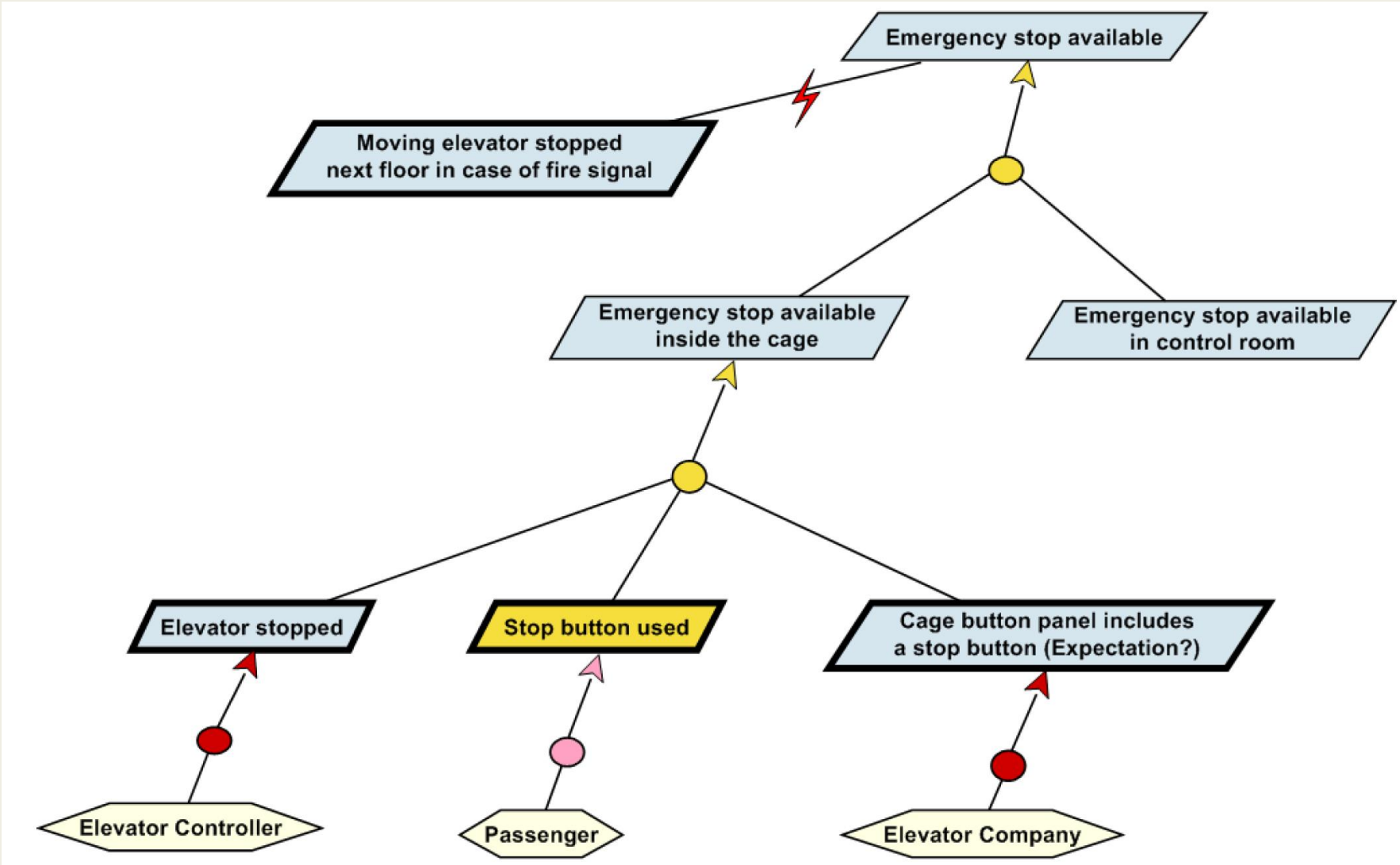
System protected against fire



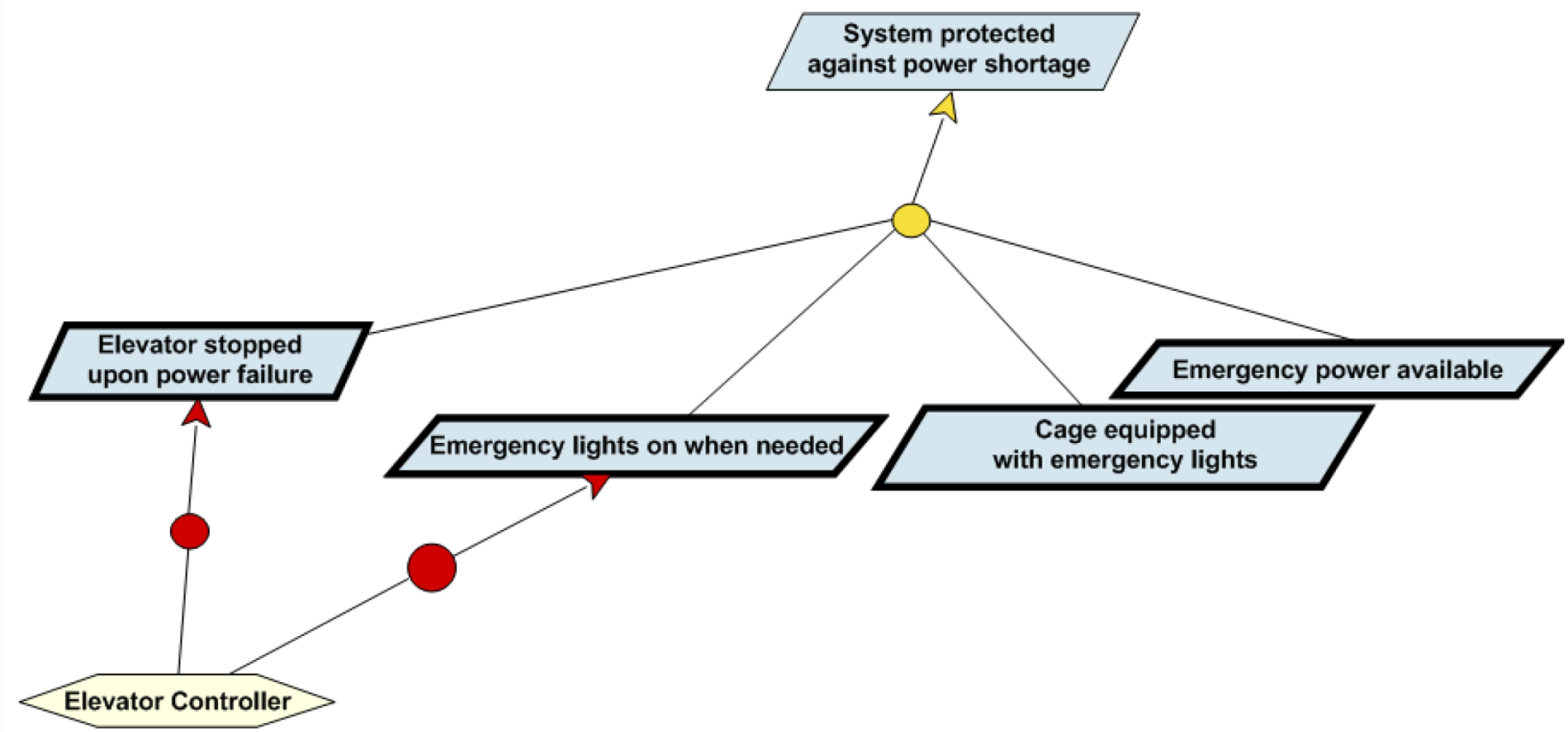
System protected against fire



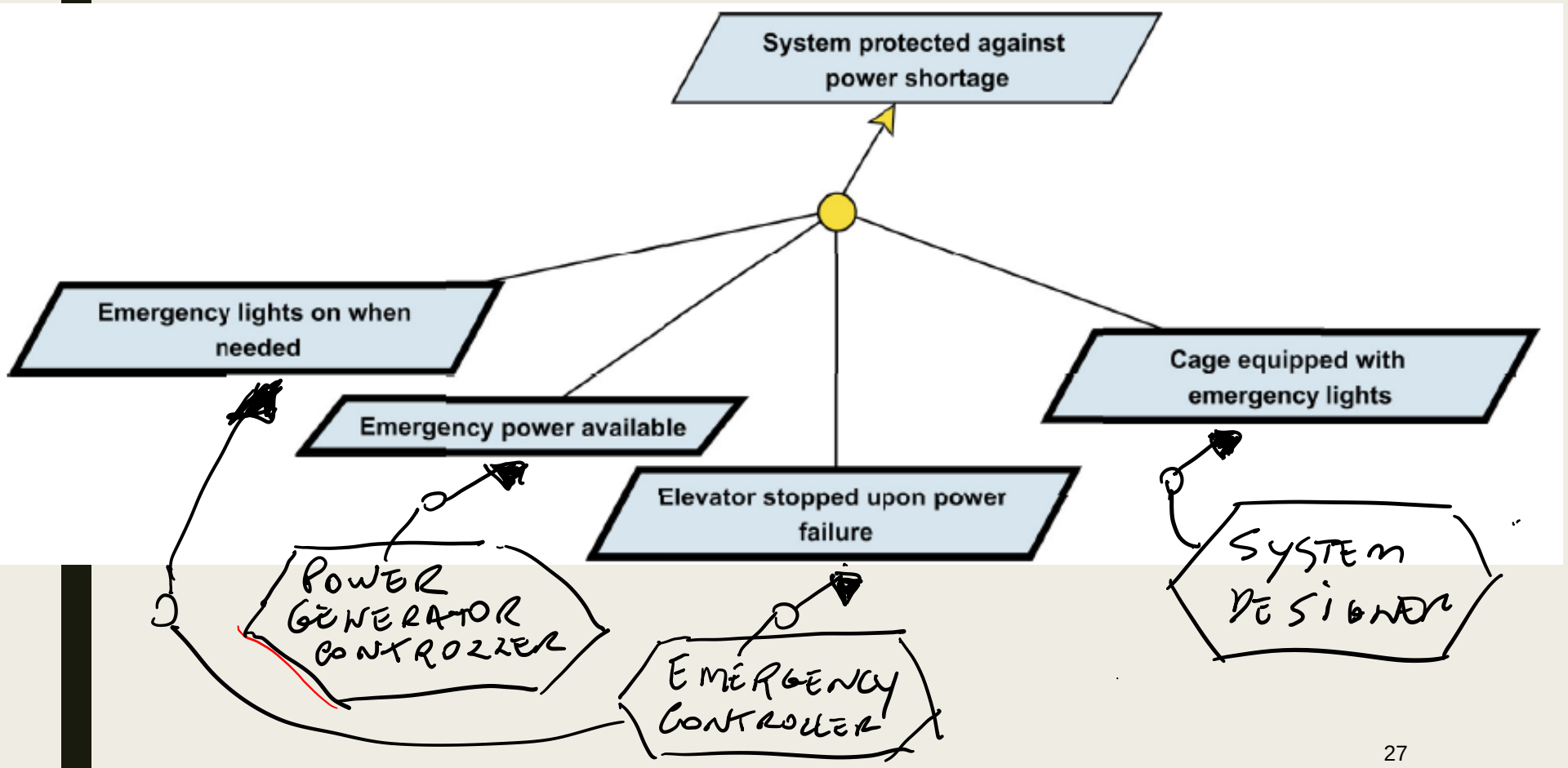
Emergency stop available



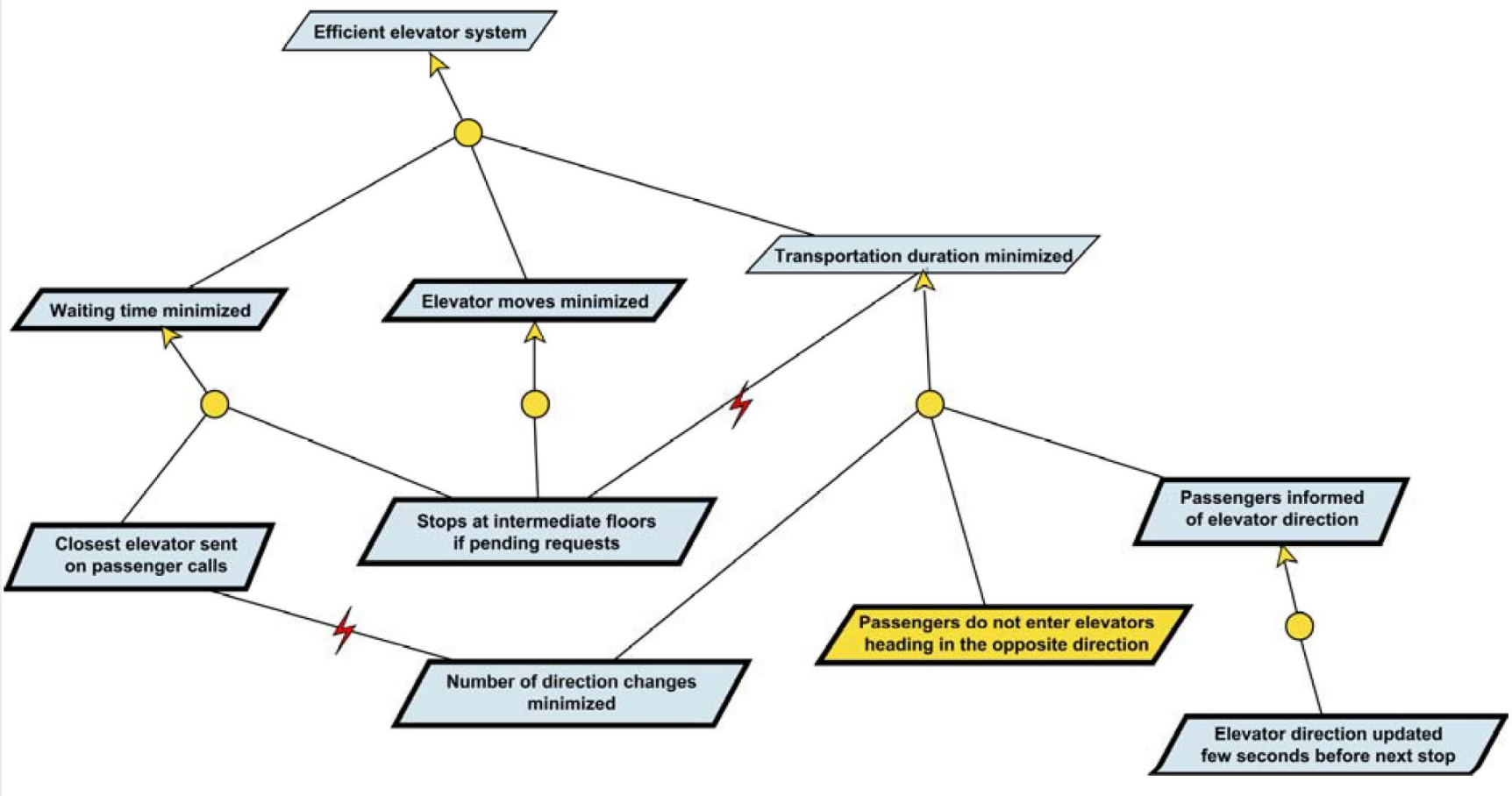
System protected against power shortage

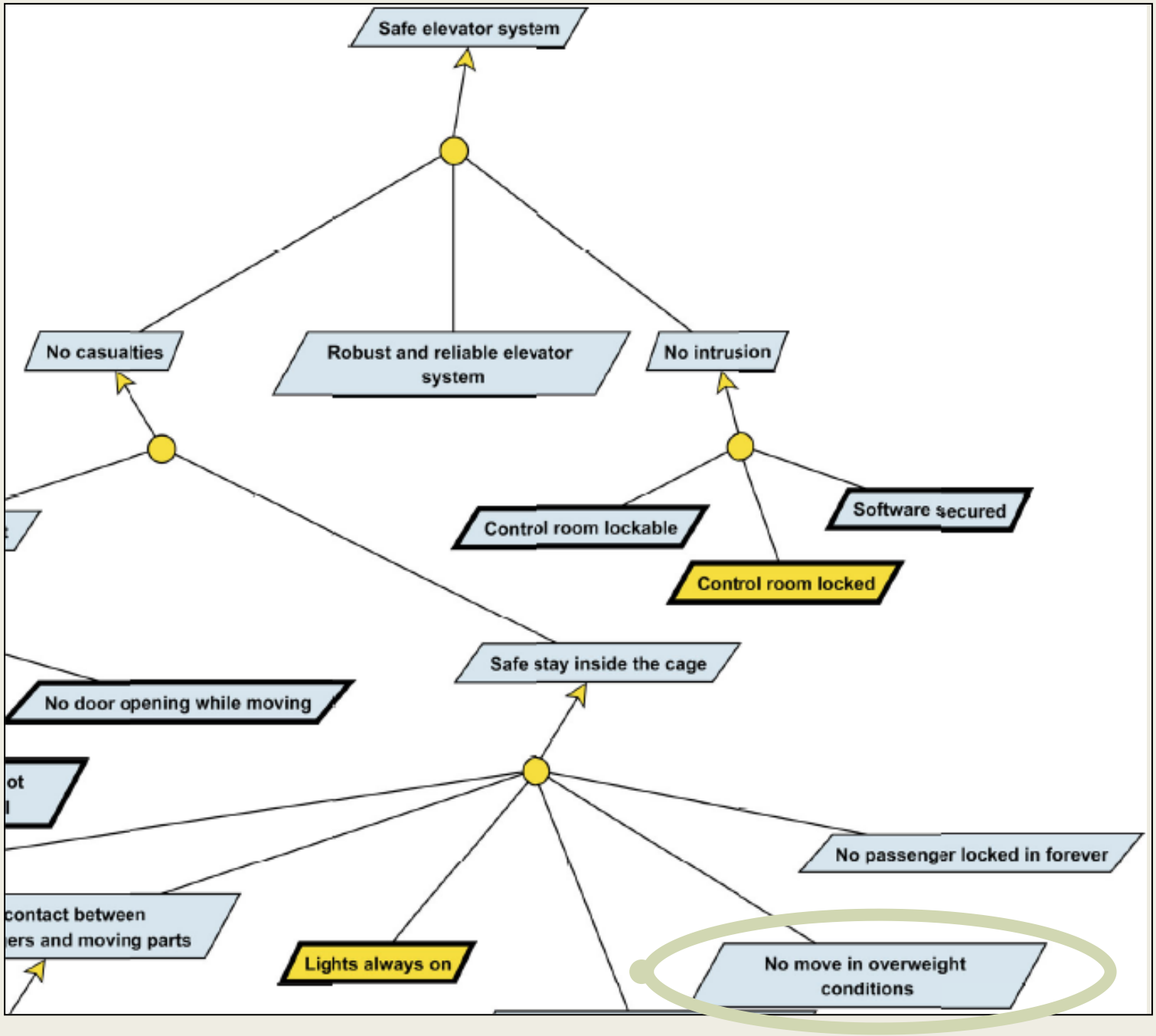


Concrete model for NFR Safety: Elevator System protected against power failure

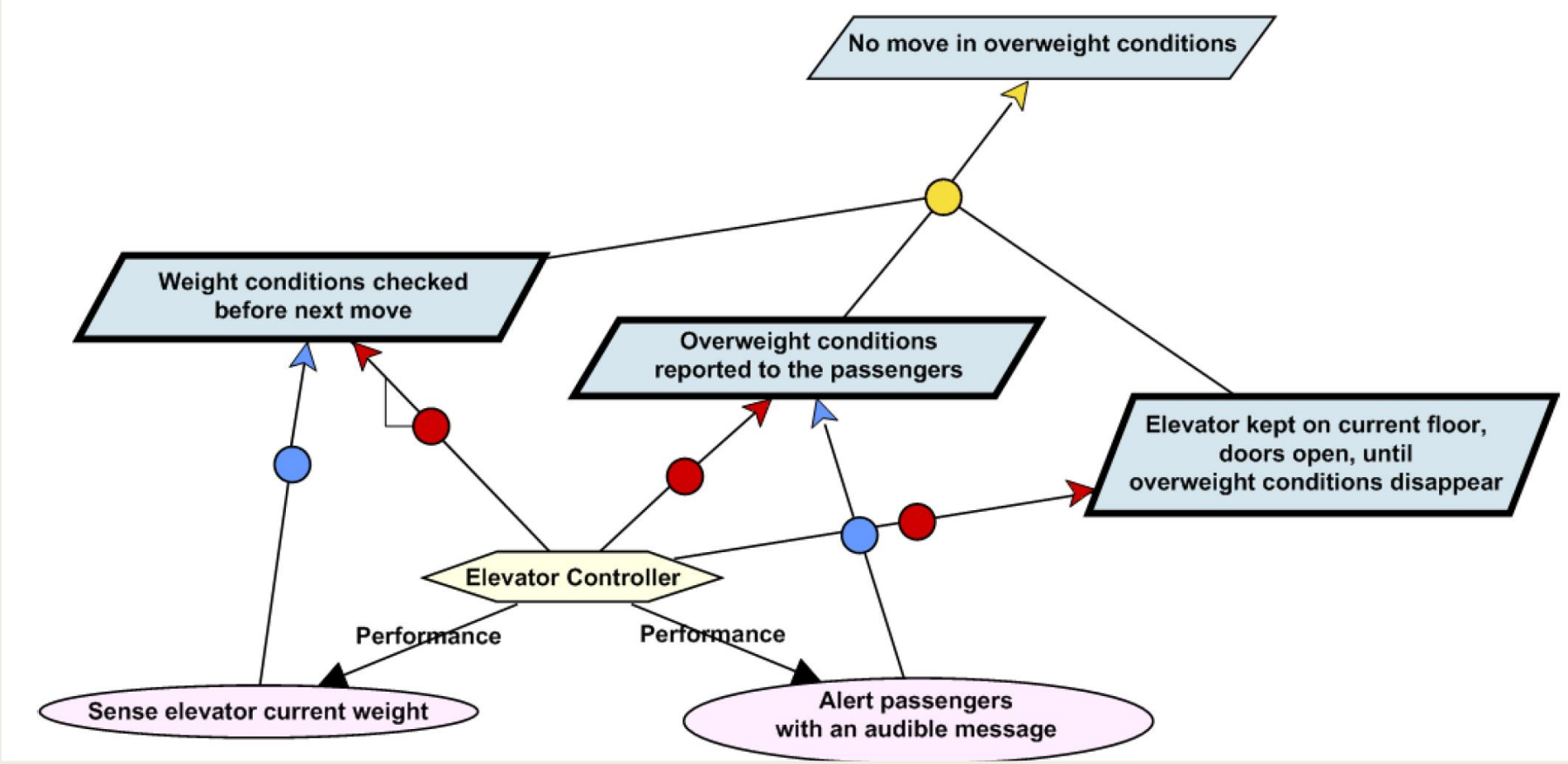


Efficient elevator system



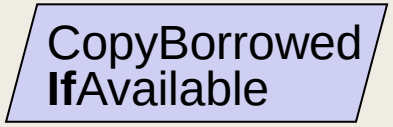

Safe elevator system
No casualties
Robust and reliable elevator system
No intrusion
Control room lockable
Software secured
Control room locked
Safe stay inside the cage
No door opening while moving
No passenger locked in forever
Lights always on
No move in overweight conditions

