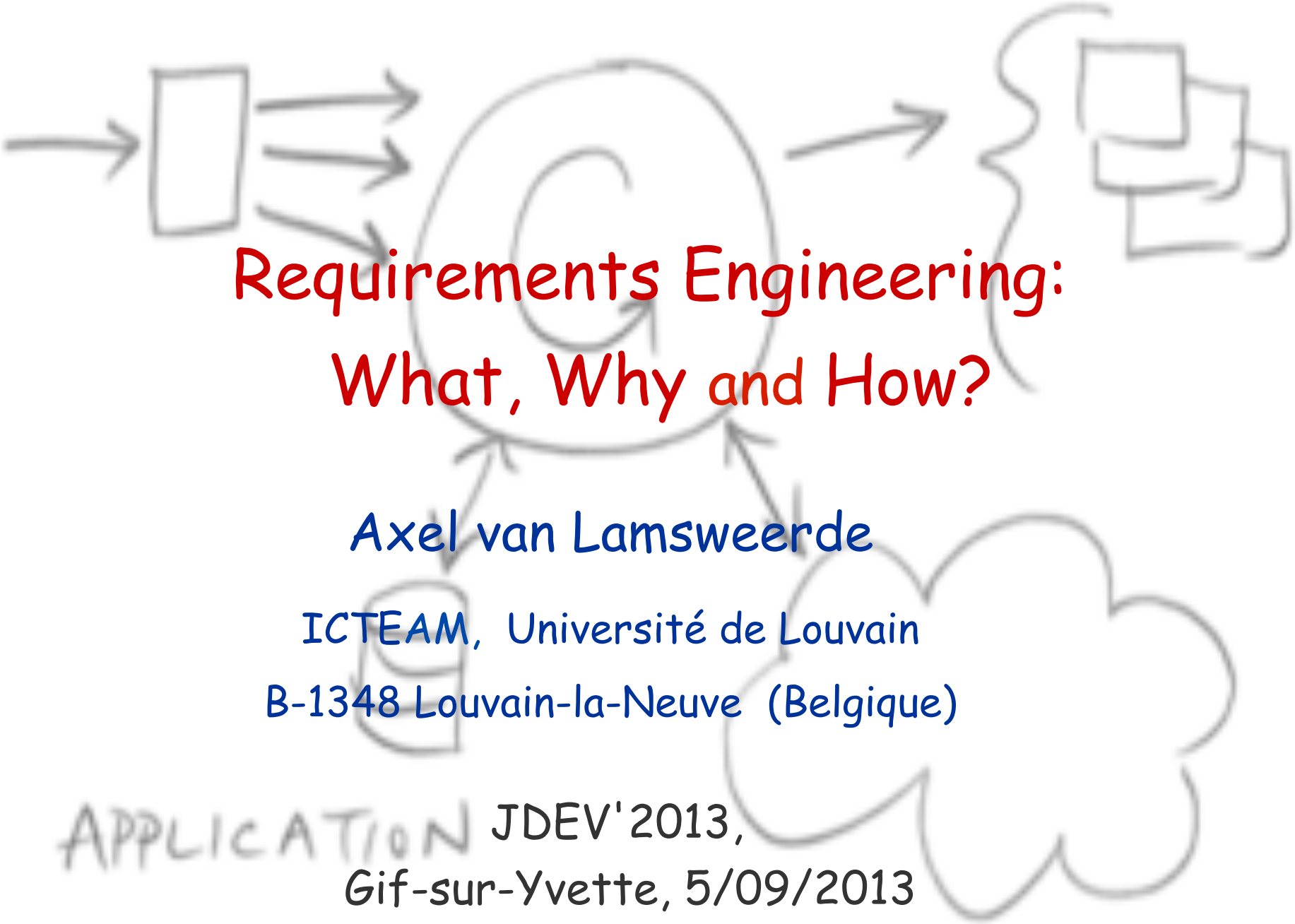


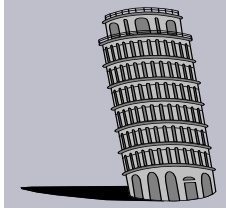
Requirements Engineering: What, Why and How?

Axel van Lamsweerde

ICTEAM, Université de Louvain

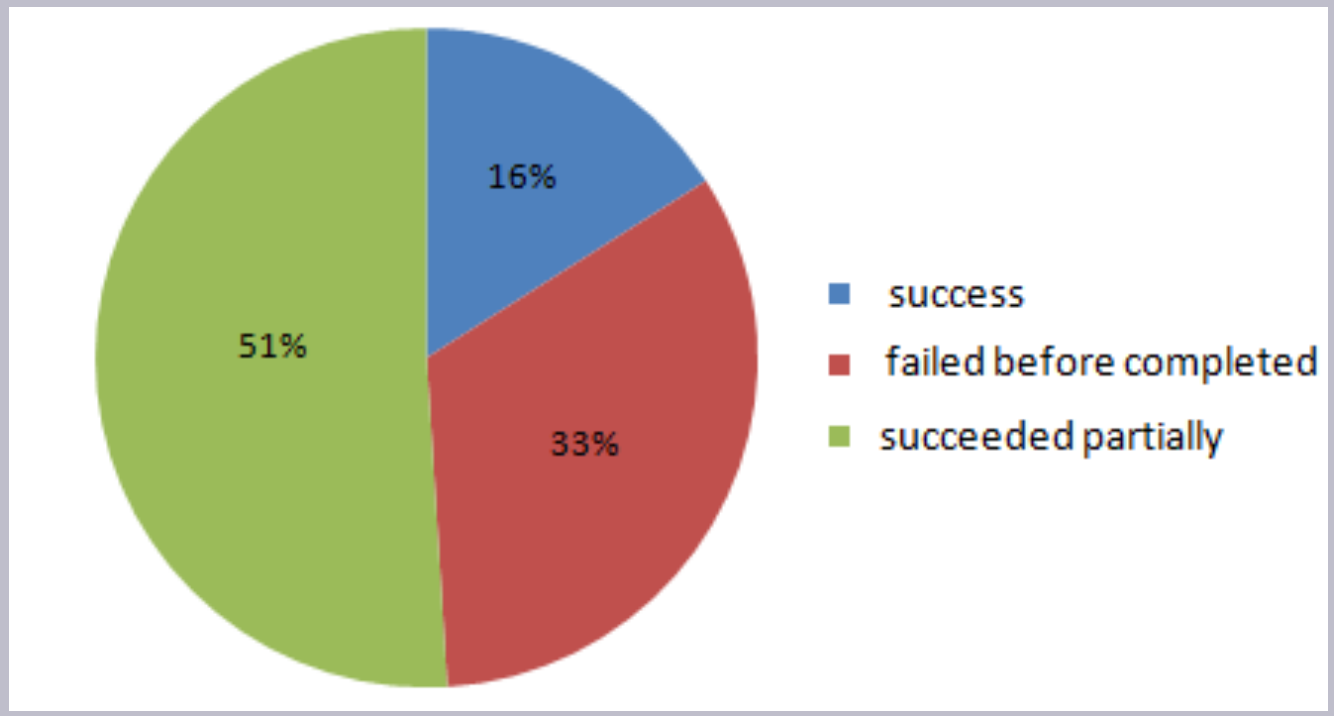
B-1348 Louvain-la-Neuve (Belgique)