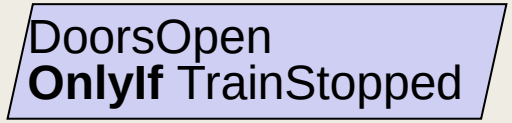
No move in overweight conditions



Building goal models: bad smells

- Do not confuse …
  - *goal …*





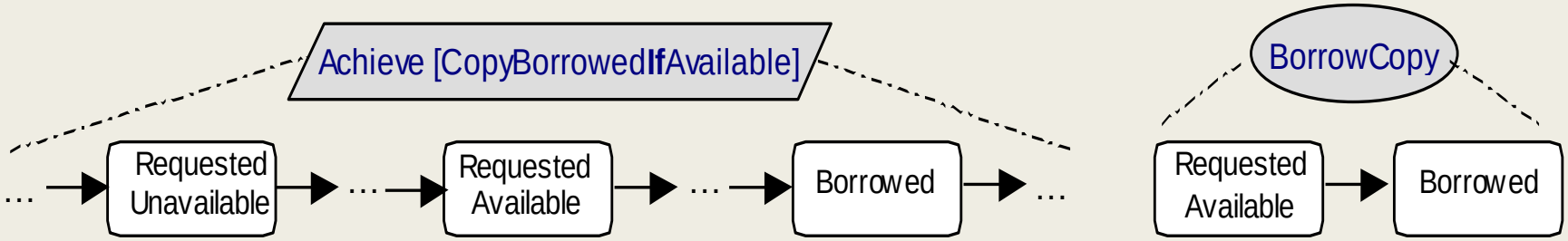
  - *operation …*

Goal ≠ service from functional model *(e.g. use case)*

  - *Services **operationalize** functional, leaf goals in refinement graph*
    - a goal is often operationalized through multiple operations
    - an operation often operationalizes multiple goals
  - *Soft goals are often not operationalized in functional model but used to select among alternatives*

Behavioral goals vs. operations

- Semantic difference
  - *Behavioral goals constrain entire sequences of state transitions*
  - *Operations constrain single state transitions*



- **Tip:** use **past participle** for goal name

  (state to be reached/maintained, quantity to be reduced/increased, ...)

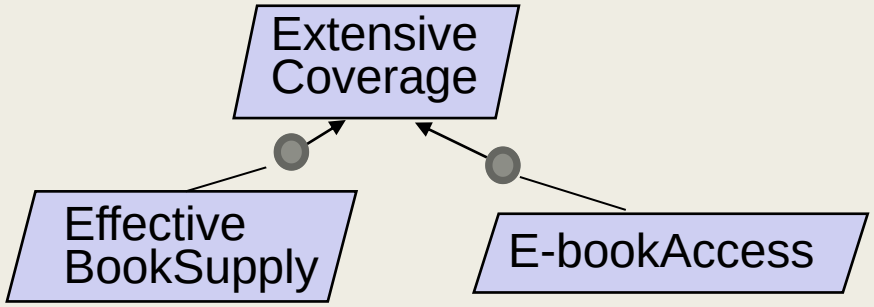
  use **infinitive** for operation name

  (action to reach/maintain that state)

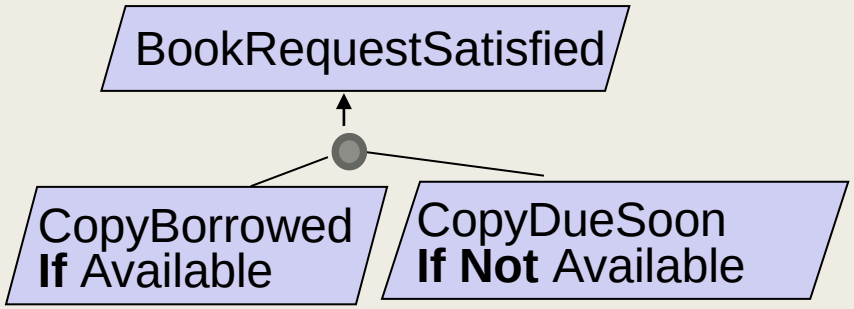
Building goal models: bad smells (2)

Do not confuse …

- *OR-refinement …*



- *AND-refinement by case …*



cf. case analysis:

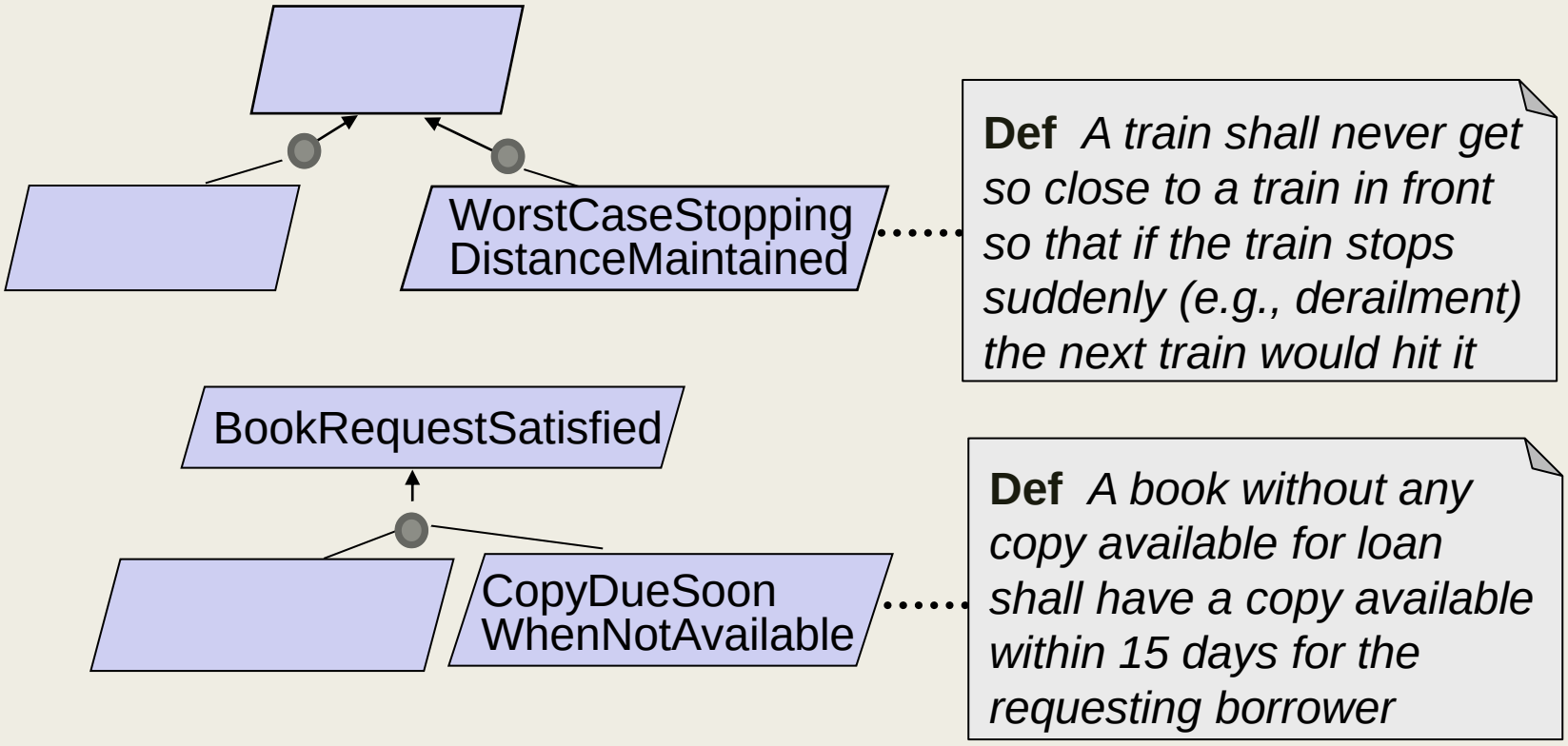
(Case1 ***or*** *Case2) ⇒ X* ***equiv*** *(Case1 ⇒ X)* ***and*** *(Case2 ⇒ X)*

OR*-refinement introduces* ***alternative*** *systems to reach parent goal*

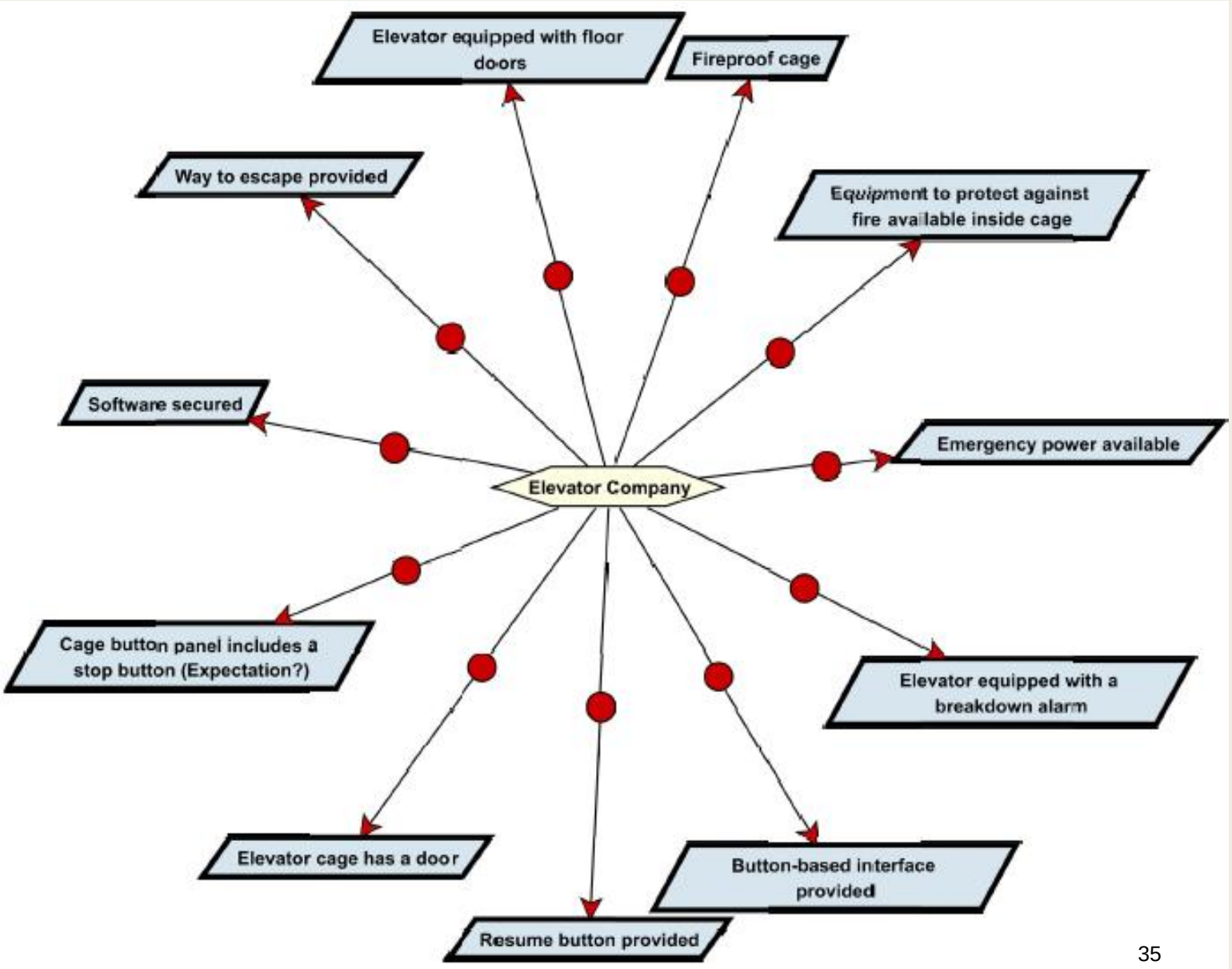
AND*-refinement by cases introduces complementary, conjoined subgoals within* ***same*** *system*

Building goal models: bad smells

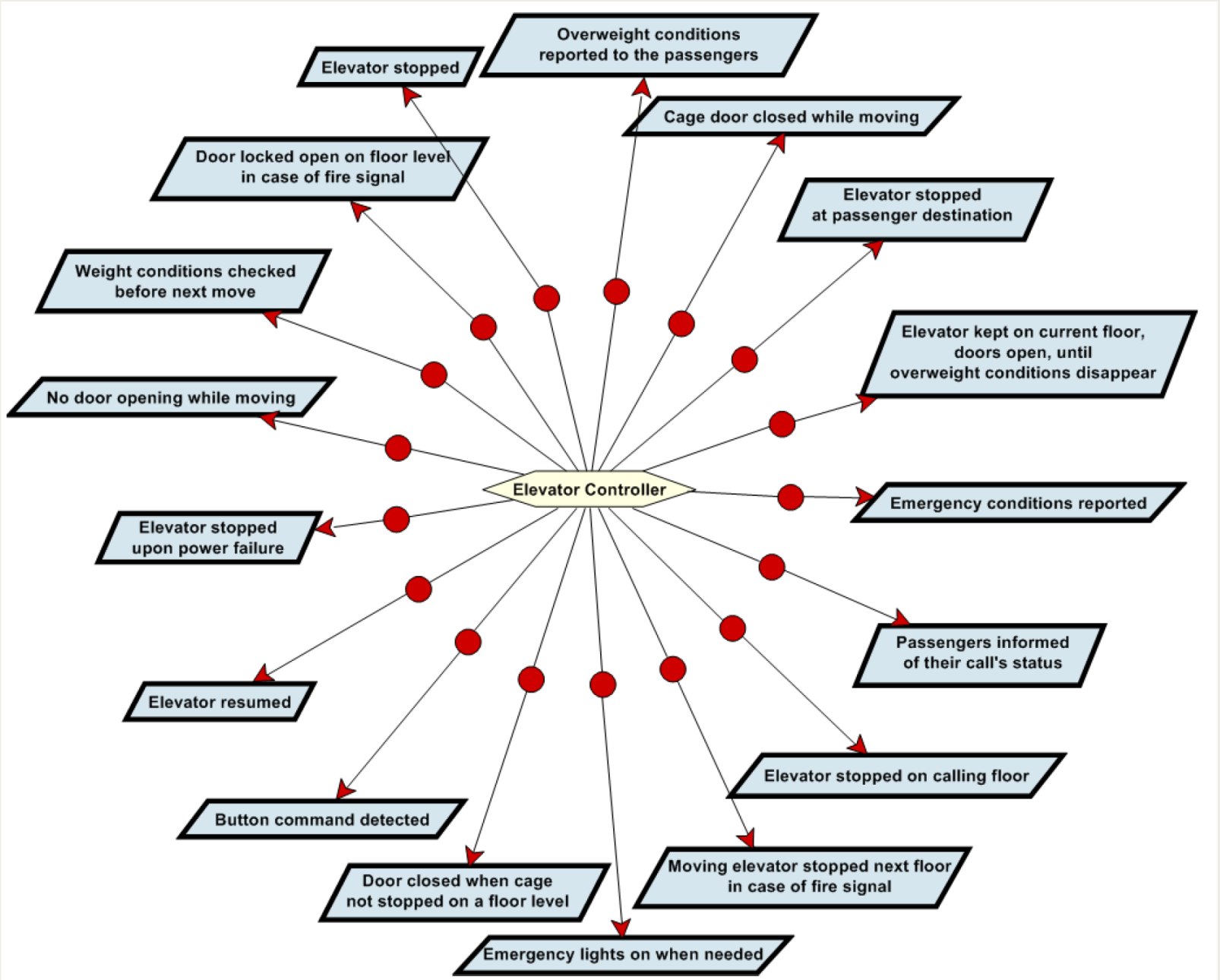
- Avoid ambiguities in goal specification & interpretation ...
  - *a precise & complete goal **definition** is essential*
  - *grounded on shared system phenomena, and agreed upon by all stakeholders*