JDEV'2013,
Gif-sur-Yvette, 5/09/2013



Motivation: the requirements problem

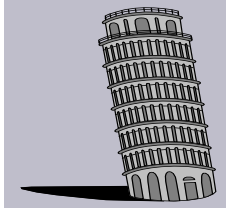
Survey of 350 US companies, **8000 projects**



(partial success = partial functionalities, excessive costs, big delays)

Major source of failure:

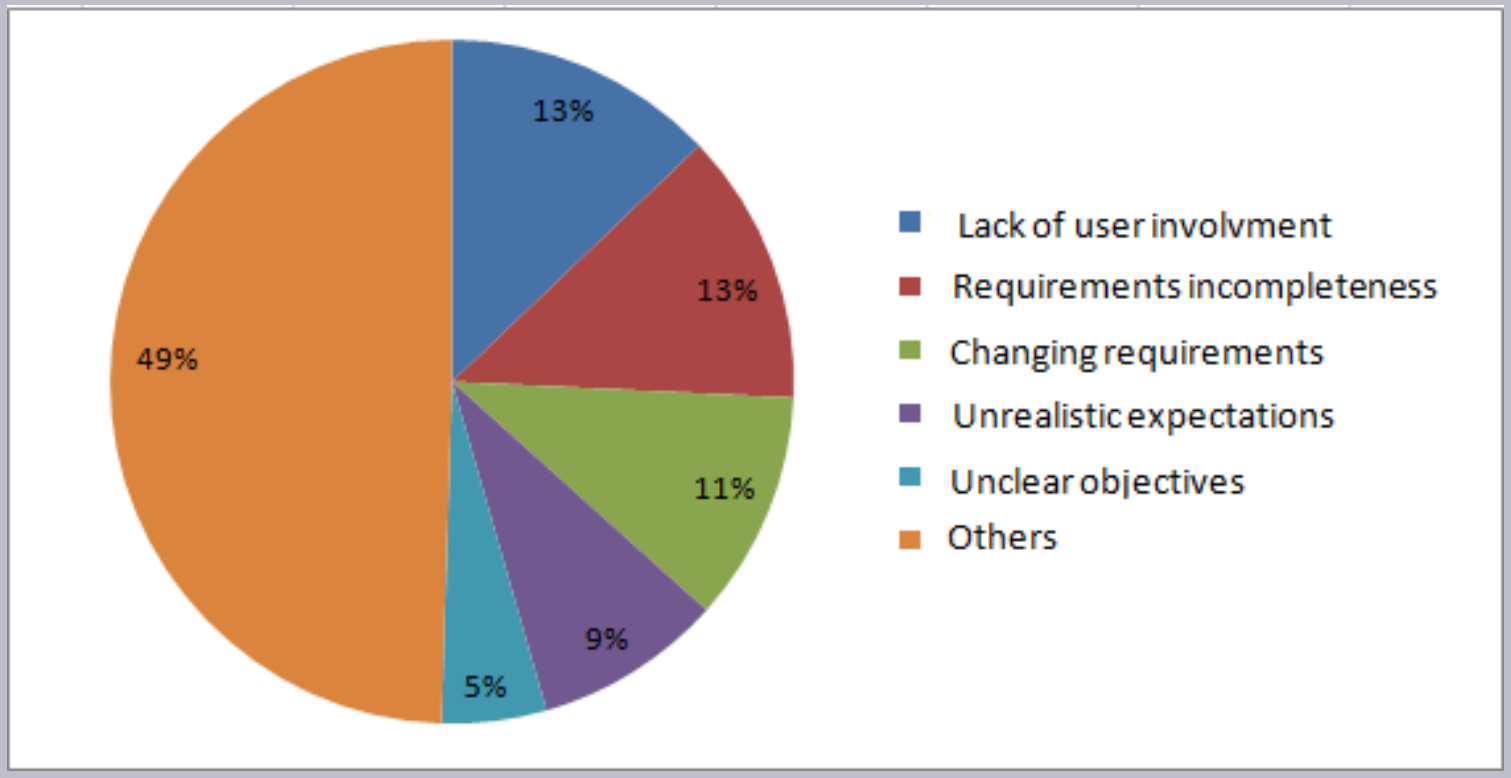
poor requirements engineering ≅ 50% responses



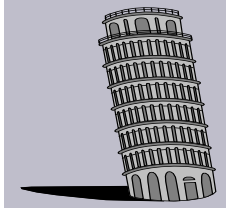
Motivation: the requirements problem (2)

Major source of failure:

poor requirements engineering ≅ 50% responses:



www.standishgroup.com/chaos.html



Motivation: the requirements problem (3)

- Survey of 3800 EUR organizations, 17 countries
- Main software problems perceived to be in...
  - requirements specification