Responsibilities of an agent



Responsibilities of elevator controller



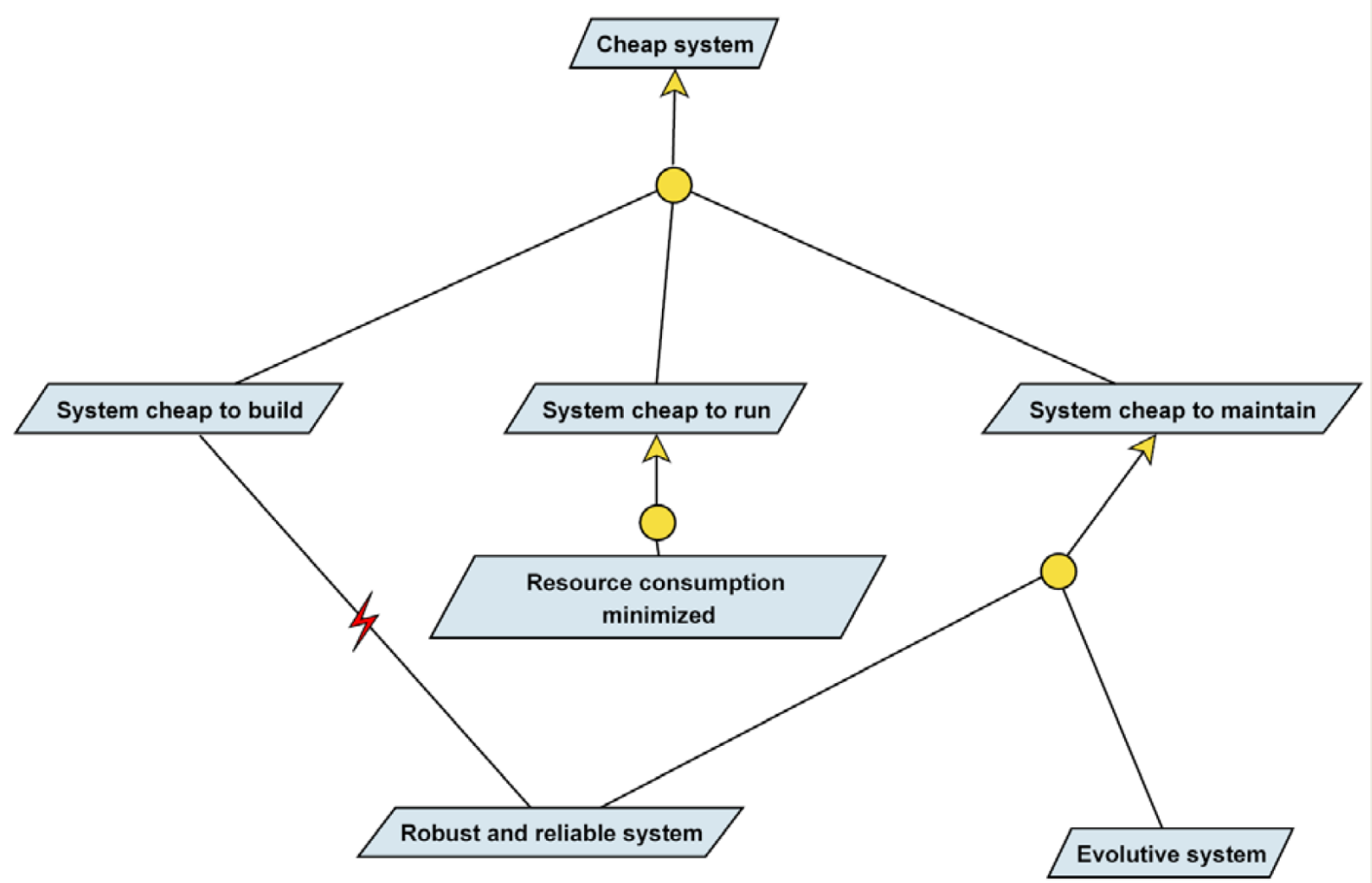
Conflicting goals

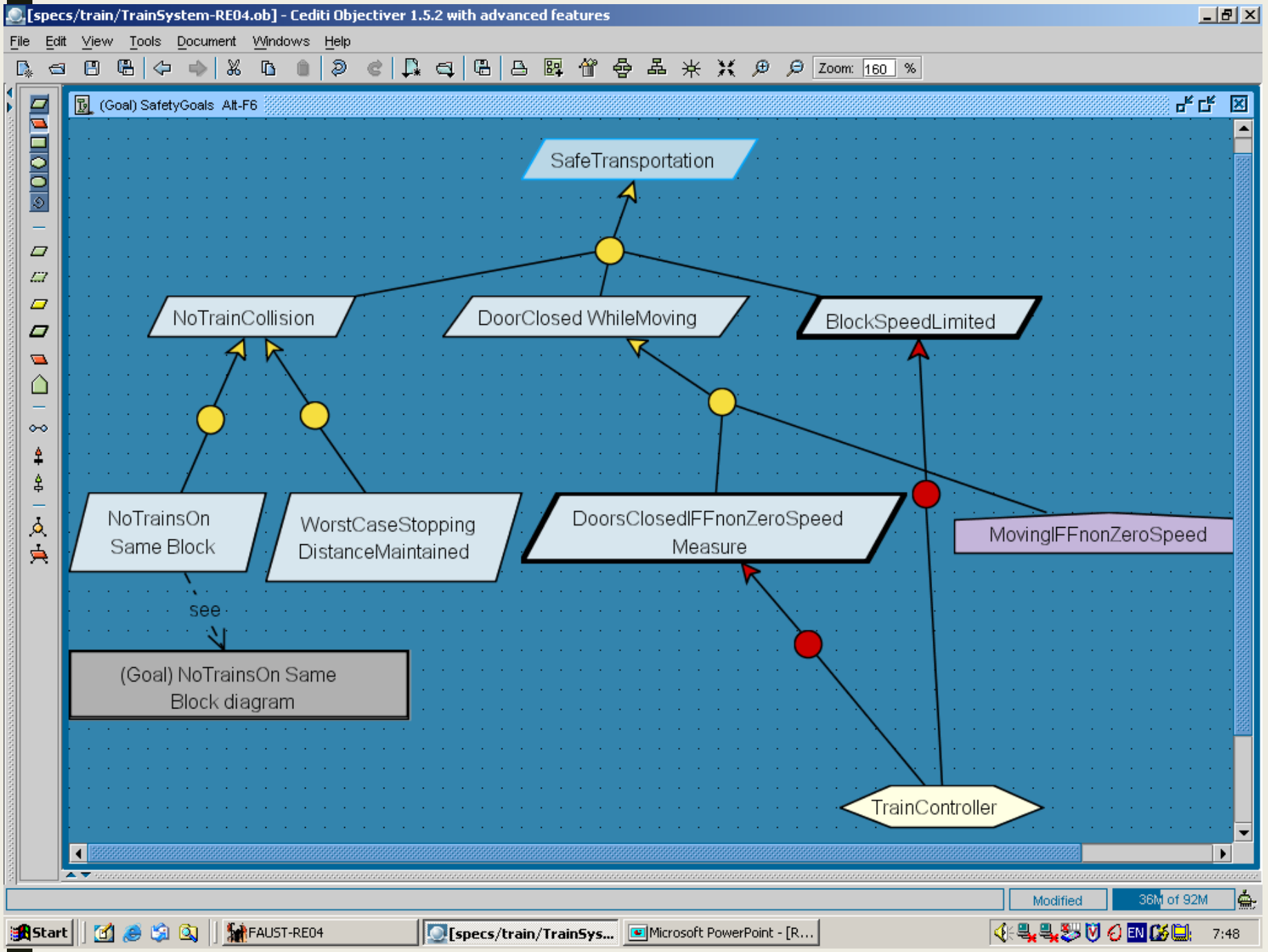
- When it is not possible to completely satisfy two goals simultaneously
  - *Performance goals may conflict with safety goals*
  - *Information goals may conflict with security and privacy goals*
- Dealing with conflicts (or more generally, with obstacles) allows
  - *to build a more complete requirements document and*
  - *To build a more robust system*
- Obstacles prevent goals from being achieved
  - *Defensive approach*

Conflict management

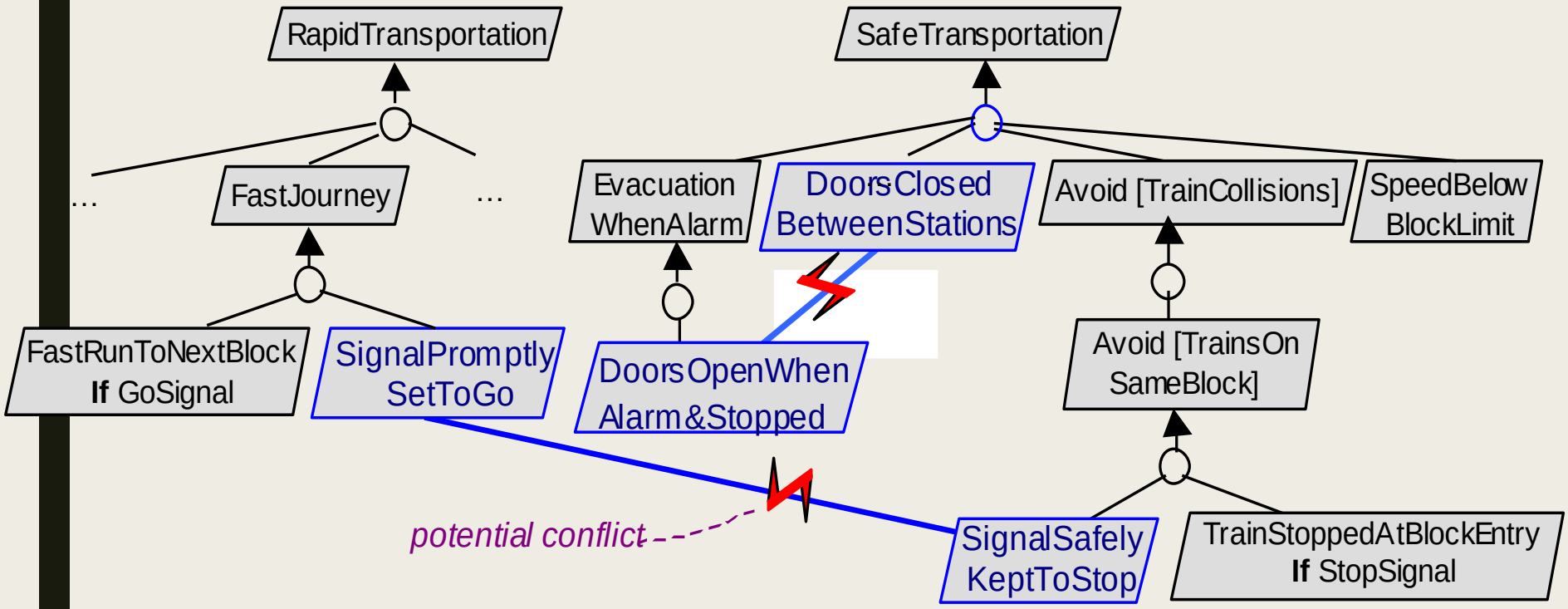
- When conflicts are detected:
  - *Negotiation to conflict resolution*
    - Select alternatives or
    - Re-evaluate the priorities or
    - Revise requirements

Conflict Identification: Generic goal for economic aspects of a system

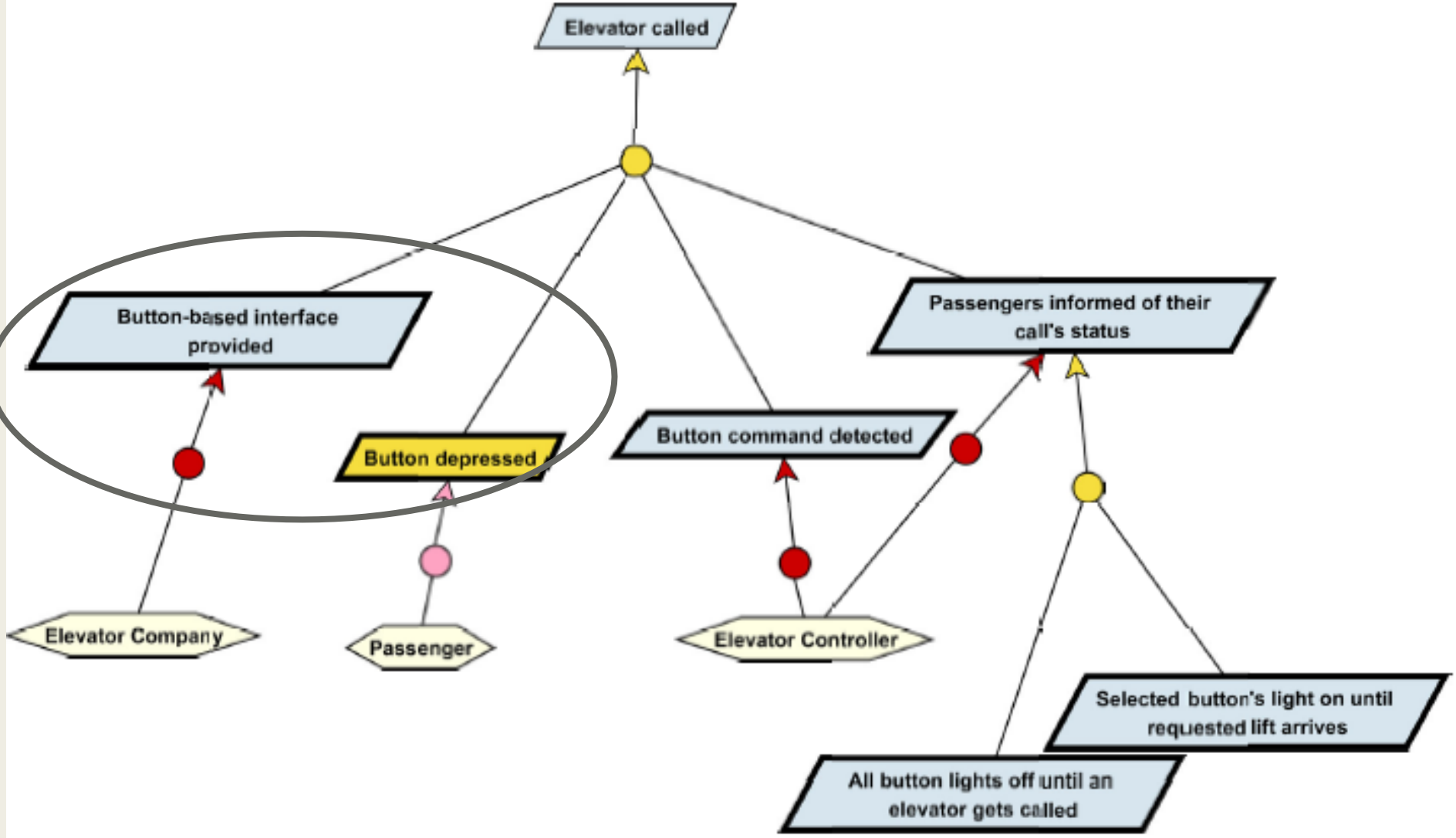



[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Zoom: 160 %
(Goal) SafetyGoals Alt-F6
SafeTransportation
NoTrainCollision
DoorClosed WhileMoving
BlockSpeedLimited
NoTrainsOn Same Block
WorstCaseStopping DistanceMaintained
DoorsClosedIFFnonZeroSpeed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

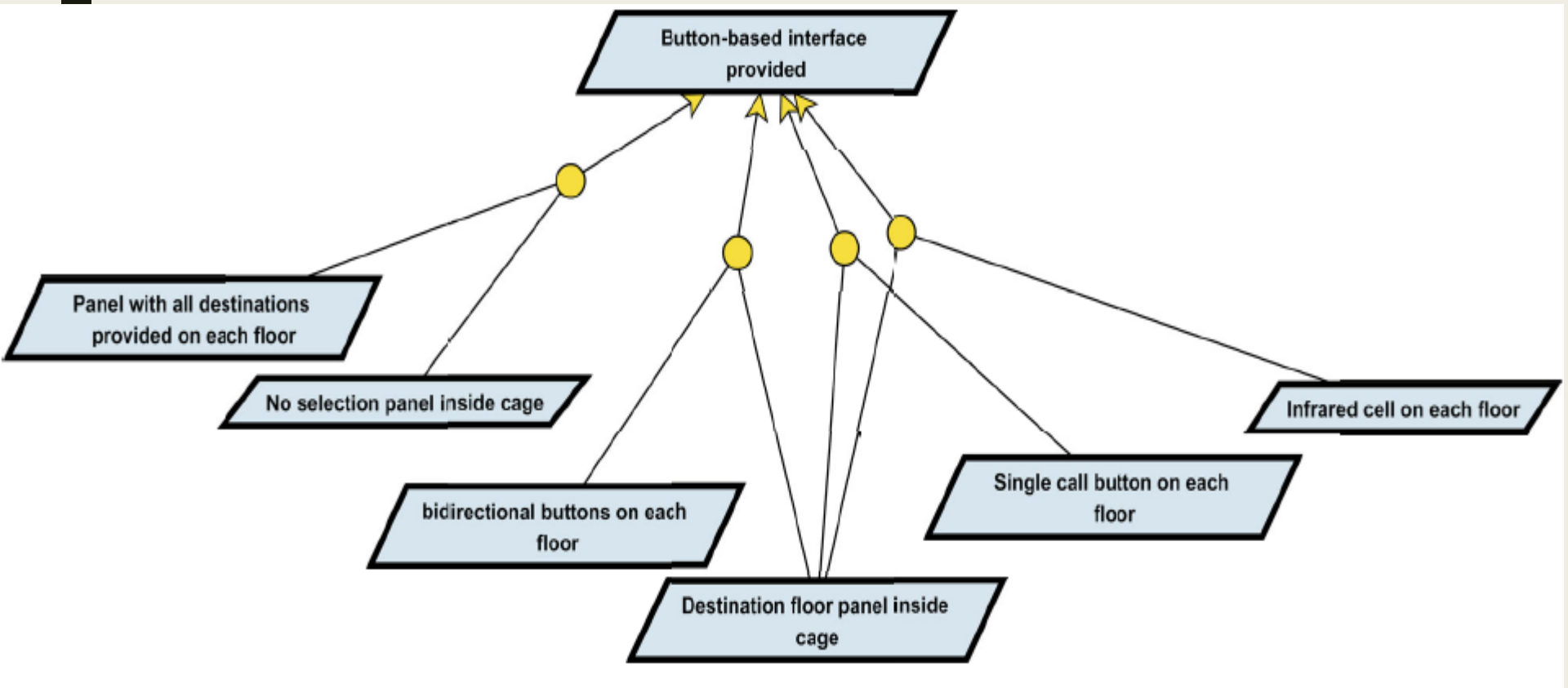
Capturing potential conflicts among goals



Remember the elevator system ...



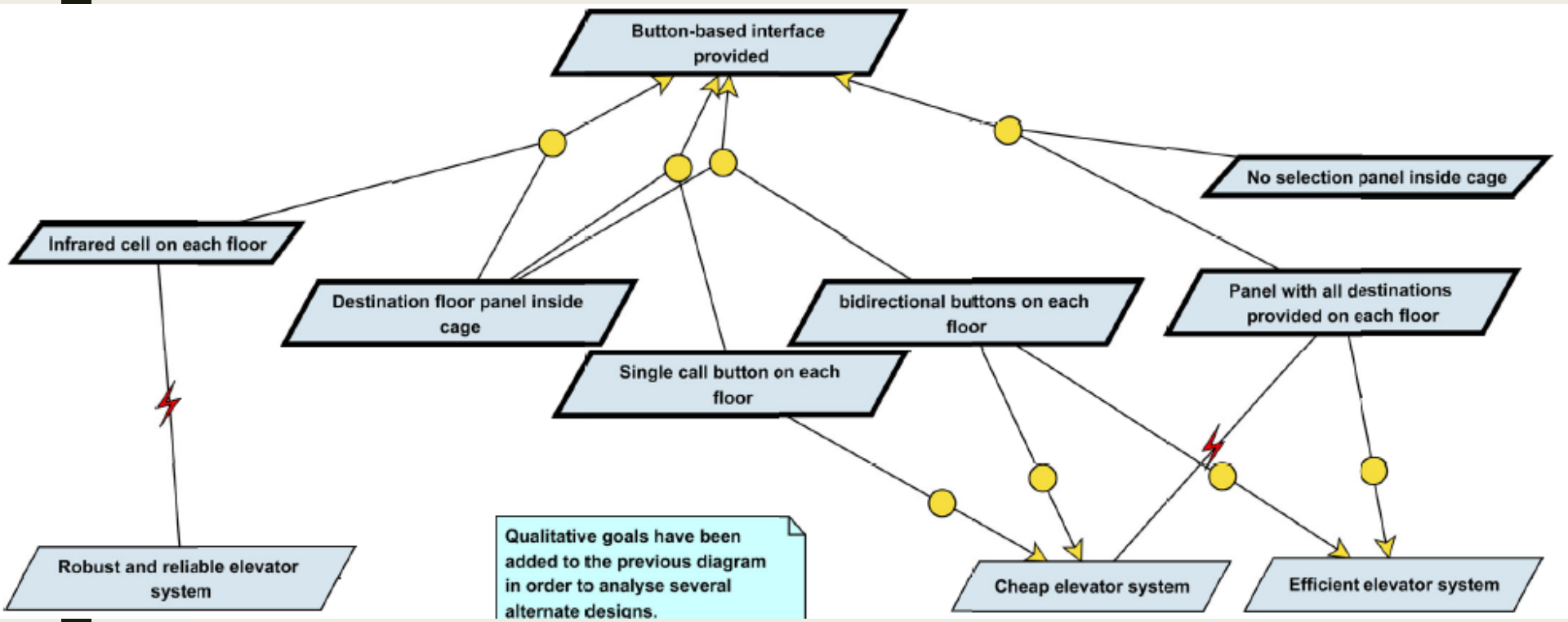
Alternatives



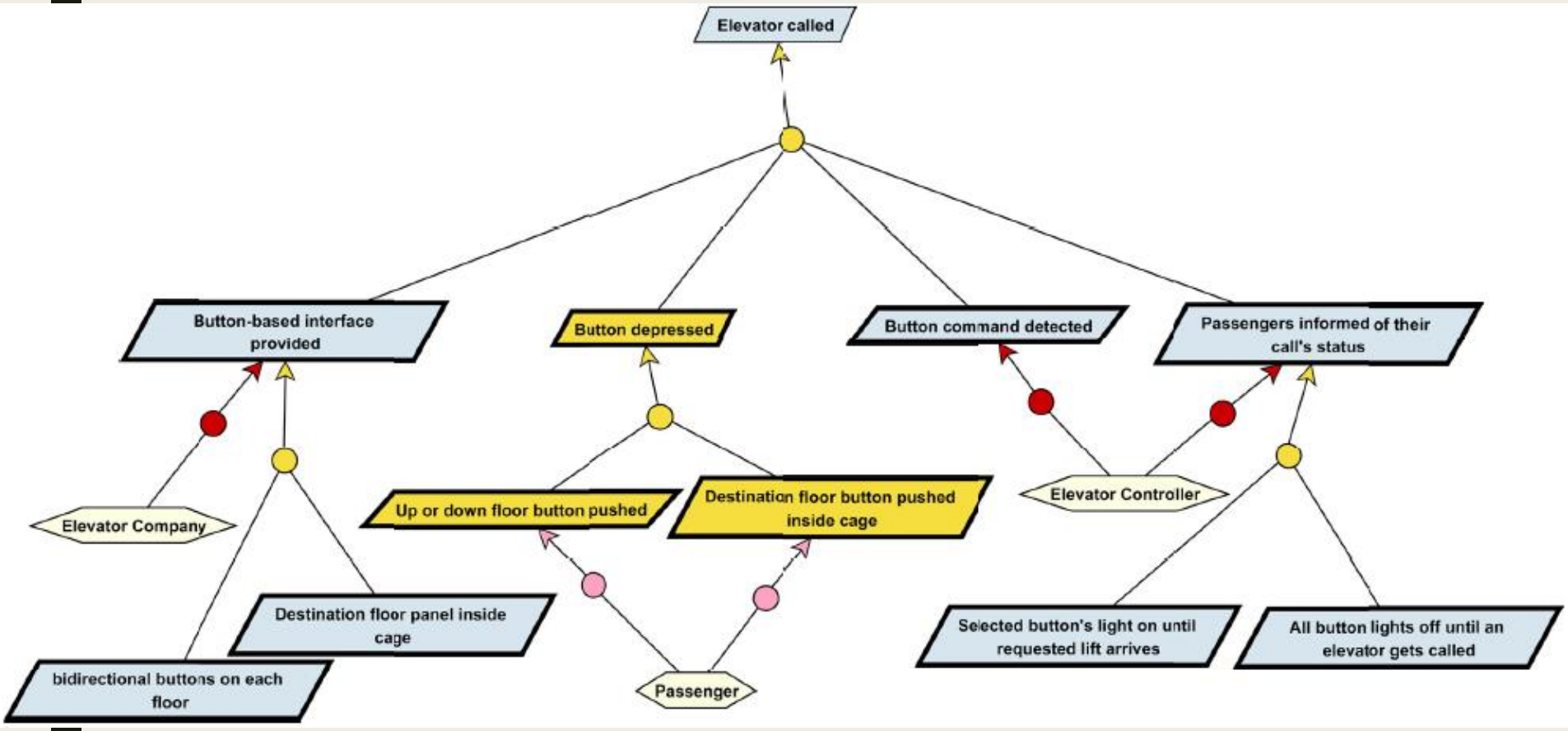
Adding Qualitative goals

- What happen if we add:
  - *Robust and reliable elevator system*
  - *Cheap elevator system*
  - *Efficient elevator system*
- Which conflicts are identified here?

Analysis of the button based interface with NF goals



Elevator called with selected alternatives



What are obstacles?

- Motivation: goals in refinement graph are often too ideal, likel
  to be violated under abnormal conditions

  (unintentional or intentional agent behaviors)

- **Obstacle** = condition on system for violation of corresponding assertion (generally a goal)
  - *{**O**, Dom }* |= *not G* *obstruction*

  e.g. G: *TrainStoppedAtBlockSignal **If** StopSignal*

  Dom: ***If** TrainStopsAtStopSignal **then** DriverResponsive*

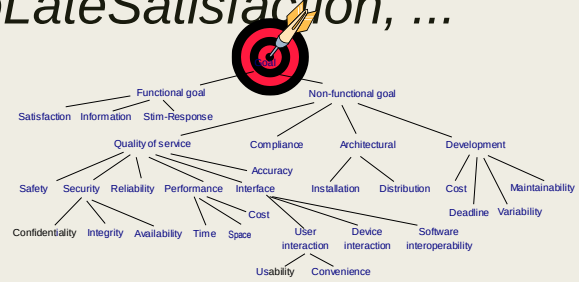
  O: *Driver**Un**responsive*

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

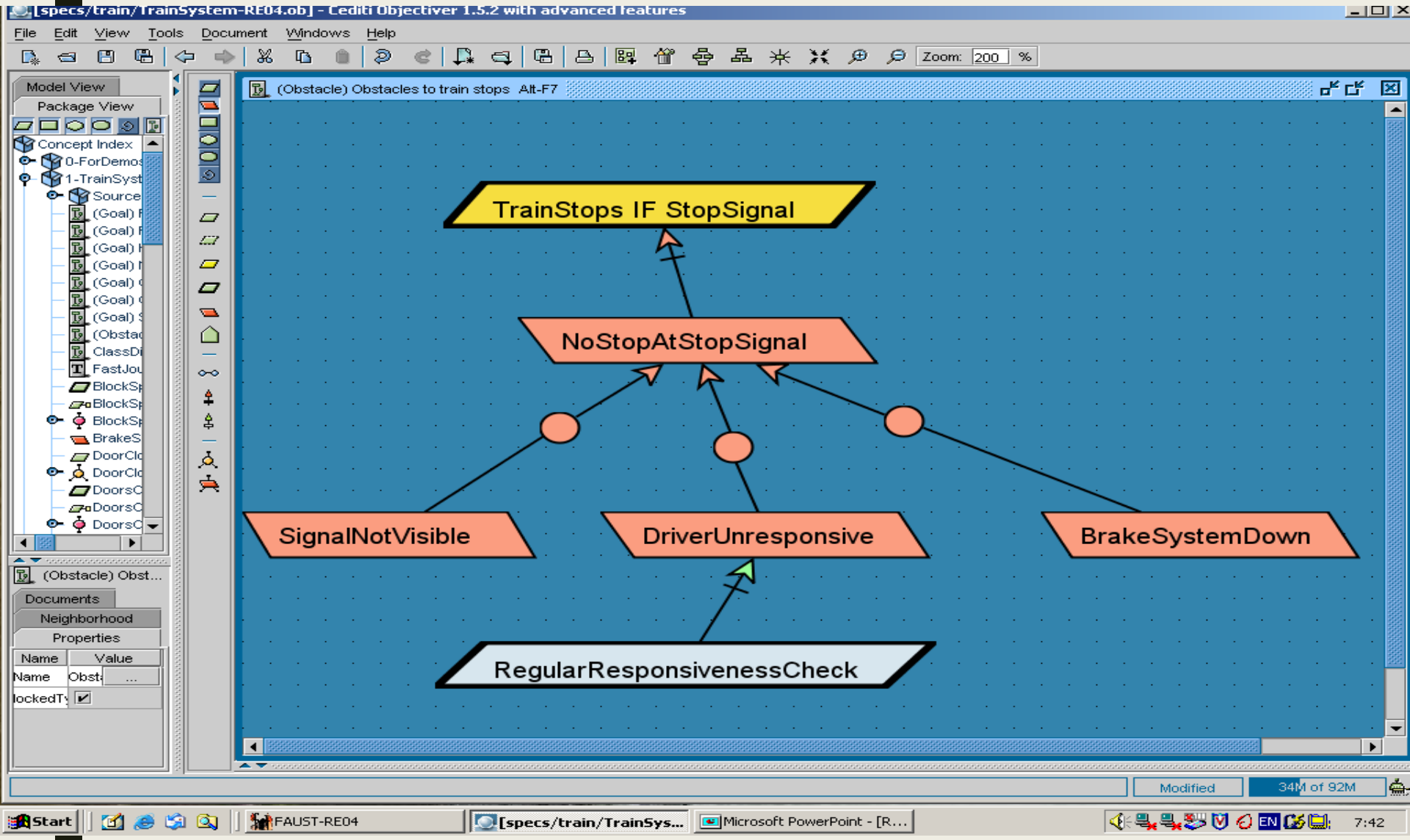
Obstacle categories for heuristic identification

Correspond to goal categories & their refinement …

- **Hazard** obstacles obstruct **Safety** goals
- **Threat** obstacles obstruct **Security** goals
  - *Disclosure, Corruption, DenialOfService, …*
- **Inaccuracy** obstacles obstruct **Accuracy** goals
- **Misinformation** obstacles obstruct **Information** goals
  - *NonInformation, WrongInformation, TooLateInformation, …*
- **Dissatisfaction** obstacles obstruct **Satisfaction** goals
  - *NonSatisfaction, PartialSatisfaction, TooLateSatisfaction, …*
- **Unusability** obstacles obstruct **Usability** goals
- …

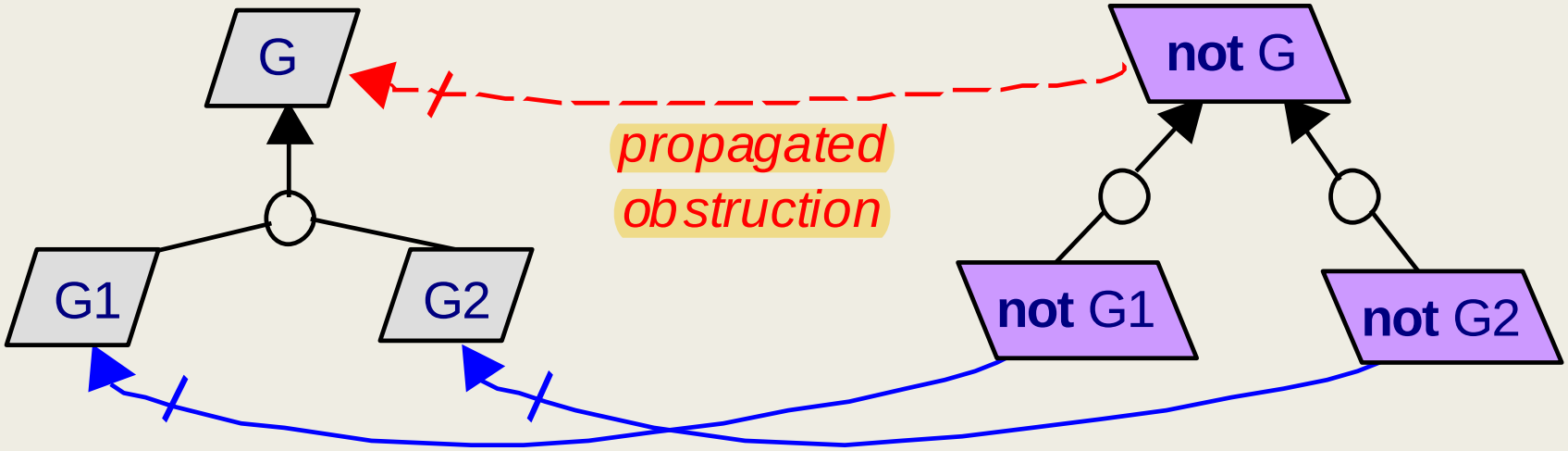


Risk analysis can be anchored on goal models



Obstructions propagate bottom-up in goal AND-refinement trees

- Cf. De Morgan's law: **not** (G1 **and** G2) equivalent to **not** G1 **or not** G2



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

Annotating obstacle diagrams

DriverUnresponsive

annotation

Obstacle DriverUnresponsive

precise definition

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

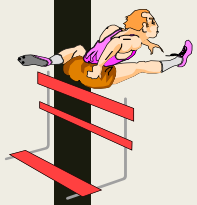
[FormalSpec ... in temporal logic for analysis, not in this chapter ...]

[Category Hazard]

[Likelihood likely]

[Criticality catastrophic]

features



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:**

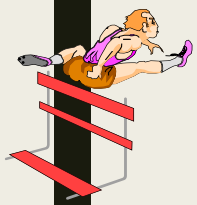
  For selected goals in the goal model …

  – *identify as many obstacles to it as possible;*

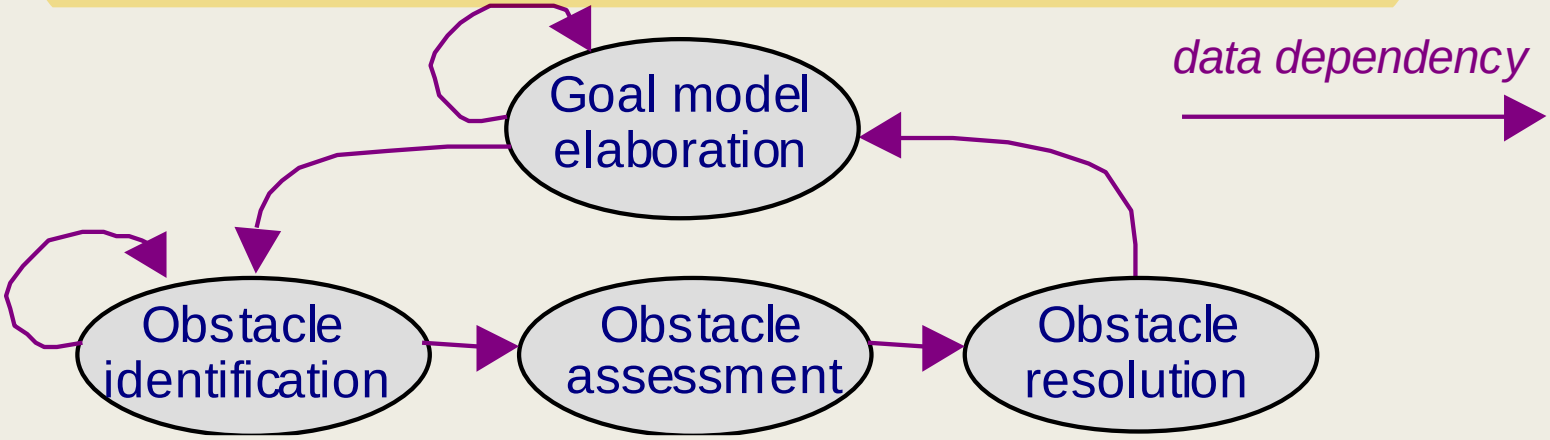
  – *assess their likelihood & severity;*

  – *resolve them according to likelihood & severity*

  => *new goals as countermeasures in the goal model*

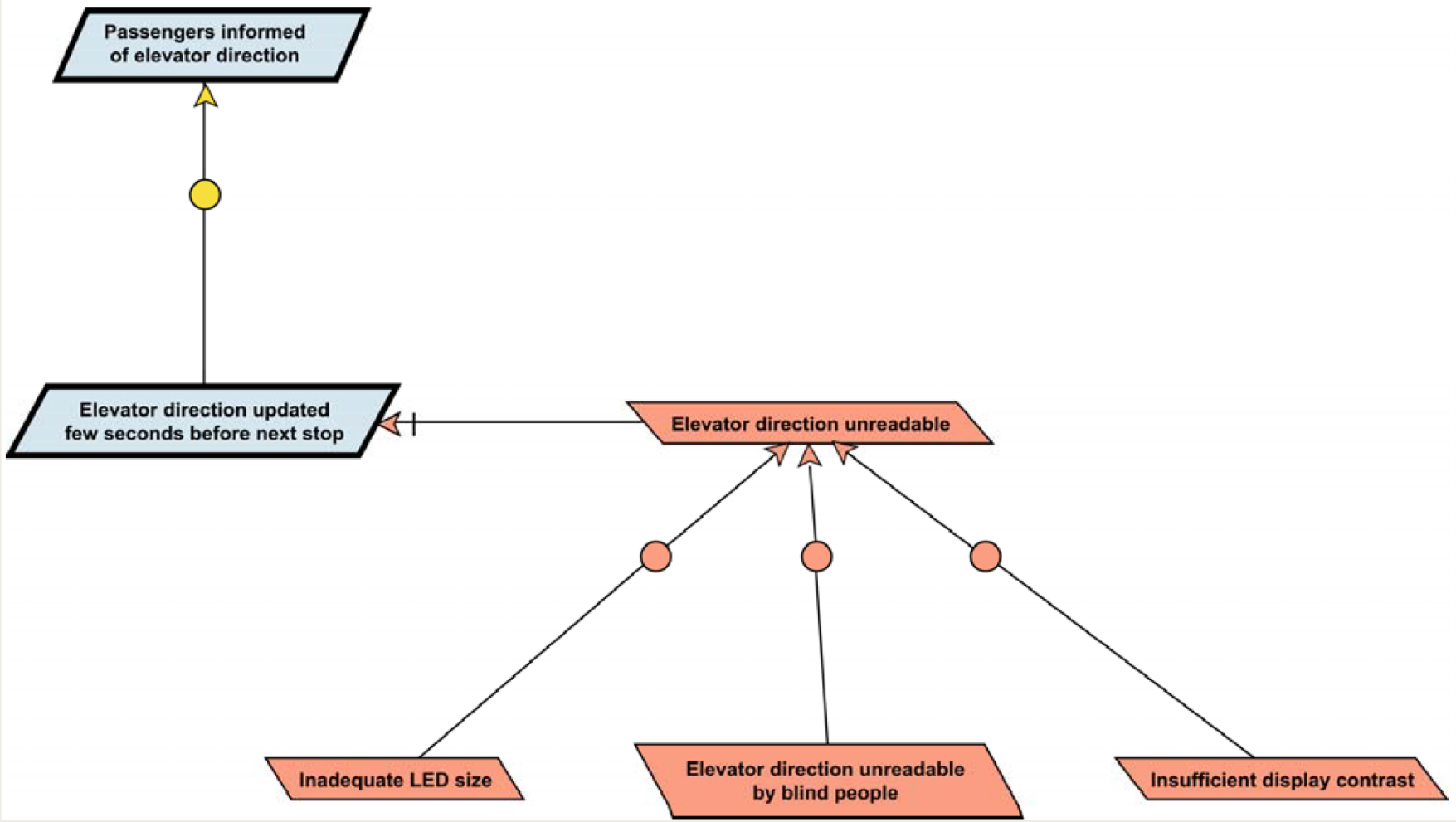


Obstacle analysis and goal model elaboration are intertwined

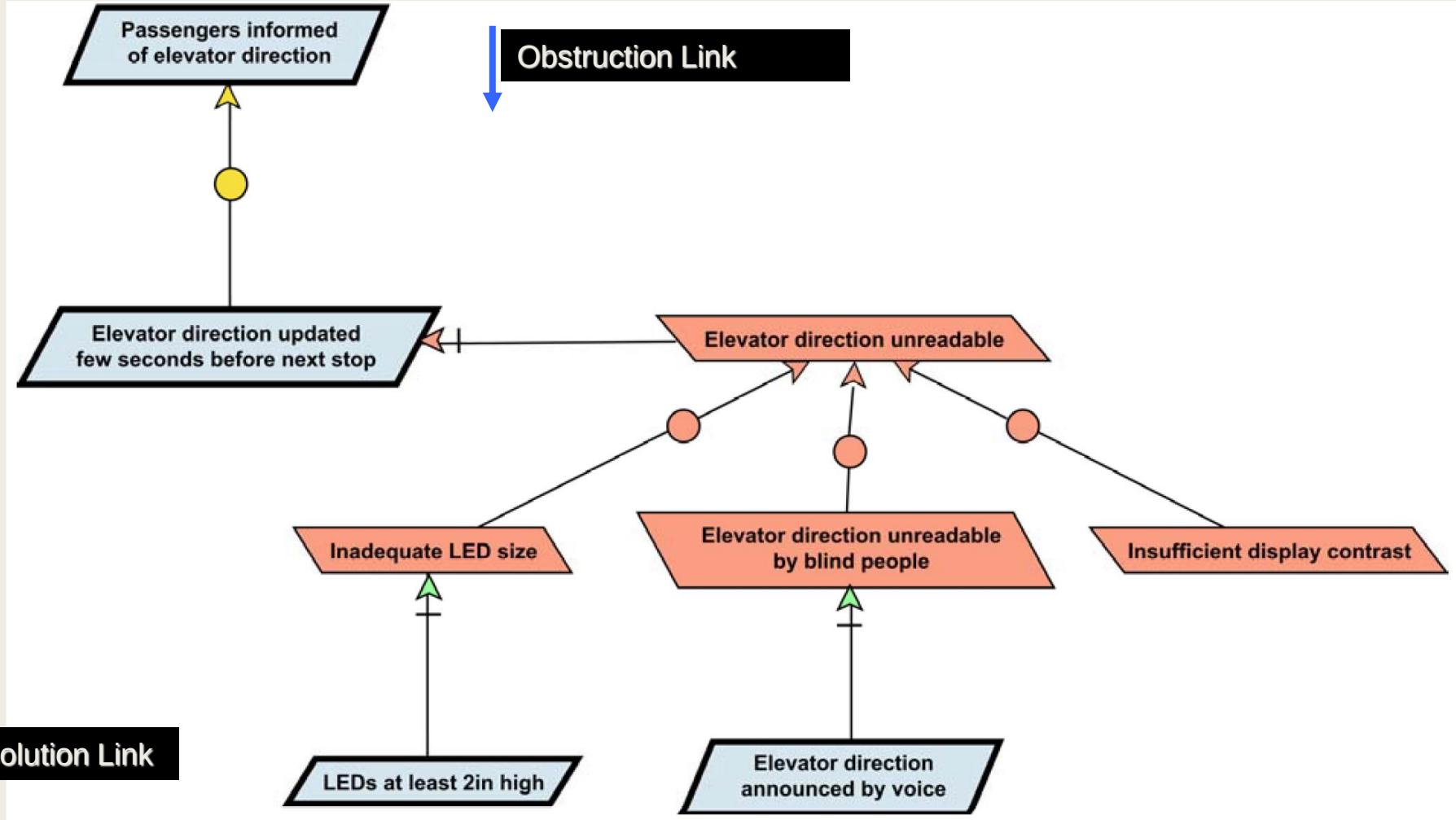


- Goal-obstacle analysis loop terminates when remaining obstacles can be tolerated
  - *unlikely or acceptable consequences*
- Which goals to consider in the goal model?
  - ***leafgoals*** *(requirements or expectations): easier to refine what is wan than what is not wanted (+ up-propagation in goal model)*
  - *based on annotated Priority & Category (Hazard, Security, …)*