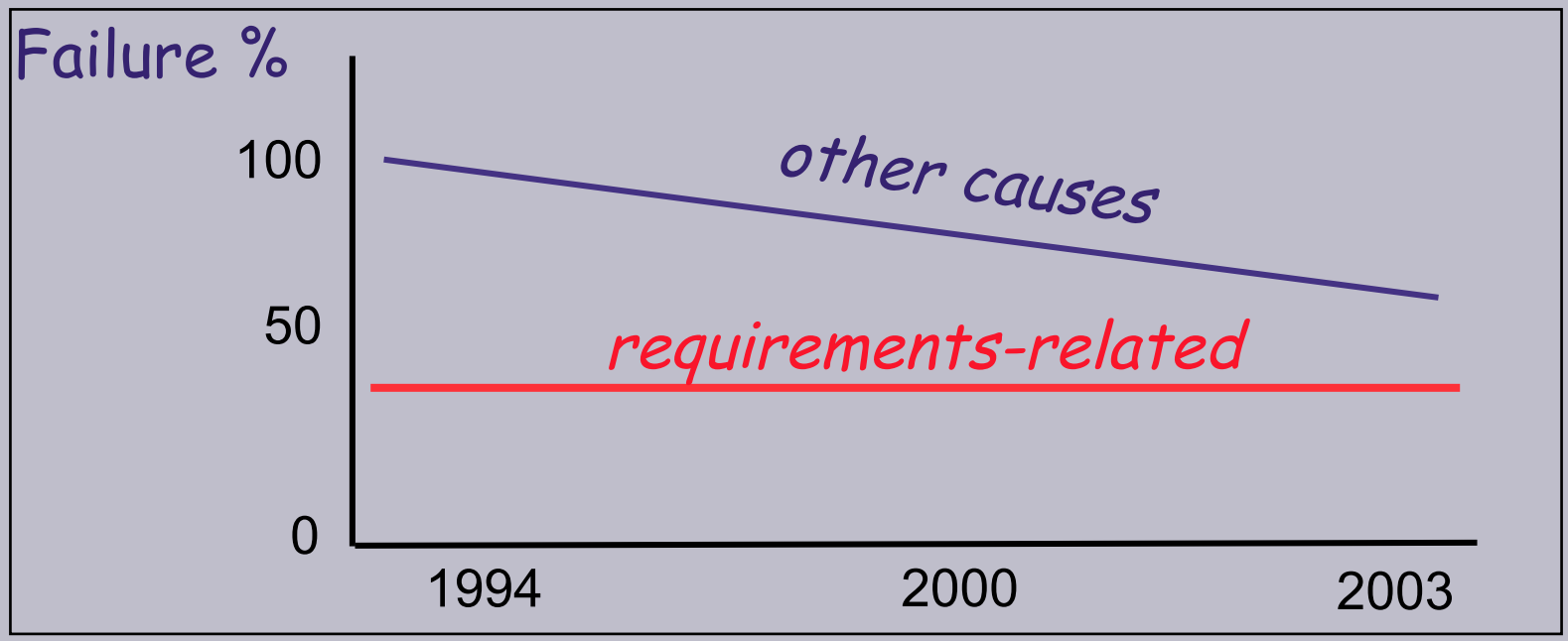
    > 50% responses

  - requirements evolution management

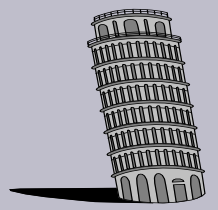
    50% responses

[European Software Institute, 1996]

The requirements problem is perceived to persist in spite of progress in software technology

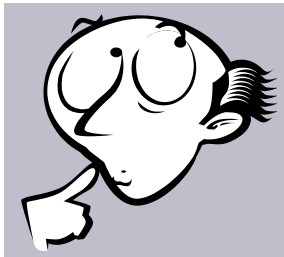


[J. Maresco, IBM developersWork, 2007]



Outline

- Requirements engineering (RE)
  - What it is
  - Why it is difficult
  - Why it is important
- Models for RE
  - Why models, model what?
- Goal-oriented RE
  - Goal-based model building
  - Goal-oriented model analysis for RE



Requirements Engineering, roughly ...

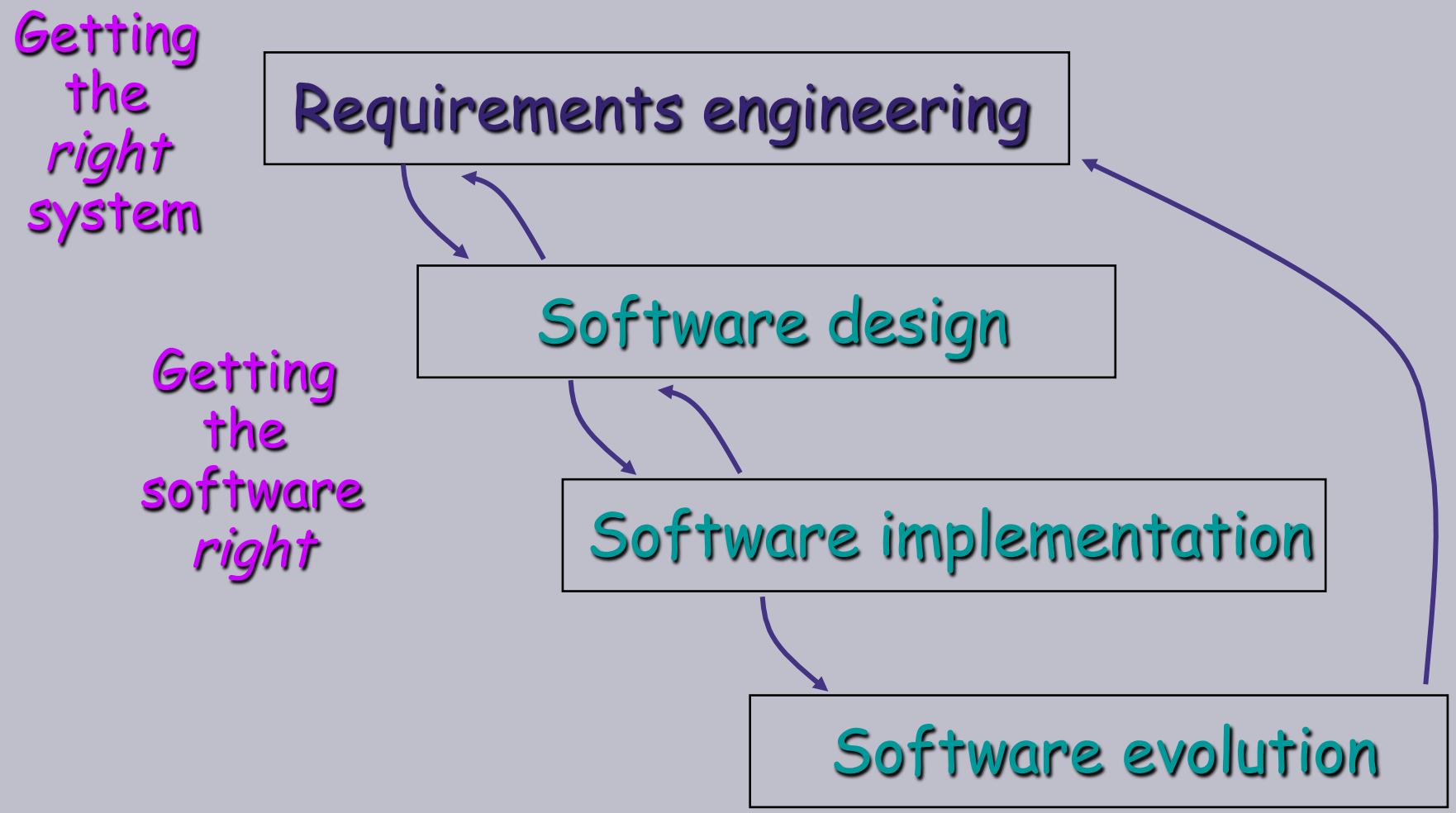
- Analyze problems with an existing system (system-***as-is***),
- Identify objectives & opportunities for new system (system-***to-be***),
- Define functionalities of, constraints on, responsibilities in system-to-be,
- Specify all of these in a *requirements document*

System = software + environment
(people, devices, other software)

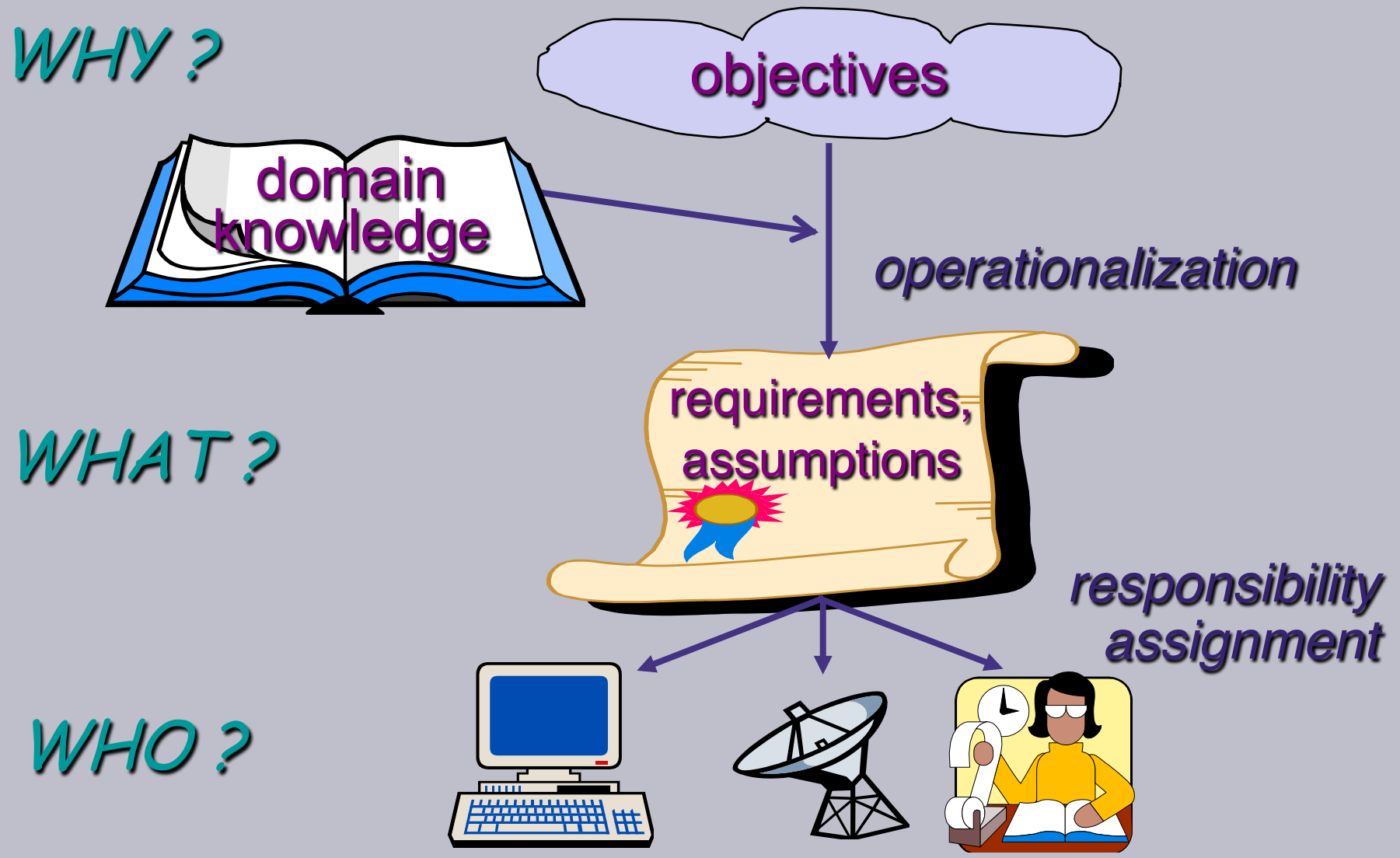
Example:
transportation between airport terminals

- Problem (system-***as-is***):
  - passengers frequently missing flight connections among terminals; slow & inconvenient bus transportation
  - number of passengers increasing regularly
- Objectives, options (system-***to-be***):
  - support high-frequency trains between terminals
  - with **or** without train drivers ?
- Functionalities, constraints:
  - software-based control of train accelerations, of doors opening etc. to achieve *prompt* and *safe* transportation
- Deliverable: requirements document for system-to-be

Requirements in the software lifecycle

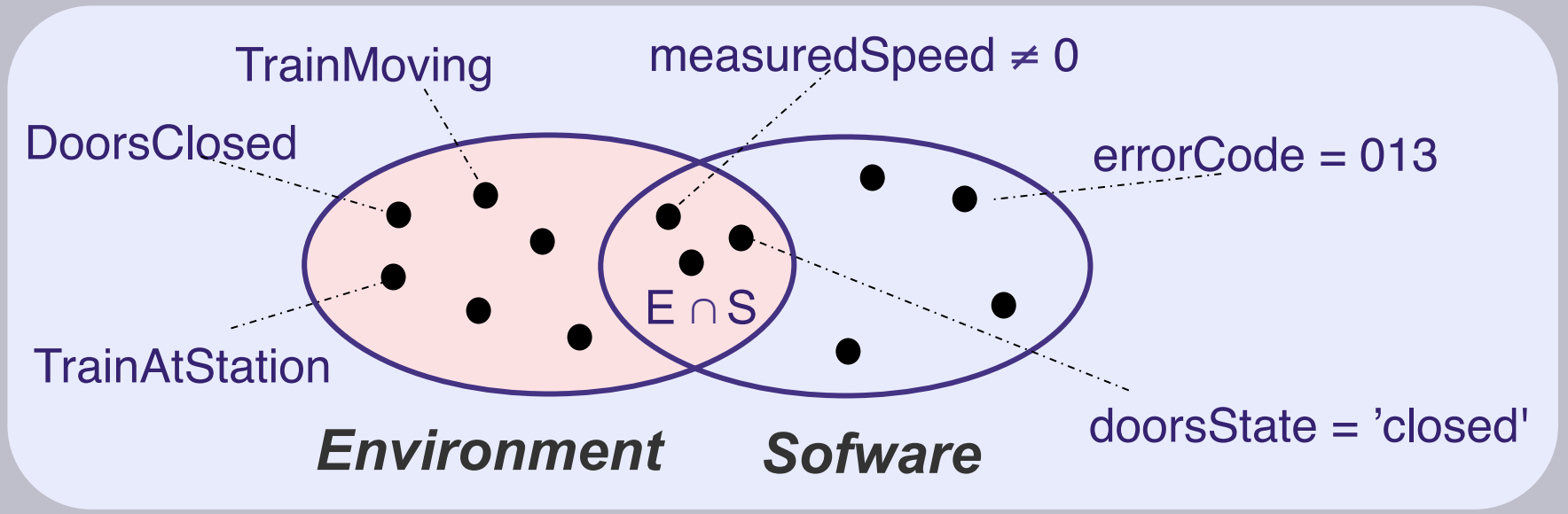


What is RE about ?