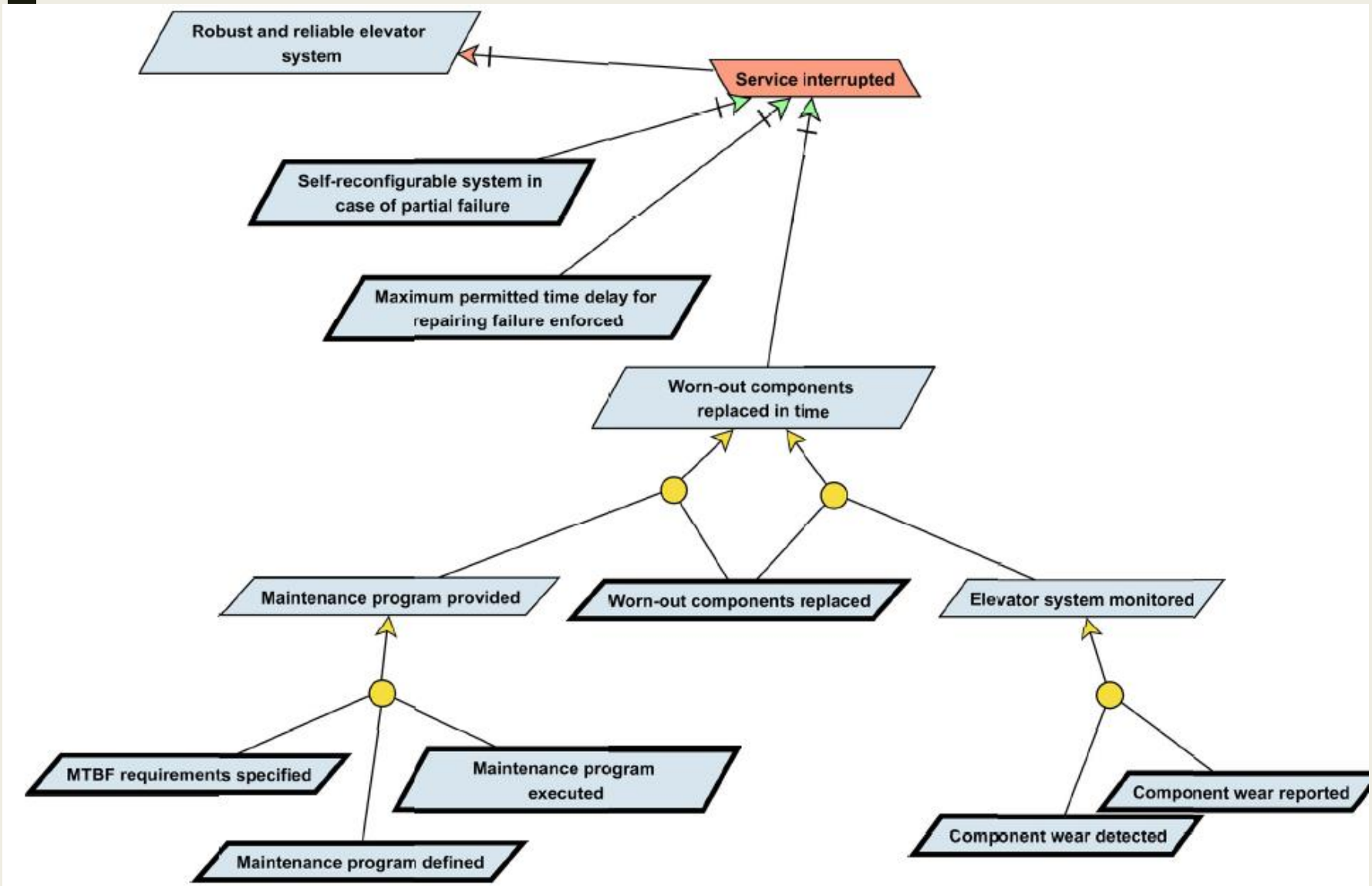
Obstacles


Passengers informed of elevator direction
Elevator direction updated few seconds before next stop
Elevator direction unreadable
Inadequate LED size
Elevator direction unreadable by blind people
Insufficient display contrast

Obstacles


Passengers informed of elevator direction
Obstruction Link
Elevator direction updated few seconds before next stop
Elevator direction unreadable
Inadequate LED size
Elevator direction unreadable by blind people
Insufficient display contrast
LEDs at least 2in high
Elevator direction announced by voice
Resolution Link

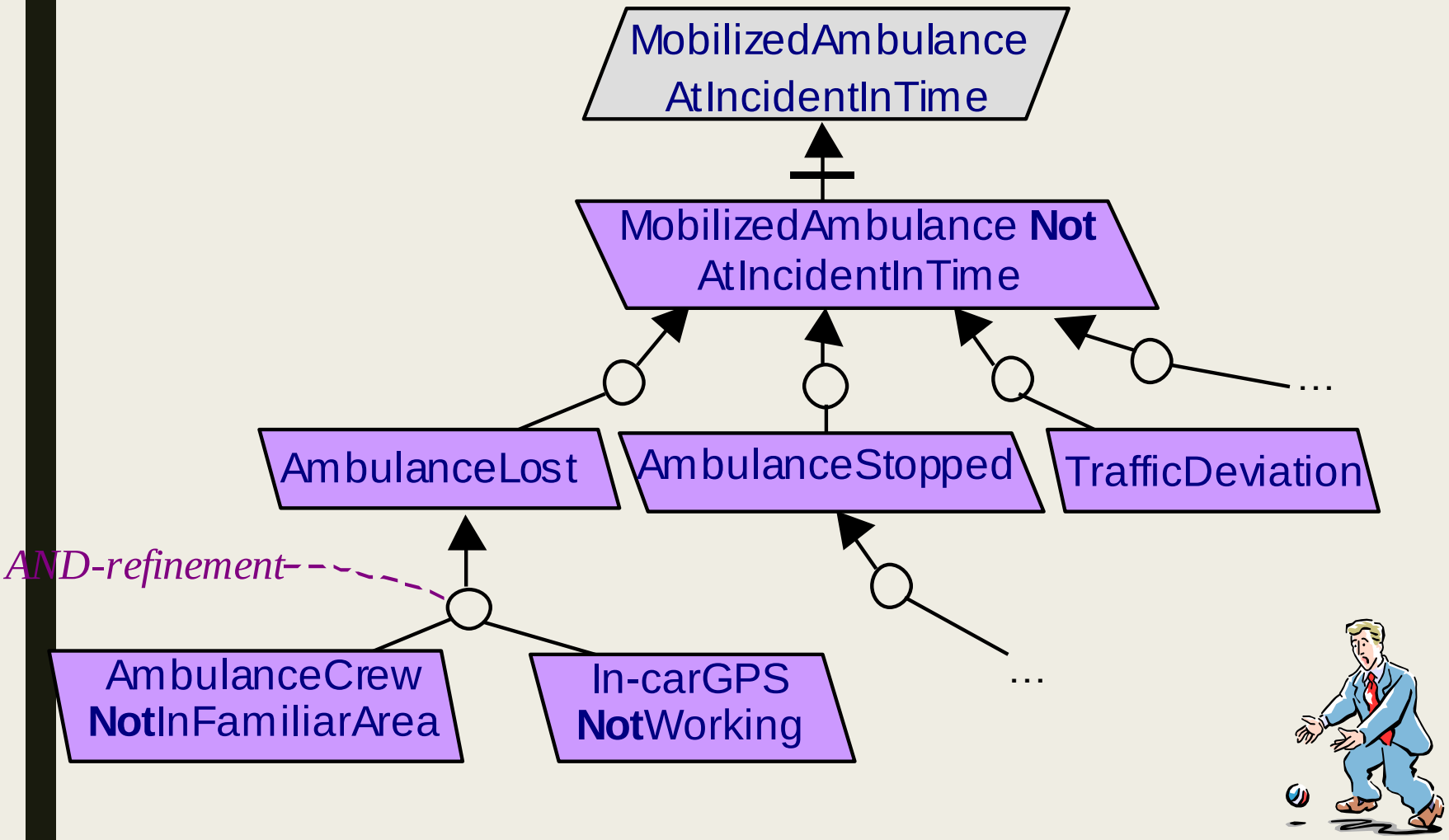
Robust and reliable elevator system



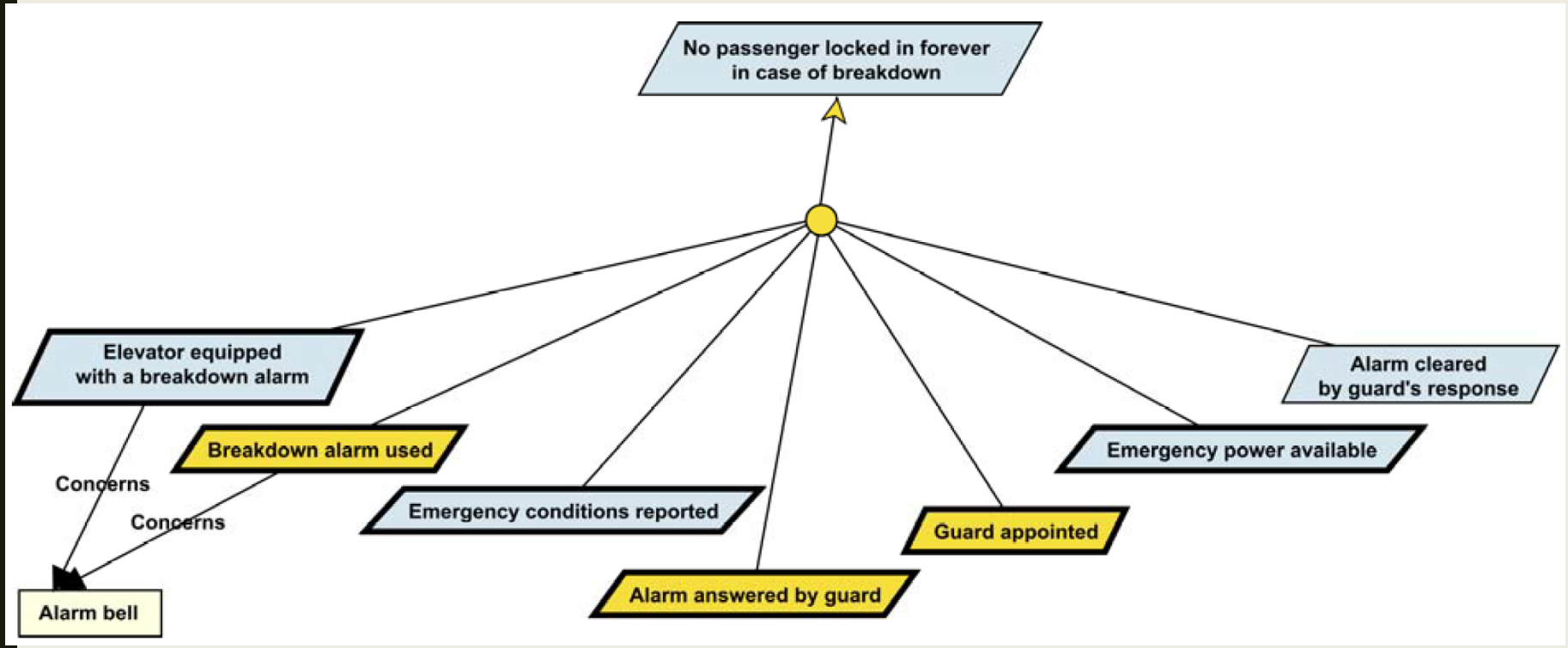
Exercise

- Mobilise Ambulance at incident in time
- Obstacle: Mobilise Ambulance NOT at incident in time

Obstacle diagrams as AND/OR refinement trees (2)



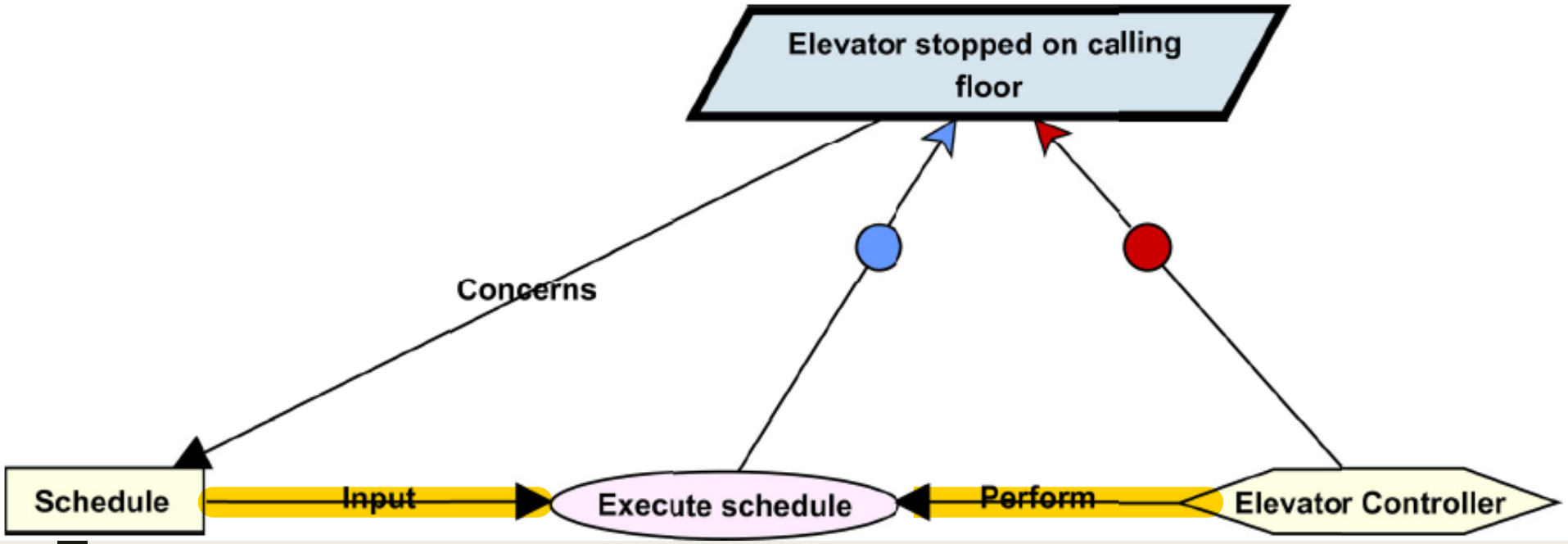
Objects


No passenger locked in forever in case of breakdown
Elevator equipped with a breakdown alarm
Breakdown alarm used
Concerns
Concerns
Alarm bell
Emergency conditions reported
Alarm answered by guard
Guard appointed
Emergency power available
Alarm cleared by guard's response

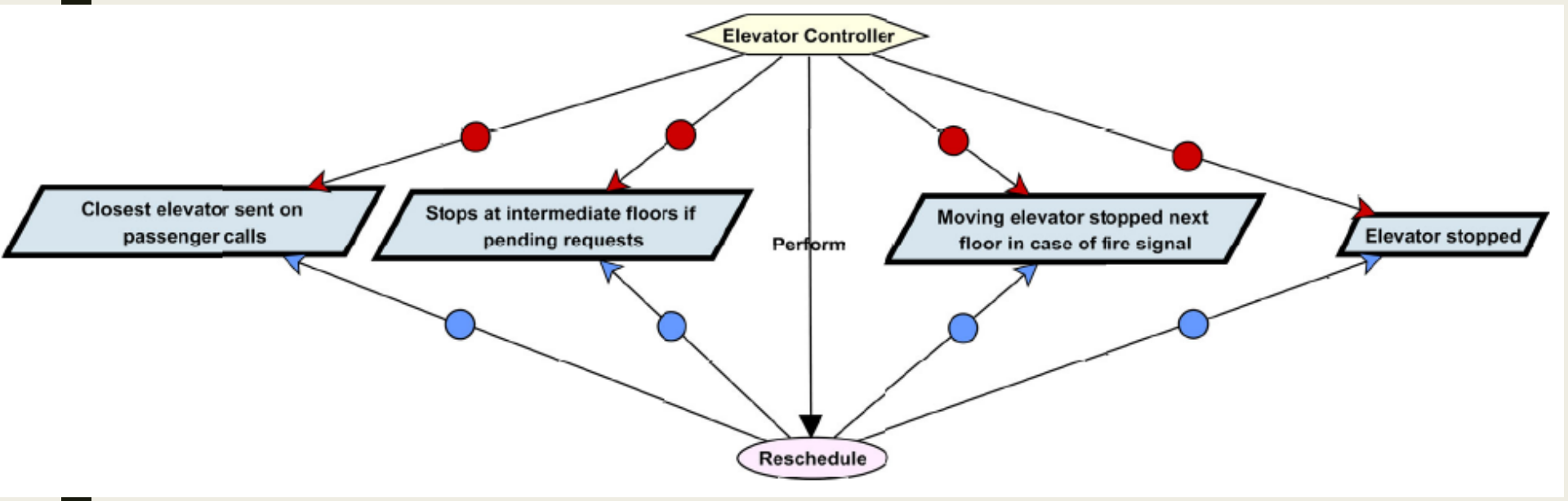
Operation Model (simplified)

- This model describes all teh behaviors that agents need to fulfill their reqs.
- Behaviors are expressed in terms of operations performed by the agents
- Operations work on objects defined in the object model
- Operations are used to operationalize or fulfill reqs

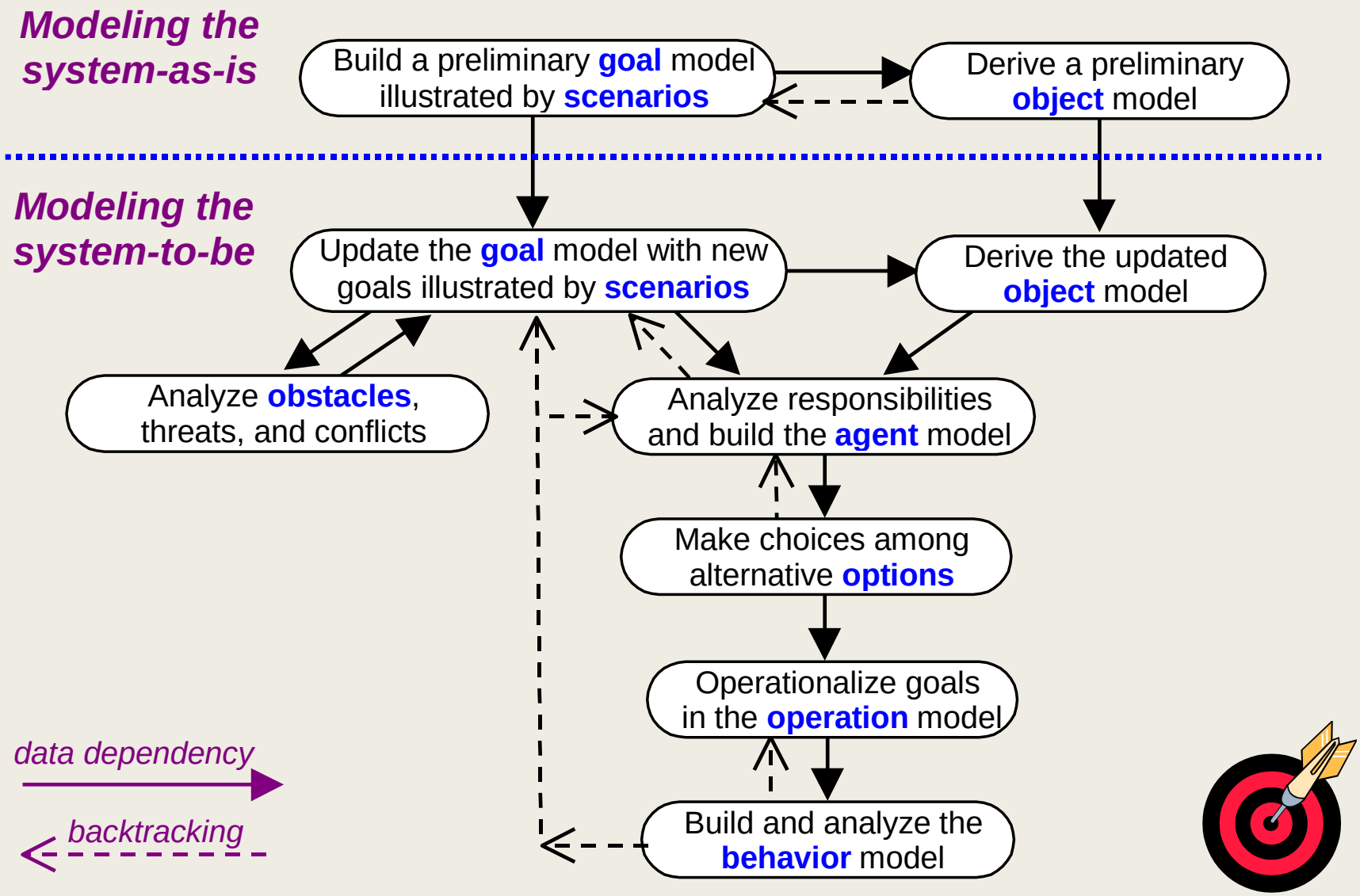
Responsibility-Operationalization-performance