Requirements are about the **problem world**

The World and the Machine [M. Jackson]



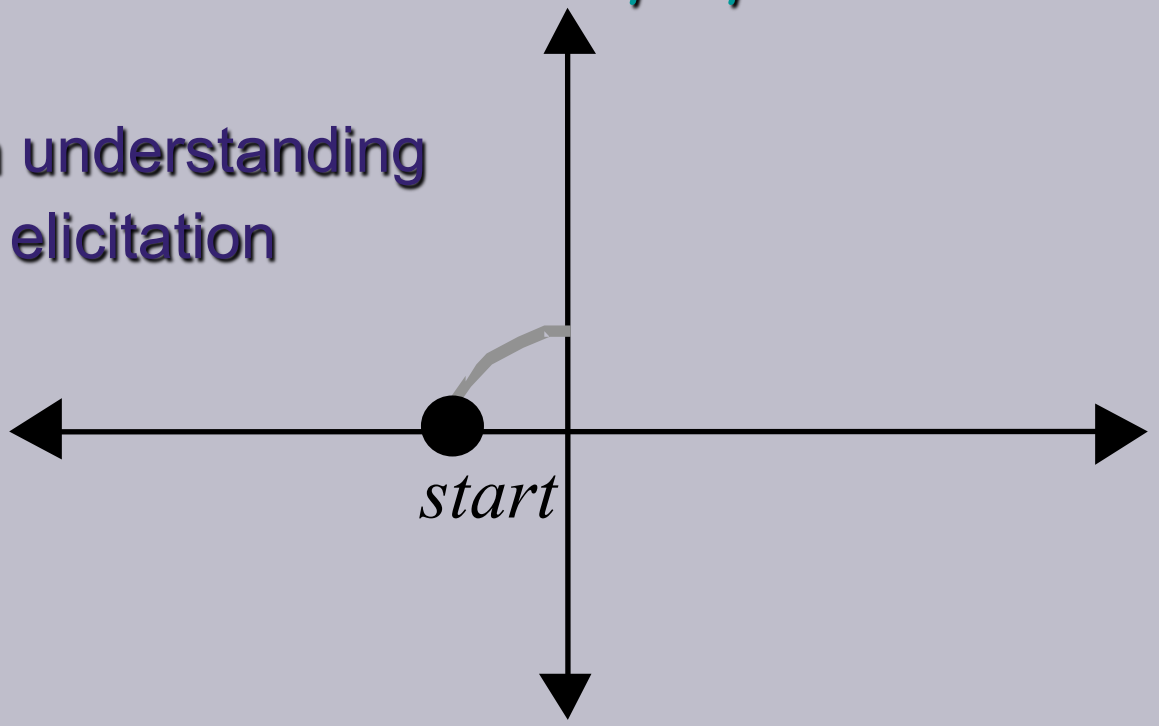
Requirements are based on environment *properties & assumptions*

on environment variables & shared variables

The RE process

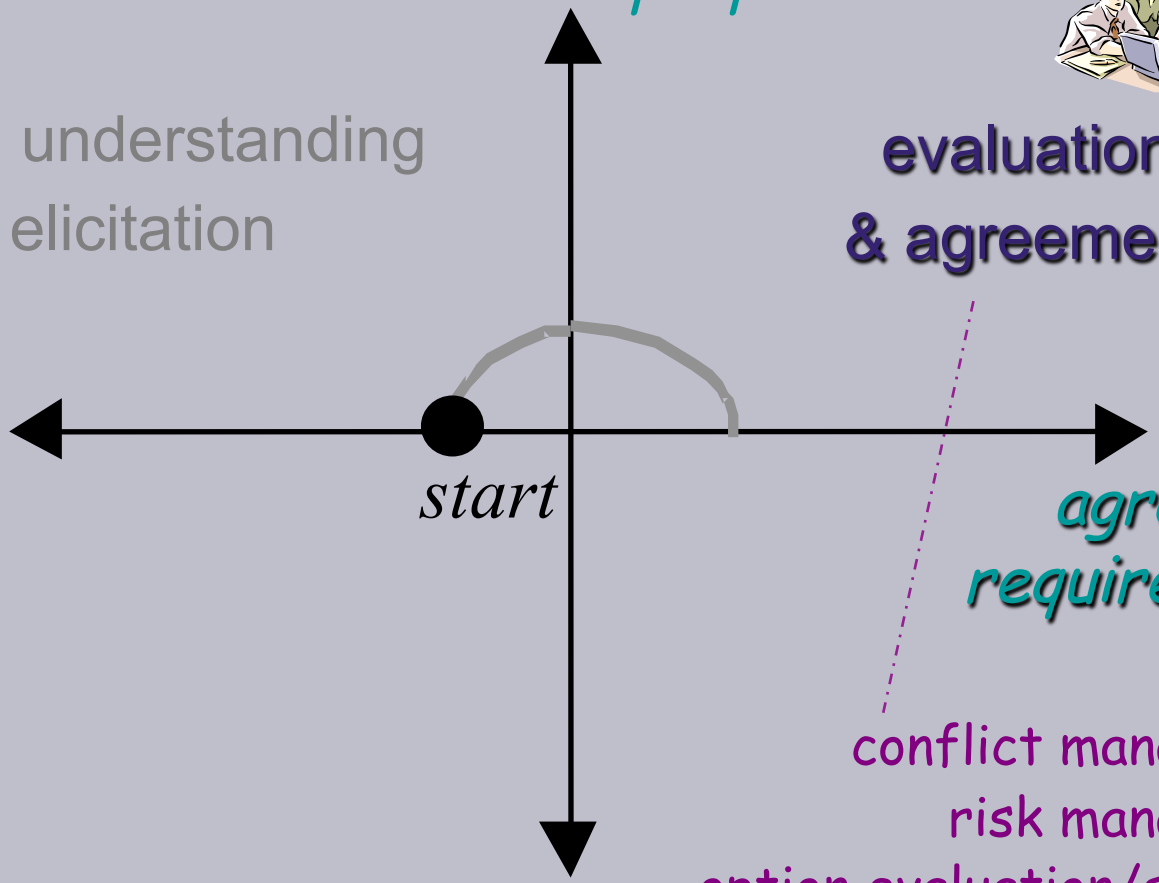
alternative proposals

domain understanding
& elicitation



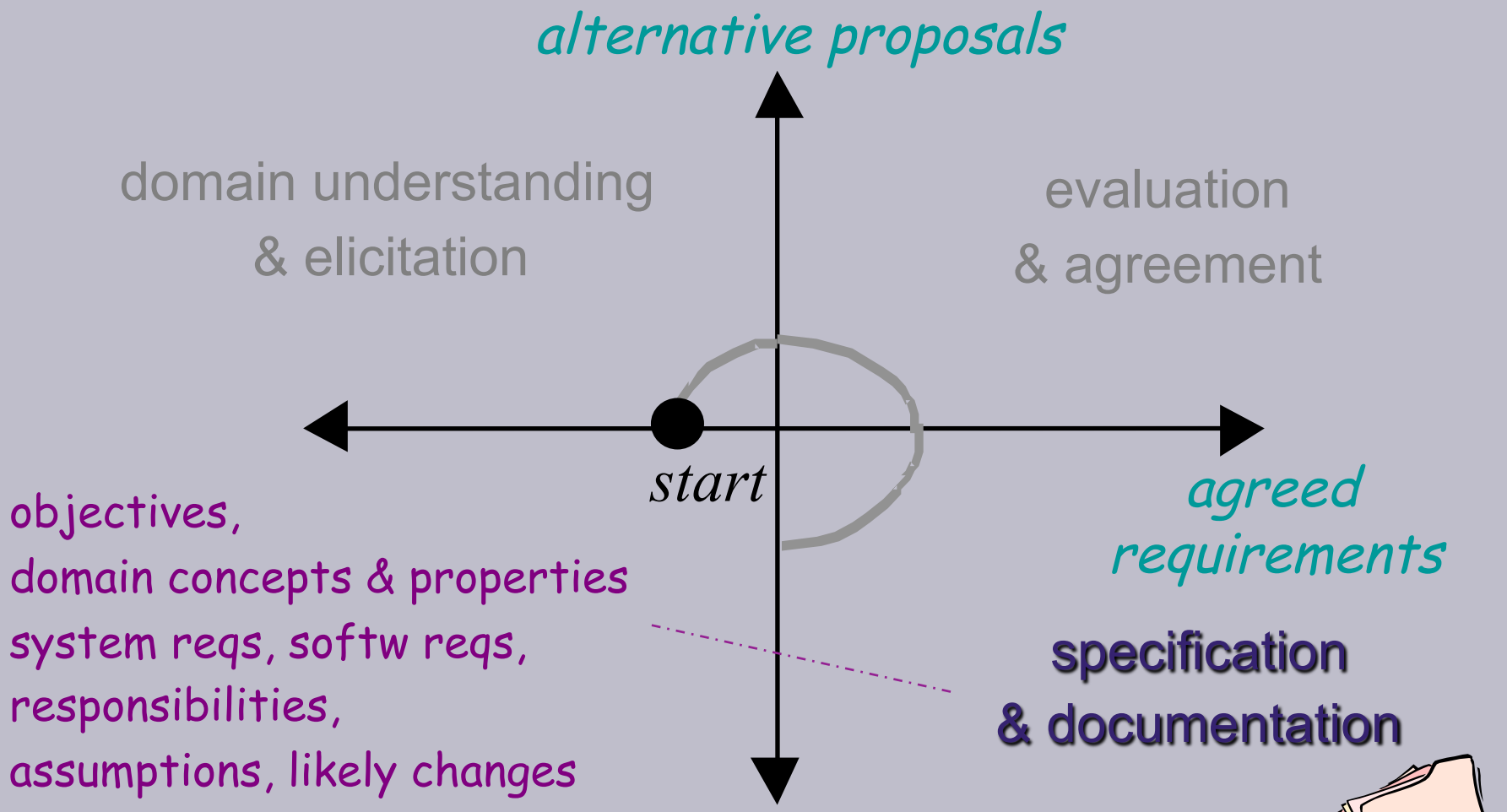
business organization,
application domain, stakeholders
problems, improvement objectives, constraints,
alternative options, preliminary requirements

The RE process

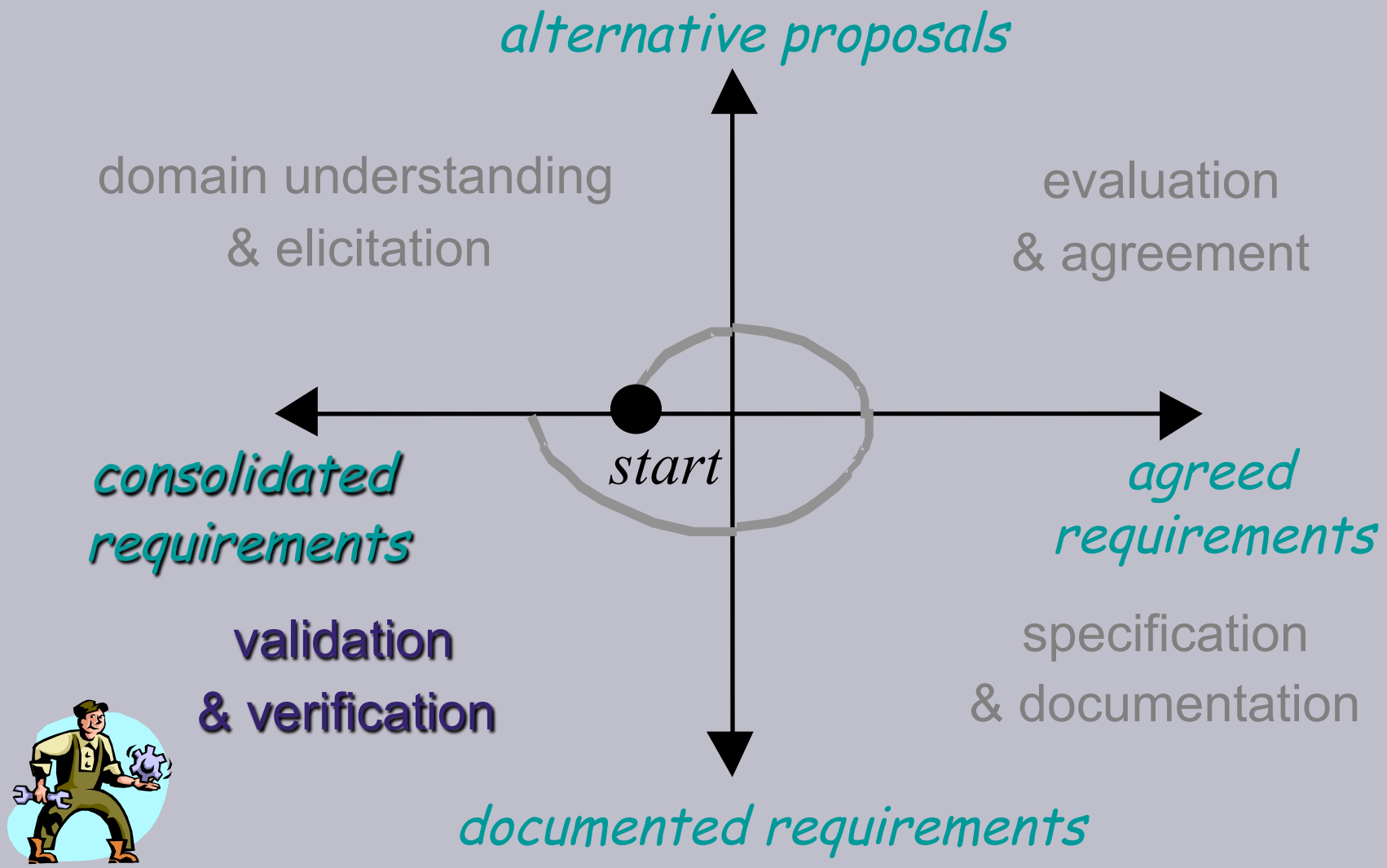


conflict management,
risk management,
option evaluation/selection,
requirements prioritization