Operationalization Reschedule

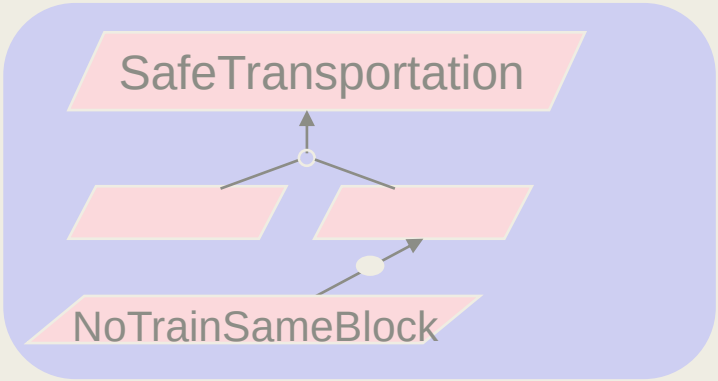


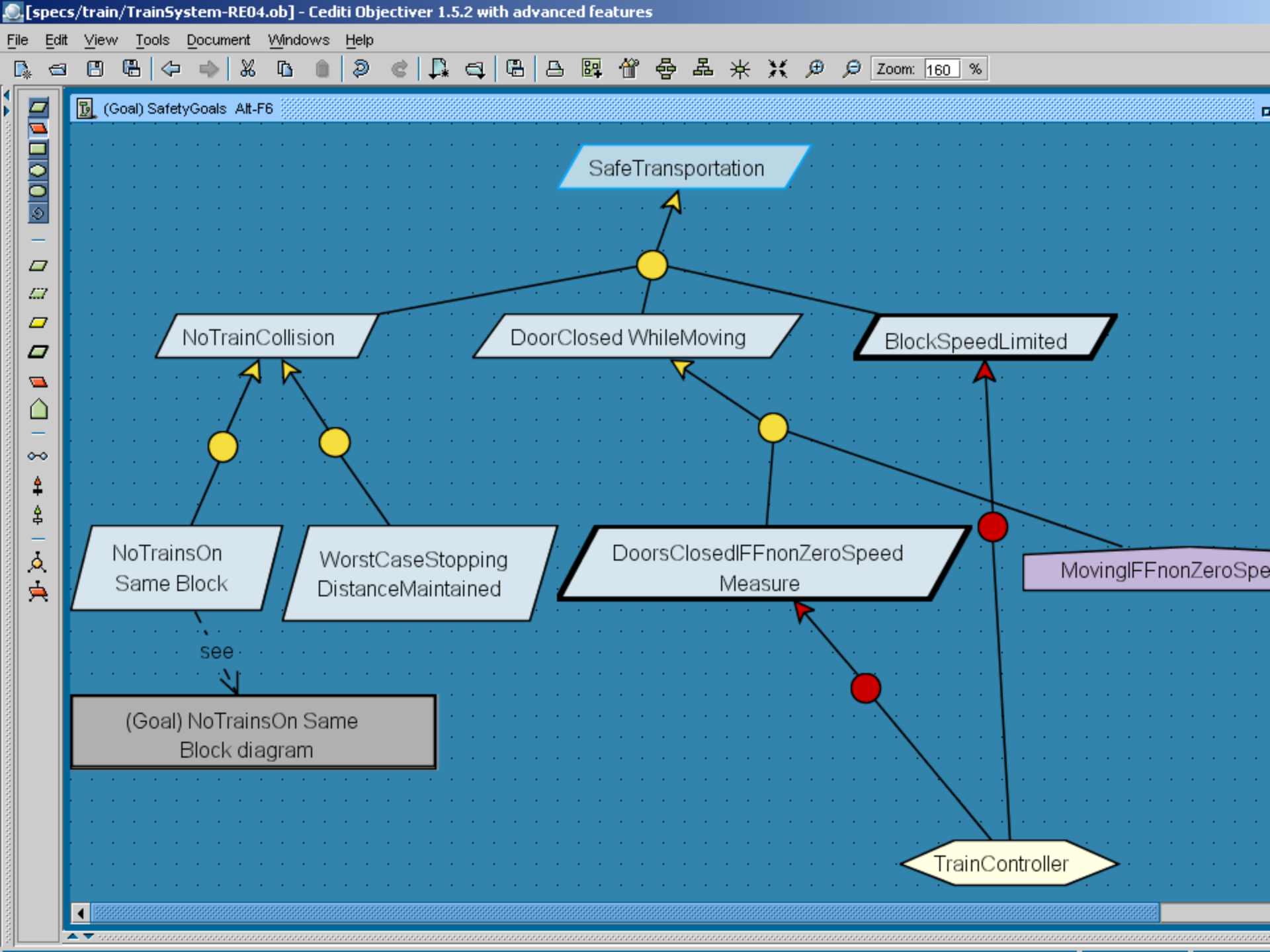
A goal-oriented method for building multi-view models

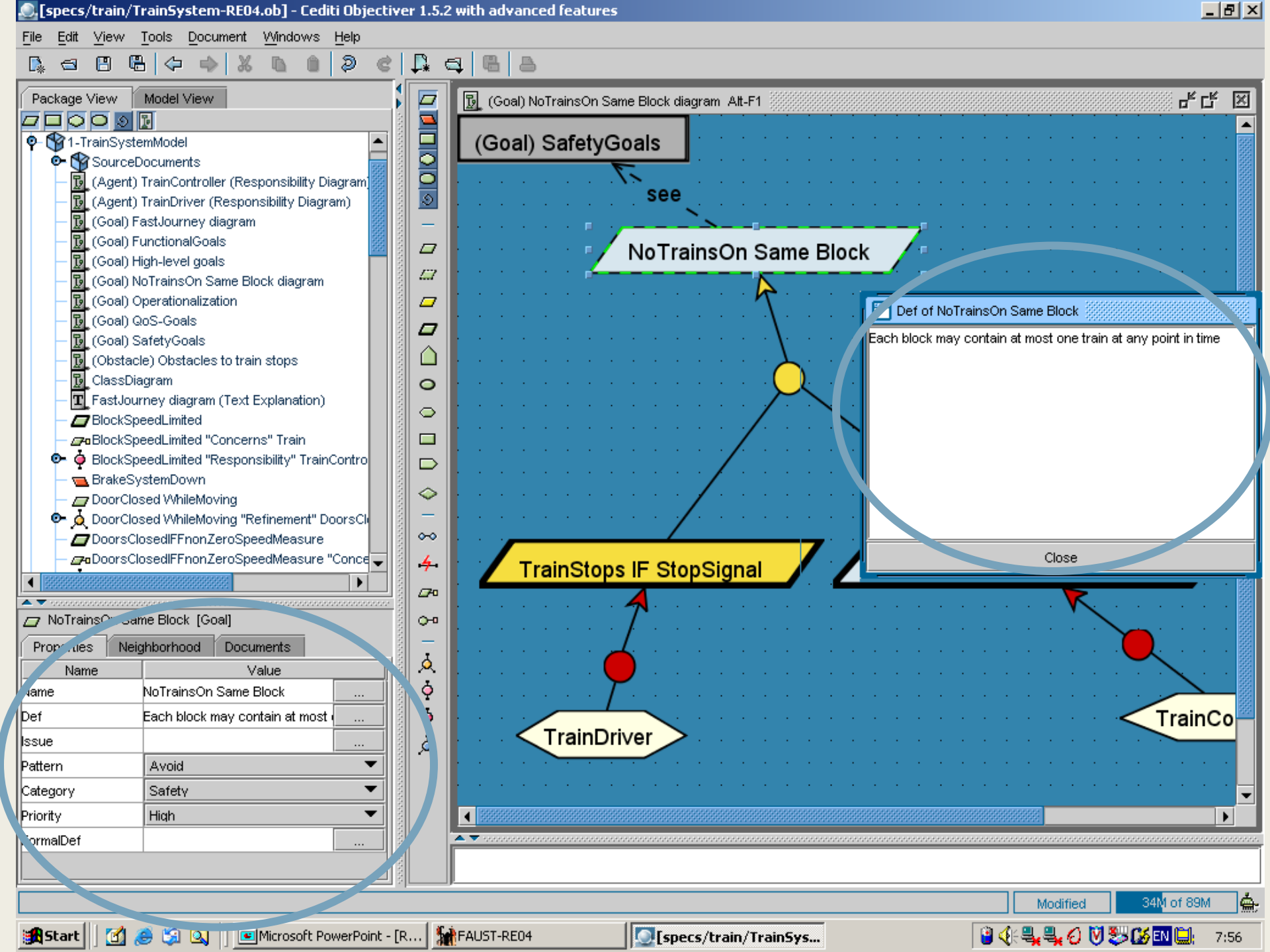


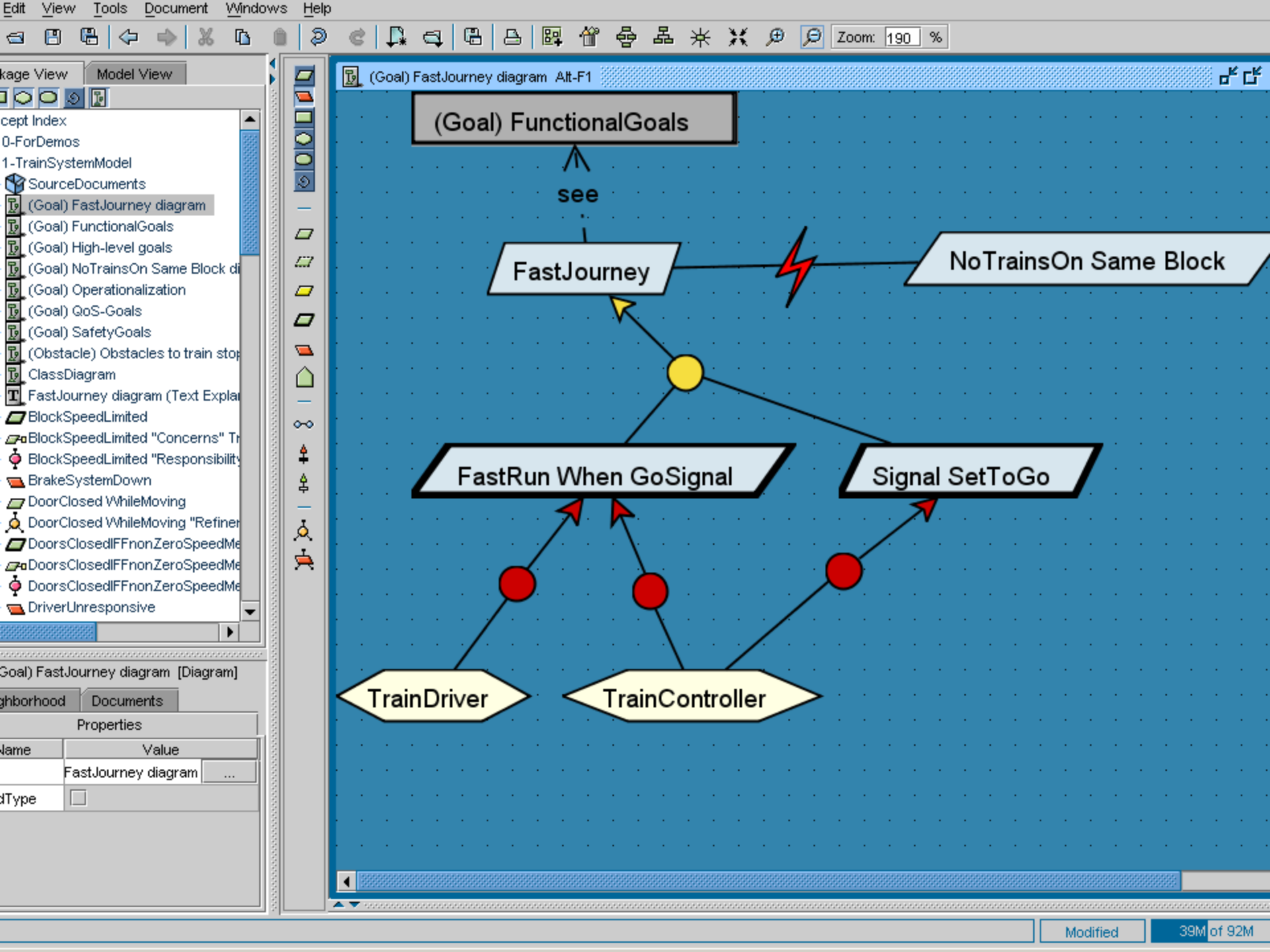
Goal-oriented model building in action

1. Domain analysis:
   refine/abstract goals



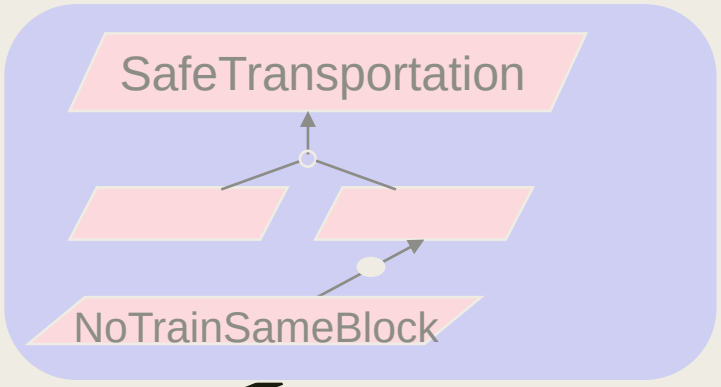

[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Zoom: 160 %
(Goal) SafetyGoals Alt-F6
SafeTransportation
NoTrainCollision
DoorClosed WhileMoving
BlockSpeedLimited
NoTrainsOn Same Block
WorstCaseStopping DistanceMaintained
DoorsClosedIFFnonZeroSpeed Measure
MovingIFFnonZeroSpe
see
(Goal) NoTrainsOn Same Block diagram
TrainController


[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Package View
Model View
1-TrainSystemModel
SourceDocuments
(Agent) TrainController (Responsibility Diagram)
(Agent) TrainDriver (Responsibility Diagram)
(Goal) FastJourney diagram
(Goal) FunctionalGoals
(Goal) High-level goals
(Goal) NoTrainsOn Same Block diagram
(Goal) Operationalization
(Goal) QoS-Goals
(Goal) SafetyGoals
(Obstacle) Obstacles to train stops
ClassDiagram
FastJourney diagram (Text Explanation)
BlockSpeedLimited
BlockSpeedLimited "Concerns" Train
BlockSpeedLimited "Responsibility" TrainContro
BrakeSystemDown
DoorClosed WhileMoving
DoorClosed WhileMoving "Refinement" DoorsCl
DoorsClosedIFFnonZeroSpeedMeasure
DoorsClosedIFFnonZeroSpeedMeasure "Conce
Same Block [Goal]
Properties
Neighborhood
Documents
Name
Value
Name
NoTrainsOn Same Block
Def
Each block may contain at most
Issue
Pattern
Avoid
Category
Safety
Priority
High
FormalDef
(Goal) NoTrainsOn Same Block diagram Alt-F1
(Goal) SafetyGoals
see
NoTrainsOn Same Block
Def of NoTrainsOn Same Block
Each block may contain at most one train at any point in time
Close
TrainStops IF StopSignal
TrainDriver
TrainCo
Modified
34M of 89M
Start
Microsoft PowerPoint - [R...
FAUST-RE04
[specs/train/TrainSys...
EN
7:56

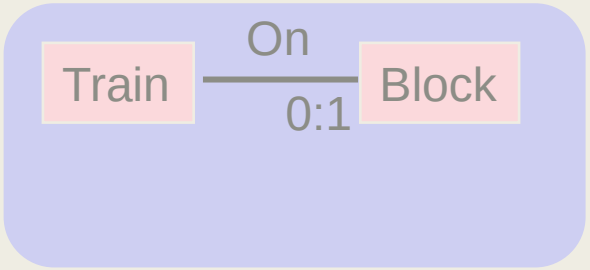

Edit View Tools Document Windows Help
Zoom: 190 %
kage View
Model View
cept Index
0-ForDemos
1-TrainSystemModel
SourceDocuments
(Goal) FastJourney diagram
(Goal) FunctionalGoals
(Goal) High-level goals
(Goal) NoTrainsOn Same Block di
(Goal) Operationalization
(Goal) QoS-Goals
(Goal) SafetyGoals
(Obstacle) Obstacles to train stop
ClassDiagram
FastJourney diagram (Text Explar
BlockSpeedLimited
BlockSpeedLimited "Concerns" Tr
BlockSpeedLimited "Responsibility
BrakeSystemDown
DoorClosed WhileMoving
DoorClosed WhileMoving "Refiner
DoorsClosedIFFnonZeroSpeedMe
DoorsClosedIFFnonZeroSpeedMe
DoorsClosedIFFnonZeroSpeedMe
DriverUnresponsive
Goal) FastJourney diagram [Diagram]
ghborhood
Documents
Properties
Name
Value
FastJourney diagram
...
dType
(Goal) FastJourney diagram Alt-F1
(Goal) FunctionalGoals
see
FastJourney
NoTrainsOn Same Block
FastRun When GoSignal
Signal SetToGo
TrainDriver
TrainController
Modified
39M of 92M

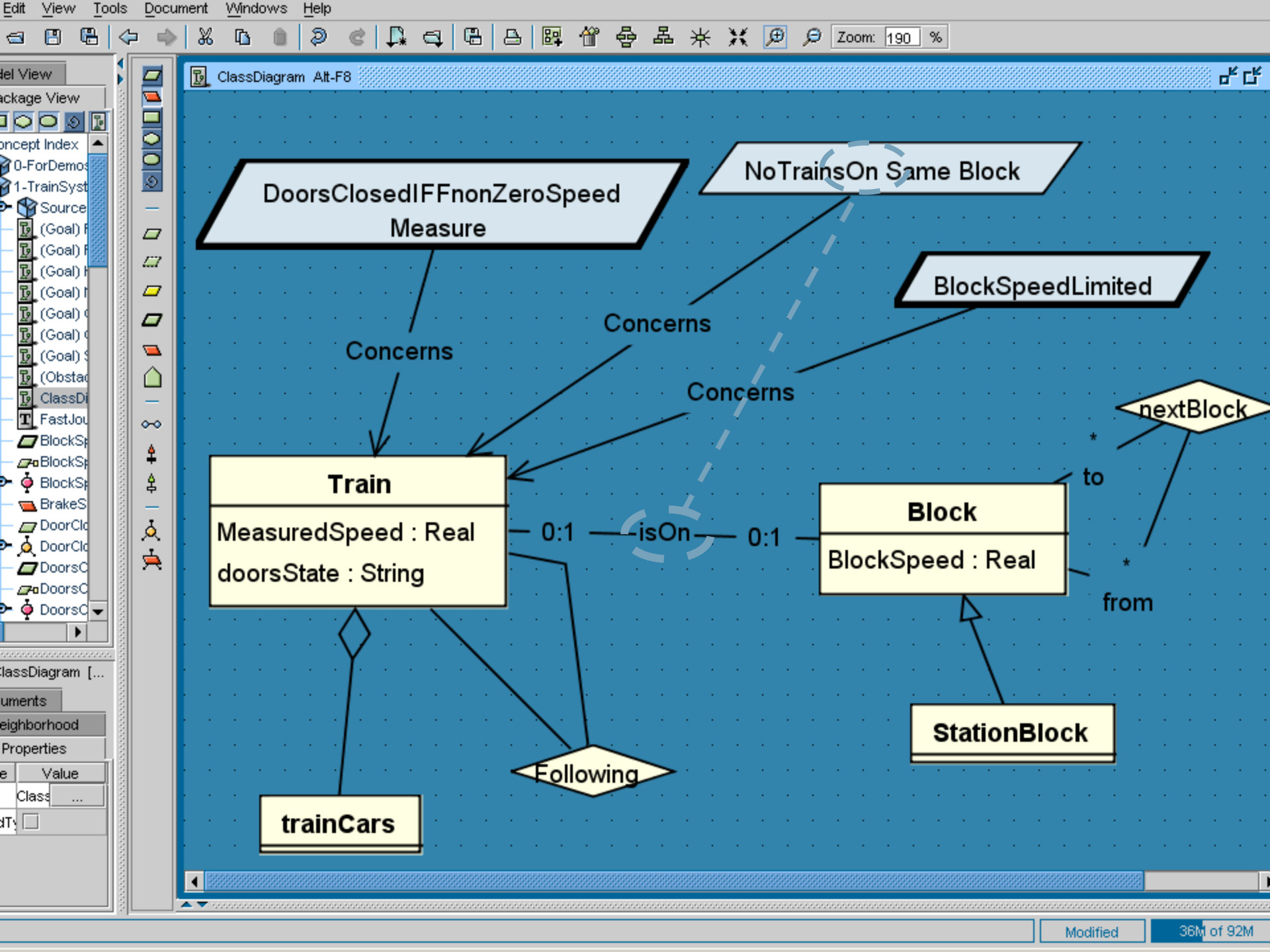
Goal-oriented model building in action

1. Domain analysis:
refine/abstract goals



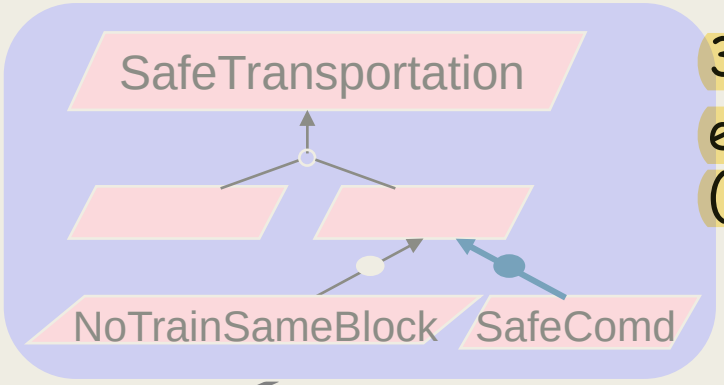
2. Domain analysis:
derive/structure
objects