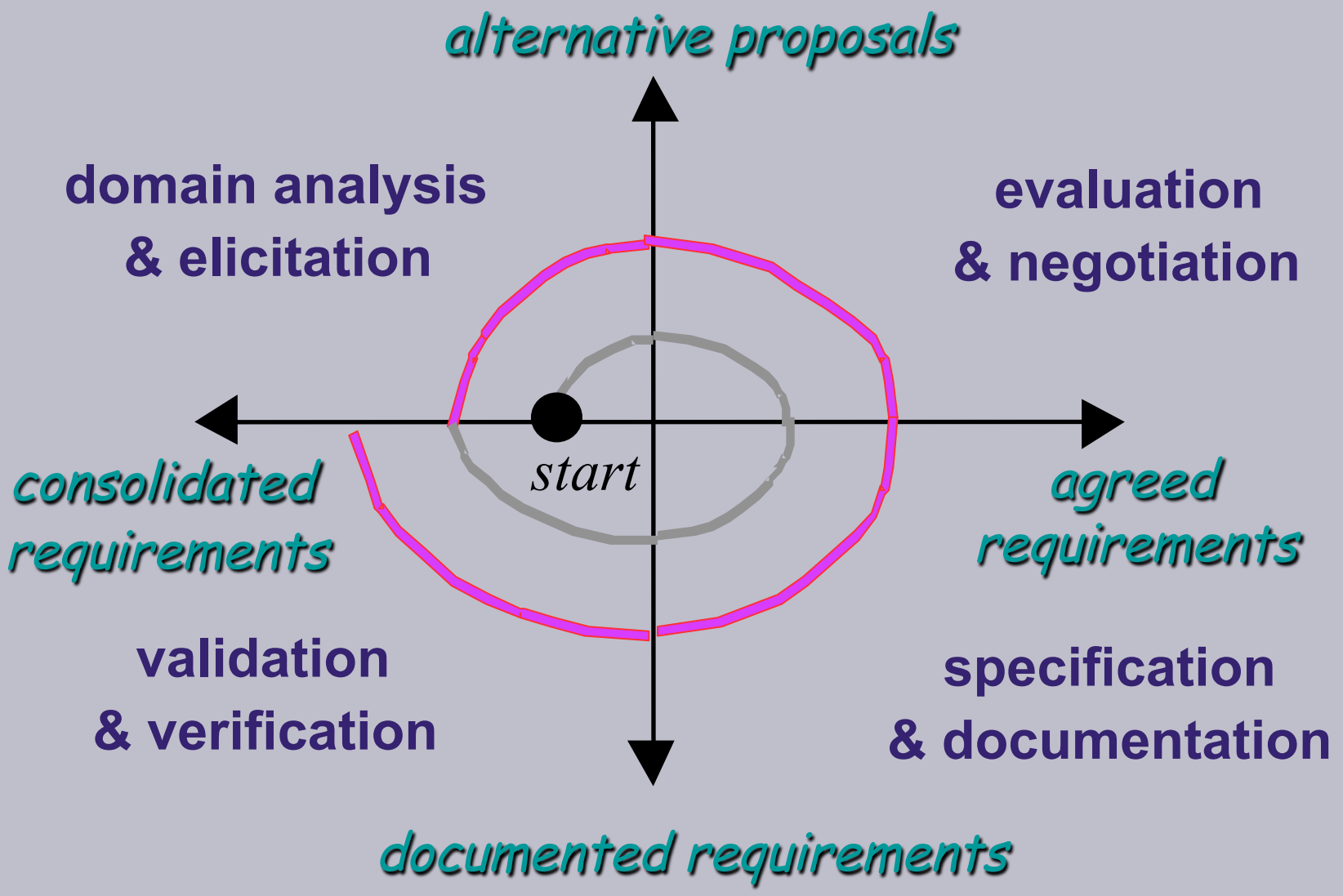
The RE process



The RE process



RE is an iterative process



Why RE is hard

- Broad scope
  - **two** systems: system-*as-is*, system-*to-be*
  - system-to-be = software **+** environment
  - hybrid environment:

    human organizations, policies

    devices, physical laws

Why RE is hard

- Broad scope
  - two systems: system-*as-is*, system-*to-be*
  - system-to-be = software + environment
  - hybrid environment:
    human organizations, policies
    devices, physical laws
- Multiple concerns
  - functional, quality, development concerns
- Multiple abstraction levels
  - strategic objectives, operational details

Why RE is hard (2)

- Multiple stakeholders
  - with different background
  - with different interests and conflicting viewpoints

    project managers
    domain experts
    users, clients
    subcontractors
    decision makers
    analysts, developers
    ...

Why RE is hard (3)

- Multiple intertwined activities during iterative
  elicitation-evaluation-specification-consolidation
  - conflict management
    functionality vs. quality vs. cost
    diverging viewpoints among stakeholders
  - risk management
    anticipation of hazards & threats

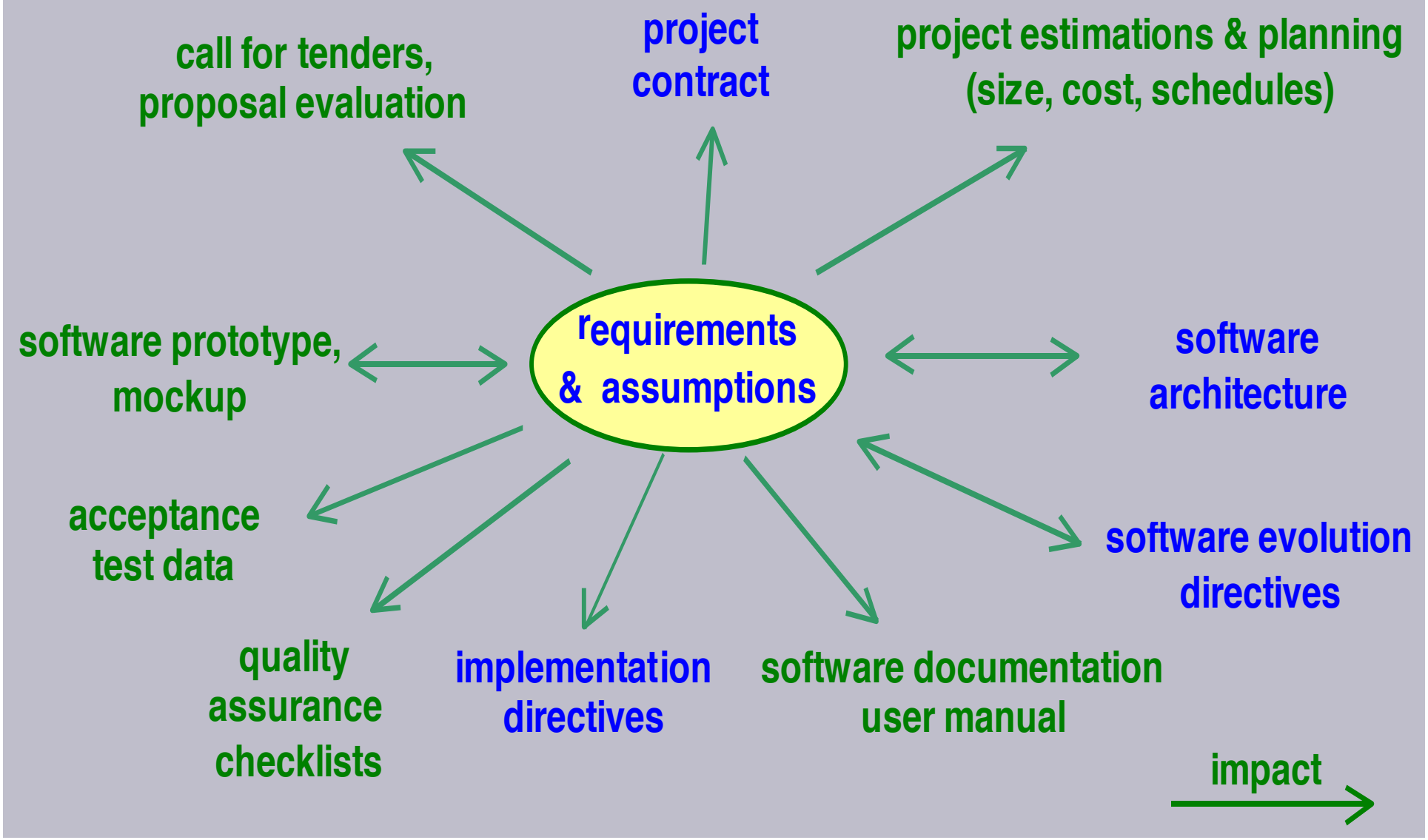
Why RE is hard (3)