Edit View Tools Document Windows Help
Zoom: 190 %
ClassDiagram Alt-F8
DoorsClosedIFFnonZeroSpeed Measure
NoTrainsOn Same Block
BlockSpeedLimited
Concerns
Concerns
Concerns
Train
MeasuredSpeed : Real
doorsState : String
0:1
isOn
0:1
Block
BlockSpeed : Real
nextBlock
*
to
*
from
StationBlock
Following
trainCars
Modified
38M of 92M

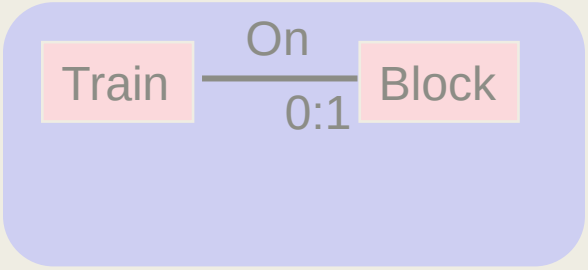
Goal-oriented model building in action

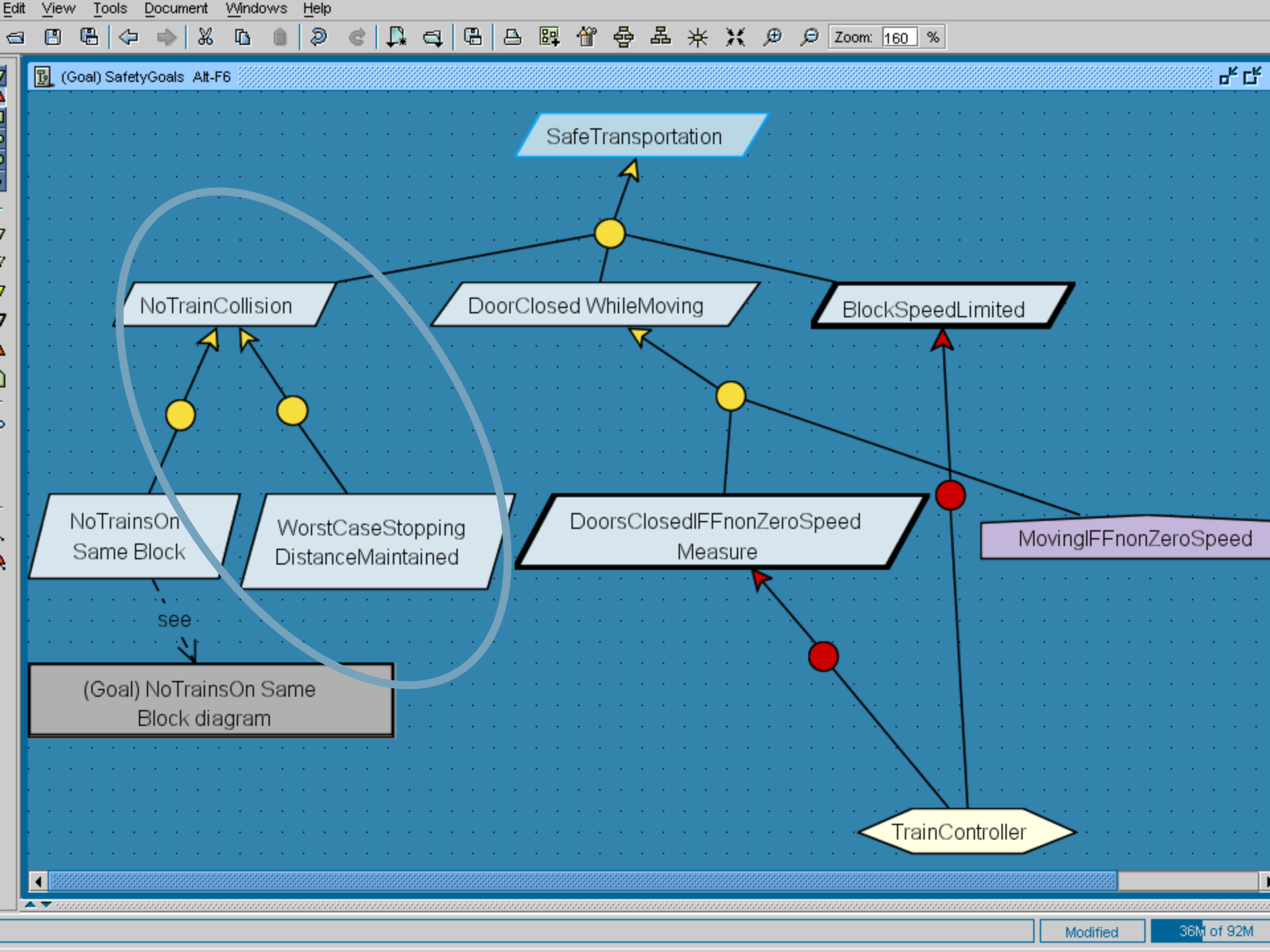
1. Domain analysis: refine/abstract goals



3. analysis: enriched goals (alternatives)

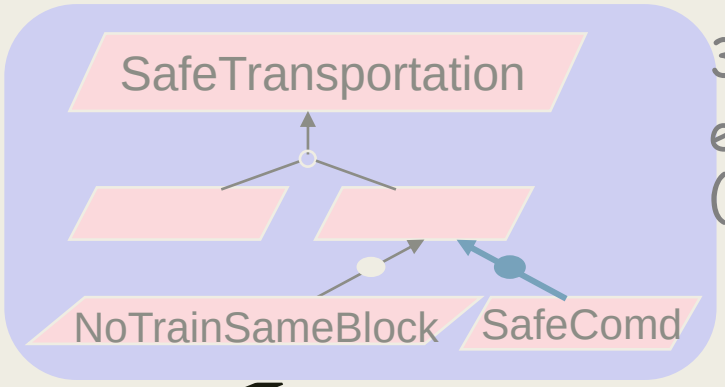
2. Domain analysis: derive/structure objects




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

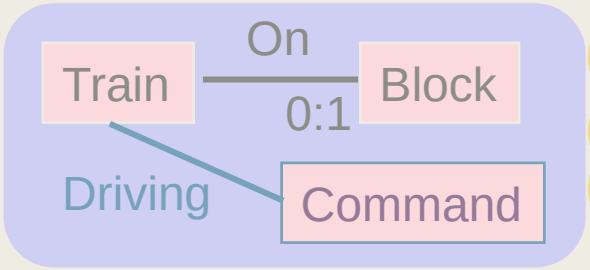
Goal-oriented model building in action

1. Domain analysis:
refine/abstract goals



3. analysis:
enriched goals
(alternatives)

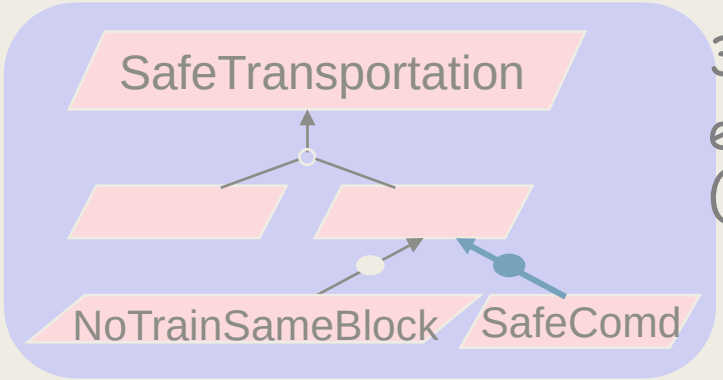
2. Domain analysis:
derive/structure
objects



4. analysis:
enriched objects
from new goals

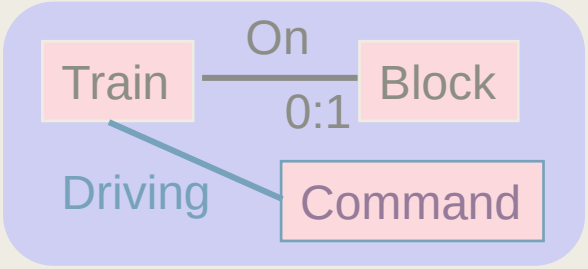
Goal-oriented model building in action

1. Domain analysis:
refine/abstract goals

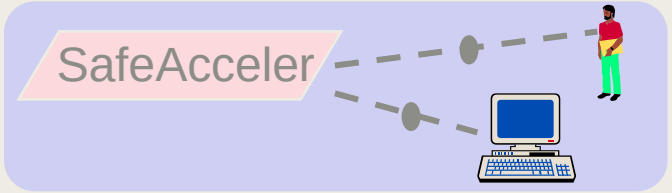


3. analysis:
enriched goals
(alternatives)

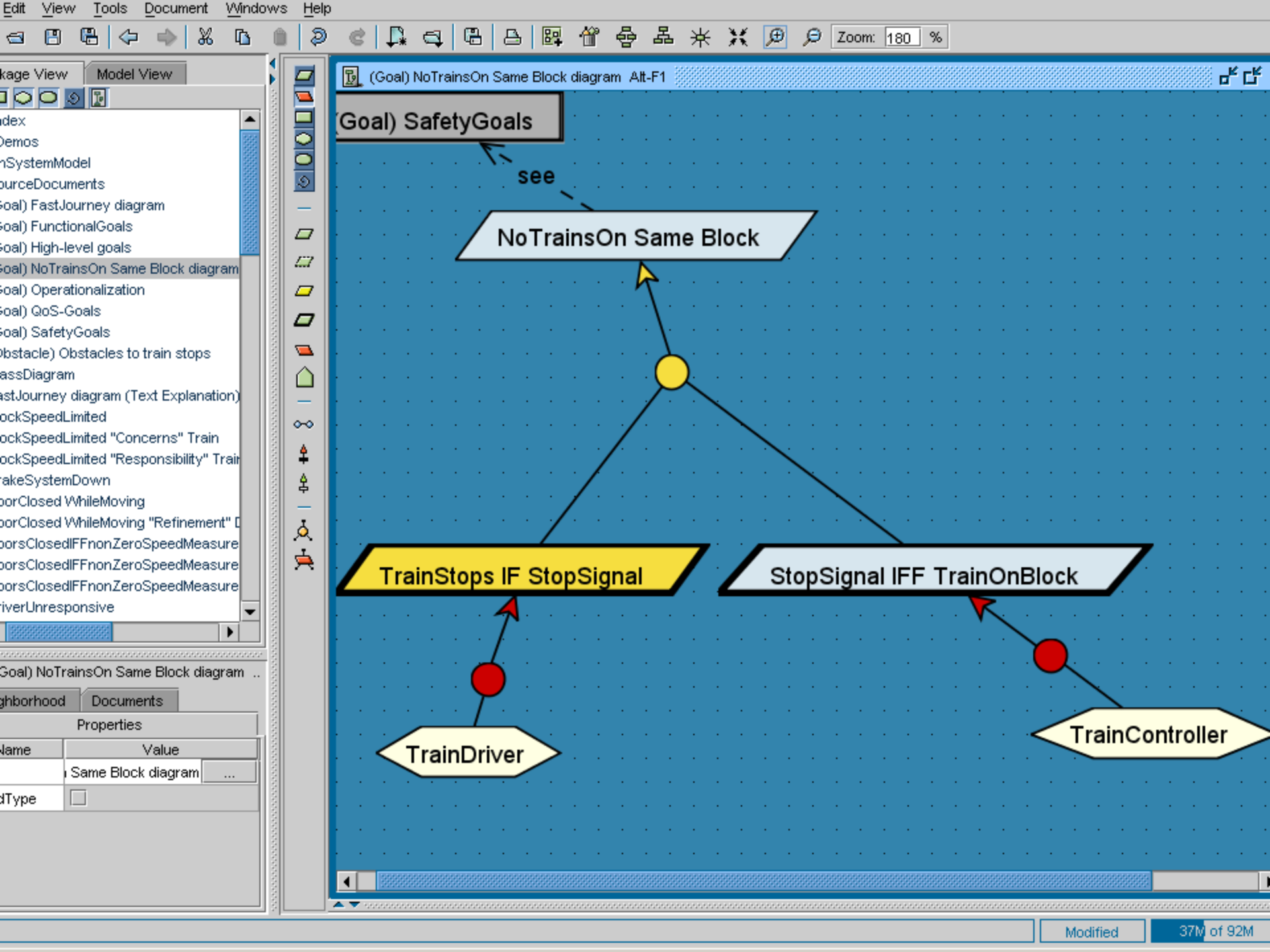
2. Domain analysis:
derive/structure
objects

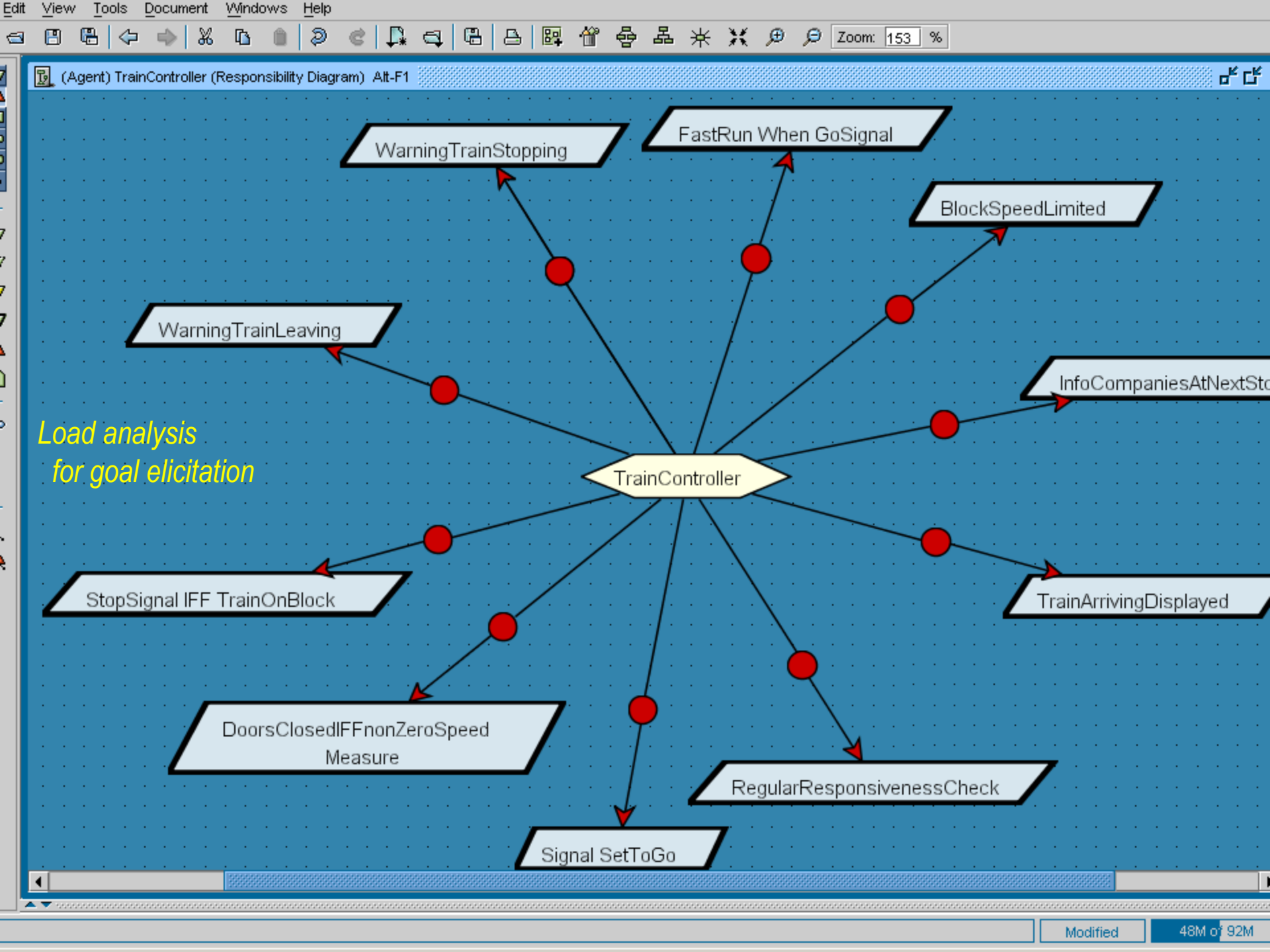


4. analysis:
enriched objects
from new goals



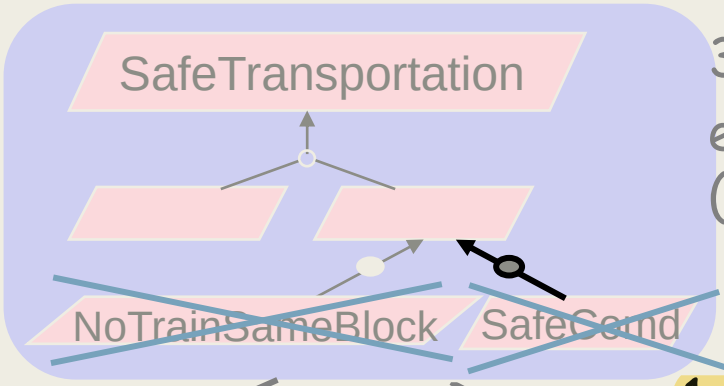
5. Responsibility analysis:
agent OR-assignment


Edit View Tools Document Windows Help
Zoom: 180 %
kage View
Model View
(Goal) NoTrainsOn Same Block diagram Alt-F1
(Goal) SafetyGoals
see
NoTrainsOn Same Block
TrainStops IF StopSignal
StopSignal IFF TrainOnBlock
TrainDriver
TrainController
(Goal) NoTrainsOn Same Block diagram ..
ghborhood
Documents
Properties
Name
Value
Same Block diagram
dType
Modified
37M of 92M


Edit View Tools Document Windows Help
Zoom: 153 %
(Agent) TrainController (Responsibility Diagram) Alt-F1
WarningTrainStopping
FastRun When GoSignal
BlockSpeedLimited
WarningTrainLeaving
InfoCompaniesAtNextSto
Load analysis
for goal elicitation
TrainController
StopSignal IFF TrainOnBlock
TrainArrivingDisplayed
DoorsClosedIFFnonZeroSpeed
Measure
RegularResponsivenessCheck
Signal SetToGo
Modified
48M of 92M

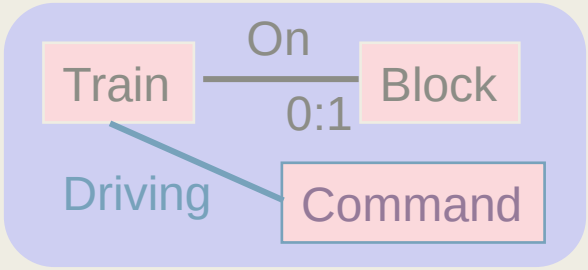
Goal-oriented model building in action

1. Domain analysis:
refine/abstract goals



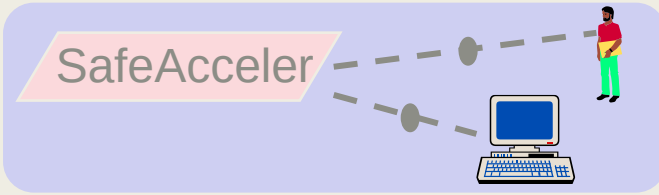
3. analysis:
enriched goals
(alternatives)

2. Domain analysis:
derive/structure
objects

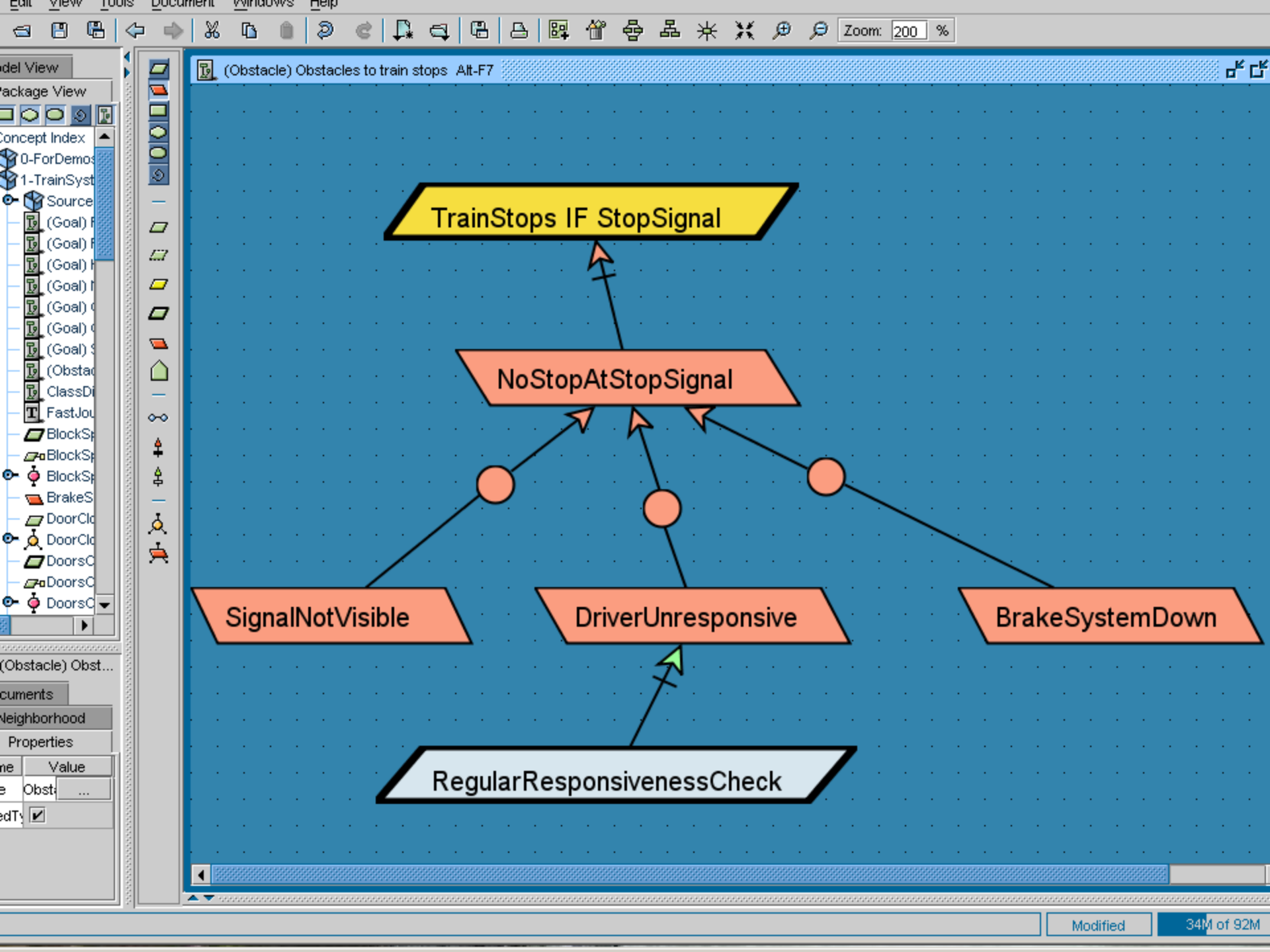


4. analysis:
enriched objects
from new goals

1-5. Obstacle & conflic
analysis

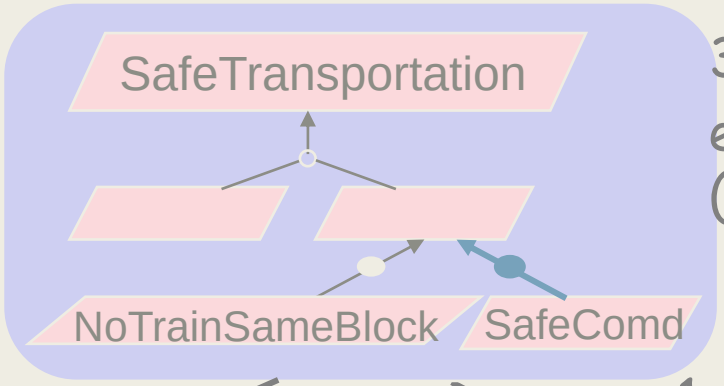


5. Responsibility analysis
agent OR-assignment


Edit View Tools Document Windows Help
Zoom: 200 %
(Obstacle) Obstacles to train stops Alt-F7
TrainStops IF StopSignal
NoStopAtStopSignal
SignalNotVisible
DriverUnresponsive
BrakeSystemDown
RegularResponsivenessCheck
Concept Index
Properties
Value
Modified
34M of 92M

Goal-oriented model building in action

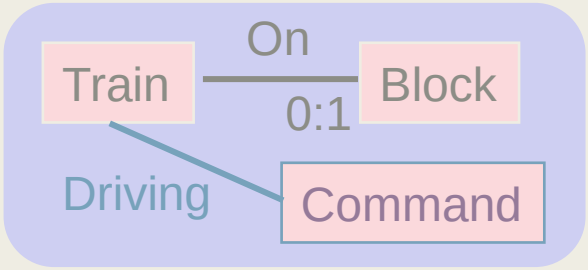
1. Domain analysis:
refine/abstract goals



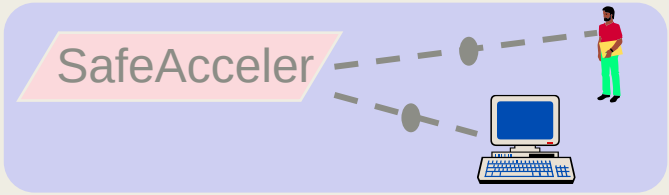
3. analysis:
enriched goals
(alternatives)

2. Domain analysis:
derive/structure
objects

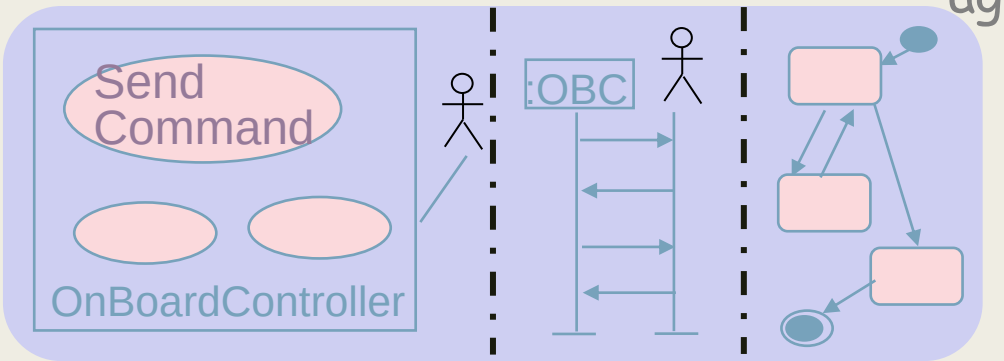
1-5. Obstacle & conflict
analysis



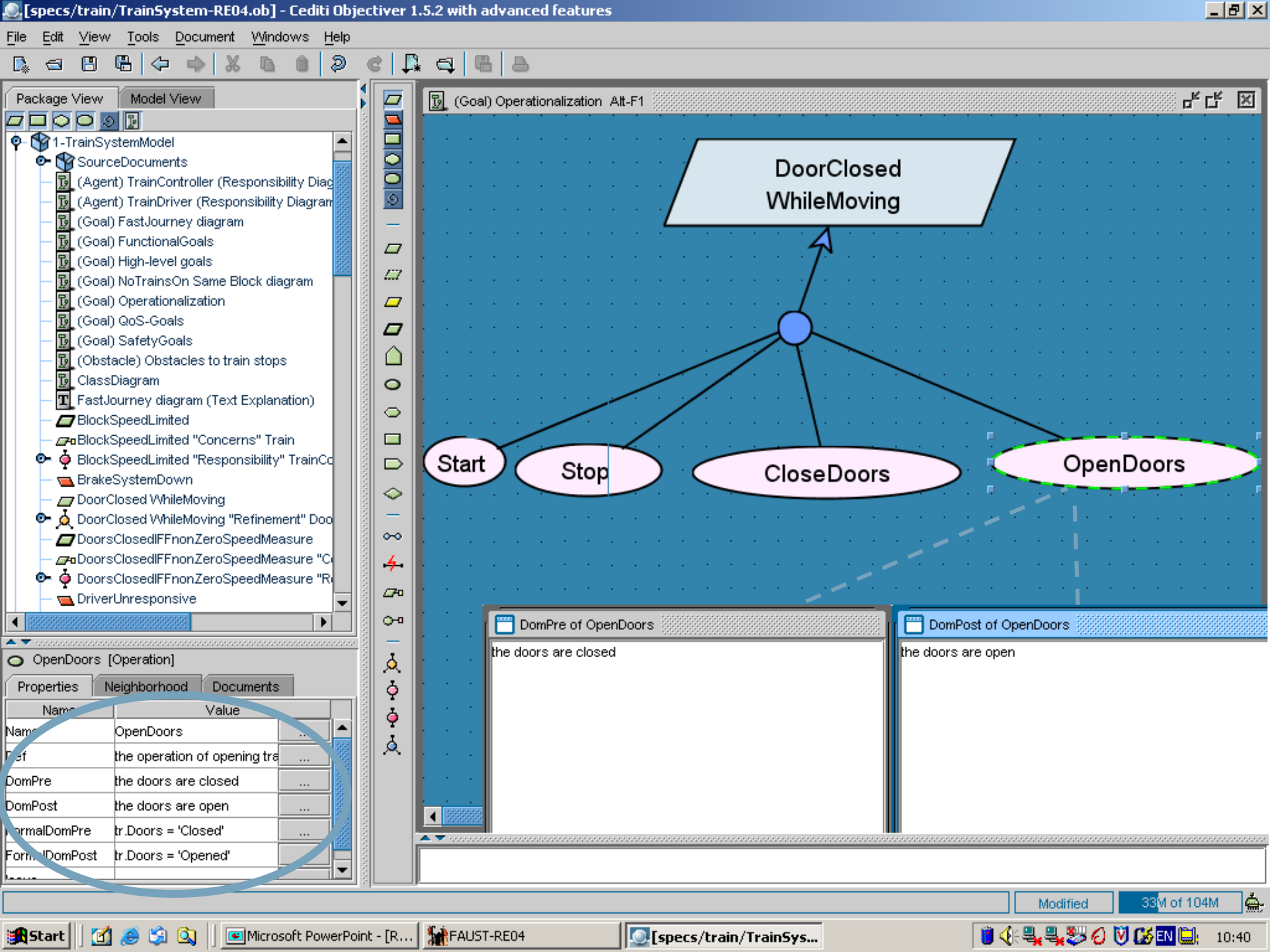
4. analysis:
enriched objects
from new goals

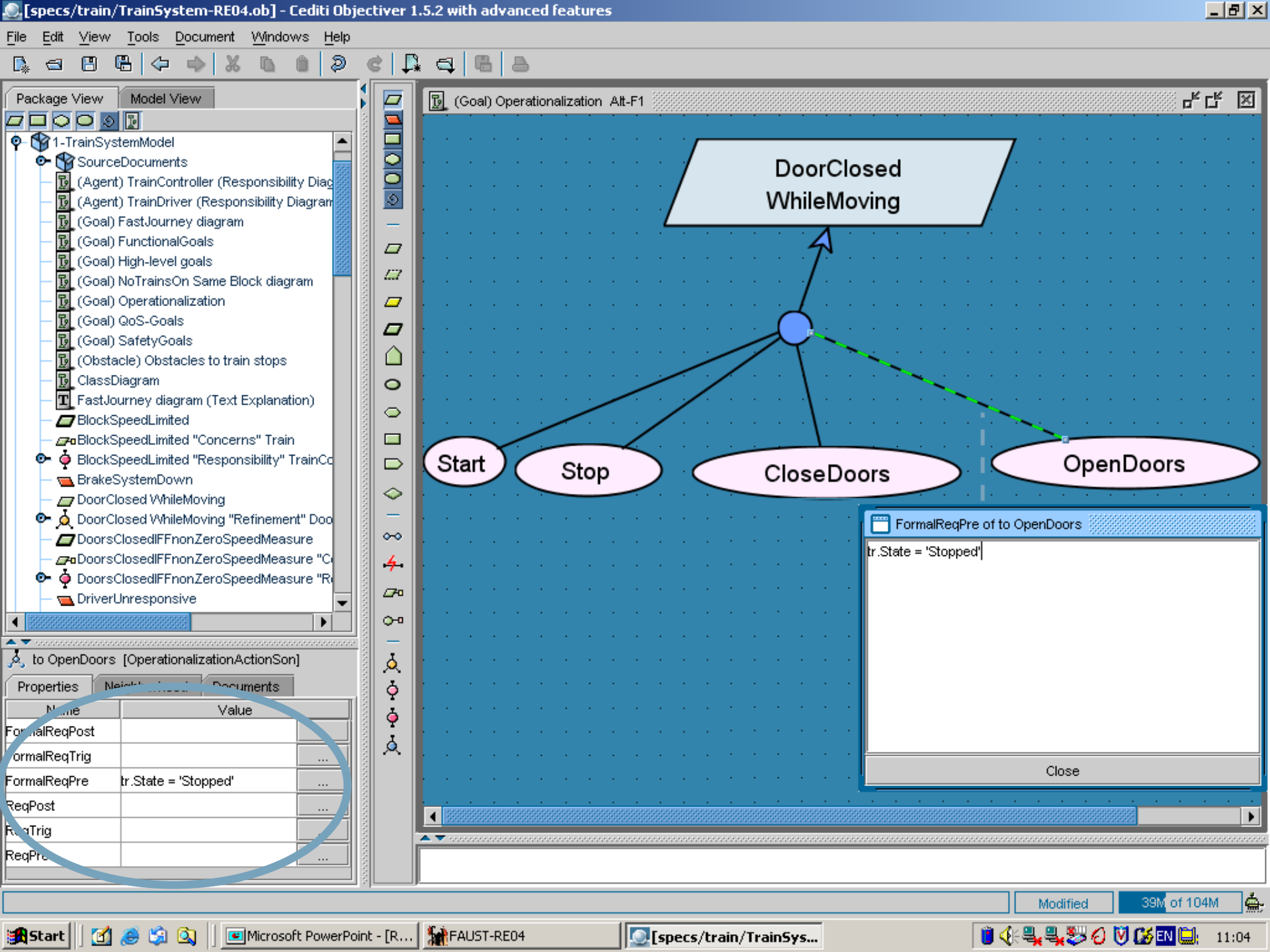


5. Responsibility analysis:
agent OR-assignment

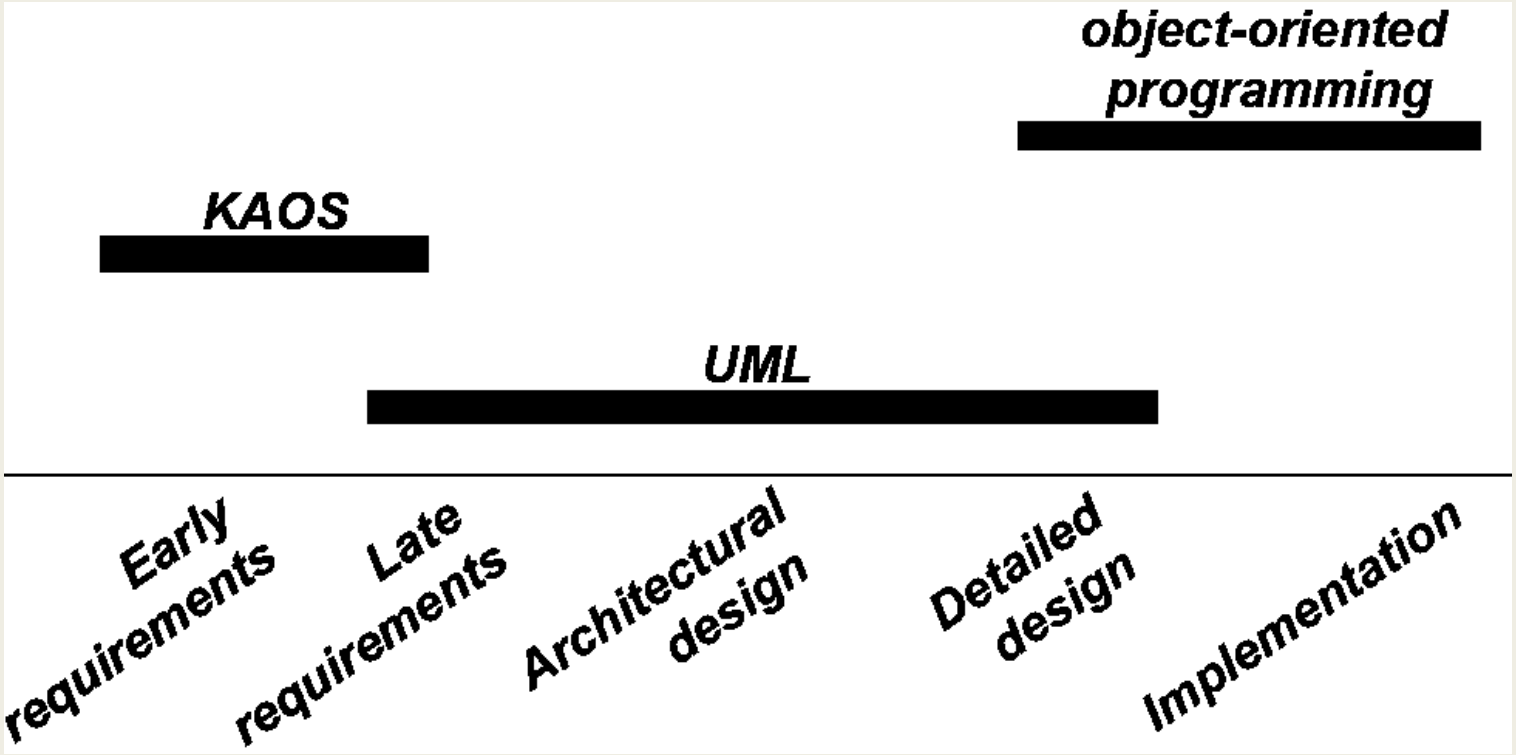


6. Operationalization
& behavior analysis


[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Package View
Model View
1-TrainSystemModel
SourceDocuments
(Agent) TrainController (Responsibility Diag
(Agent) TrainDriver (Responsibility Diagram
(Goal) FastJourney diagram
(Goal) FunctionalGoals
(Goal) High-level goals
(Goal) NoTrainsOn Same Block diagram
(Goal) Operationalization
(Goal) QoS-Goals
(Goal) SafetyGoals
(Obstacle) Obstacles to train stops
ClassDiagram
FastJourney diagram (Text Explanation)
BlockSpeedLimited
BlockSpeedLimited "Concerns" Train
BlockSpeedLimited "Responsibility" TrainCo
BrakeSystemDown
DoorClosed WhileMoving
DoorClosed WhileMoving "Refinement" Doo
DoorsClosedIFFnonZeroSpeedMeasure
DoorsClosedIFFnonZeroSpeedMeasure "C
DoorsClosedIFFnonZeroSpeedMeasure "R
DriverUnresponsive
OpenDoors [Operation]
Properties
Neighborhood
Documents
Name
Value
OpenDoors
the operation of opening tra
DomPre
the doors are closed
DomPost
the doors are open
tr.Doors = 'Closed'
tr.Doors = 'Opened'
(Goal) Operationalization Alt-F1
DoorClosed WhileMoving
Start
Stop
CloseDoors
OpenDoors
DomPre of OpenDoors
the doors are closed
DomPost of OpenDoors
the doors are open
Modified
33M of 104M
Start
Microsoft PowerPoint - [R...
FAUST-RE04
[specs/train/TrainSys...
10:40


[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Package View
Model View
1-TrainSystemModel
SourceDocuments
(Agent) TrainController (Responsibility Diag
(Agent) TrainDriver (Responsibility Diagram
(Goal) FastJourney diagram
(Goal) FunctionalGoals
(Goal) High-level goals
(Goal) NoTrainsOn Same Block diagram
(Goal) Operationalization
(Goal) QoS-Goals
(Goal) SafetyGoals
(Obstacle) Obstacles to train stops
ClassDiagram
FastJourney diagram (Text Explanation)
BlockSpeedLimited
BlockSpeedLimited "Concerns" Train
BlockSpeedLimited "Responsibility" TrainCo
BrakeSystemDown
DoorClosed WhileMoving
DoorClosed WhileMoving "Refinement" Doo
DoorsClosedIFFnonZeroSpeedMeasure
DoorsClosedIFFnonZeroSpeedMeasure "C
DoorsClosedIFFnonZeroSpeedMeasure "R
DriverUnresponsive
to OpenDoors [OperationalizationActionSon]
Properties
Documents
Name
Value
FormalReqPost
FormalReqTrig
FormalReqPre
tr.State = 'Stopped'
ReqPost
ReqTrig
ReqPre
(Goal) Operationalization Alt-F1
DoorClosed WhileMoving
Start
Stop
CloseDoors
OpenDoors
FormalReqPre of to OpenDoors
tr.State = 'Stopped'
Close
Modified
39M of 104M
Start
Microsoft PowerPoint - [R...
FAUST-RE04
[specs/train/TrainSys...
EN
11:04

Goals and UML



Bibliography

- Lamsweerde, Axel van; "*Goal-Oriented Requirements Engineering: A Guided Tour*"; *Université Catholique de Louvain; Louvain-la-Neuve; Belgium; 2001*
- "A KAOS Tutorial"; Objectiver; 2003
- Lamsweerde, Axel van; "*Goal-Oriented Requirements Engineering: From System Objectives to UML Models to Precise Software Specification*"; *Université Catholique de Louvain; Louvain-la-Neuve; Belgium; May 2003*
- Lamsweerde, Axel van; "*Requirements Engineering - - from system goals to UML models to software specifications*"; *Wiley, 2009*