- Multiple intertwined activities during iterative
  elicitation-evaluation-specification-consolidation
  - conflict management
    functionality vs. quality vs. cost
    diverging viewpoints among stakeholders
  - risk management
    anticipation of hazards and threats
  - evaluation of alternatives, prioritization
  - quality assurance
  - change anticipation

Why RE is important

- Legal impact
  - contractual commitment client-provider
- Social impact
  - *from* user satisfaction
    to degradation of working conditions
    to system rejection
- Technical impact
  - on many software-related artefacts

Requirements impact on many software artefacts



Why RE is important (2)

- Impact on certification
  - quality standards & certification authorities require mastered RE process
- Impact on economy, security, and safety
  - cost and consequences of **errors** in ...

    requirements on software

    assumptions on environment

Wide variety of errors & flaws in requirements, assumptions, domain properties

- Incompleteness *critical !*
- Inconsistency *critical !*
- Inadequacy *critical !*
- Ambiguity *critical !*
- Unintelligibility
- Unmeasurability
- Poor structure
- Overspecification
- Noise

Errors in requirements/assumptions are ...

- the most numerous
  - ± 40% of software errors
- the most persistent
  - often found very late, after software delivery
- the most expensive
  - cost ... 5x more if fixed during design
    10x more if fixed during implementation
    20x more if fixed during integration testing
    200x more if fixed after delivery
  - account for 66% of software error costs

[Boehm, Jones, Lutz, Hooks & Farry, ...]

Requirements errors are the most dangerous

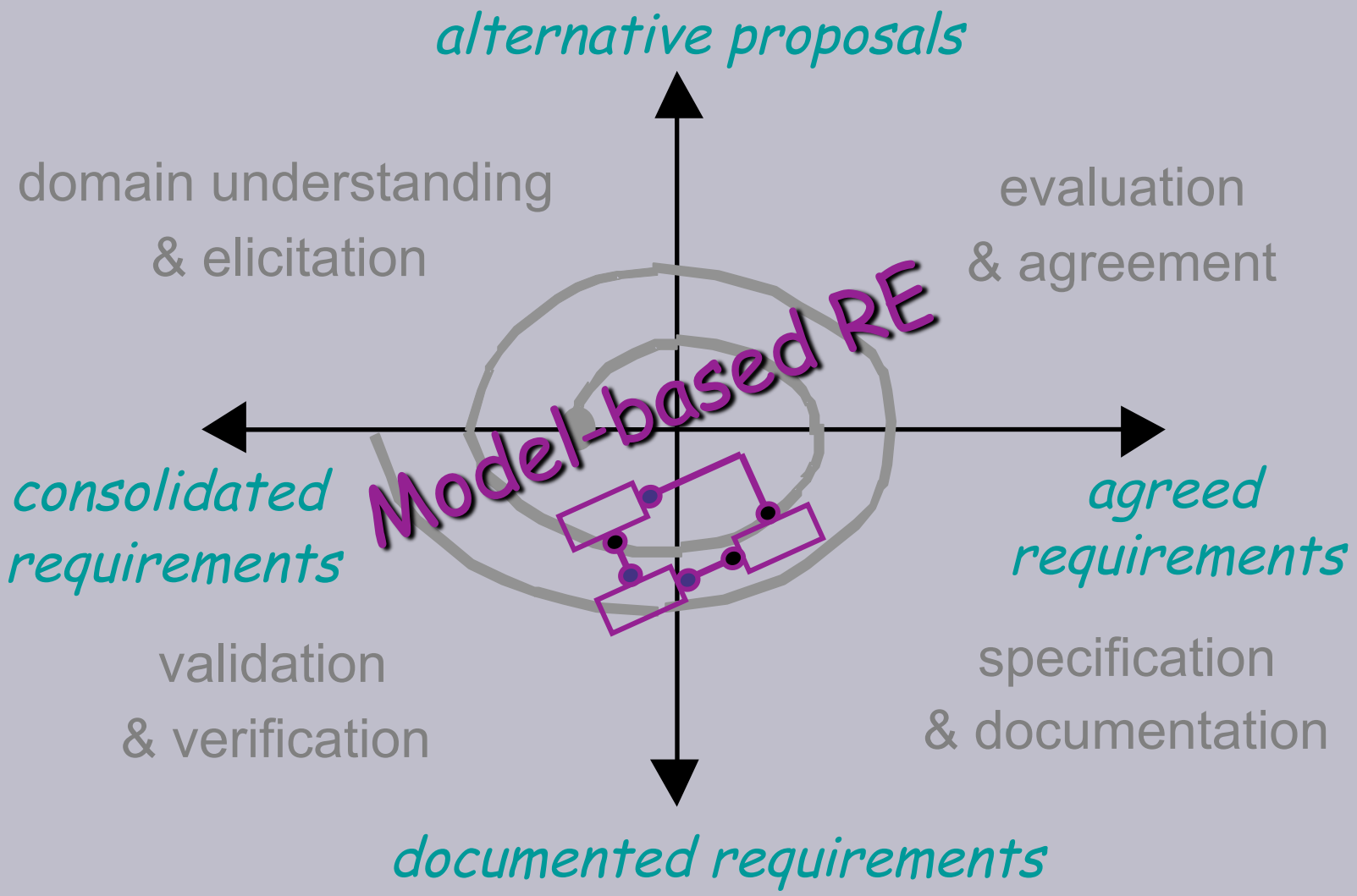
- Shooting of IranAir airbus, US Aegis/Vincennes, 1988
  - Missing timing between 2 threat events in requirements on alarm software
- Shooting of US barracks, patriot anti-missile system
  - Midden assumption on maximum usage time
- Fatal delays, London Ambulance System, 1993
  - Wrong assumptions on crew behavior, ambulance localization system, radio communication, ...
- Boeing 757 crash, Cali, 1995
  - Autopilot 's wrong timing/localization assumption on flap extension point
- Etc etc ...

Outline

- Requirements engineering (RE)
  - What it is
  - Why it is difficult
  - Why it is important

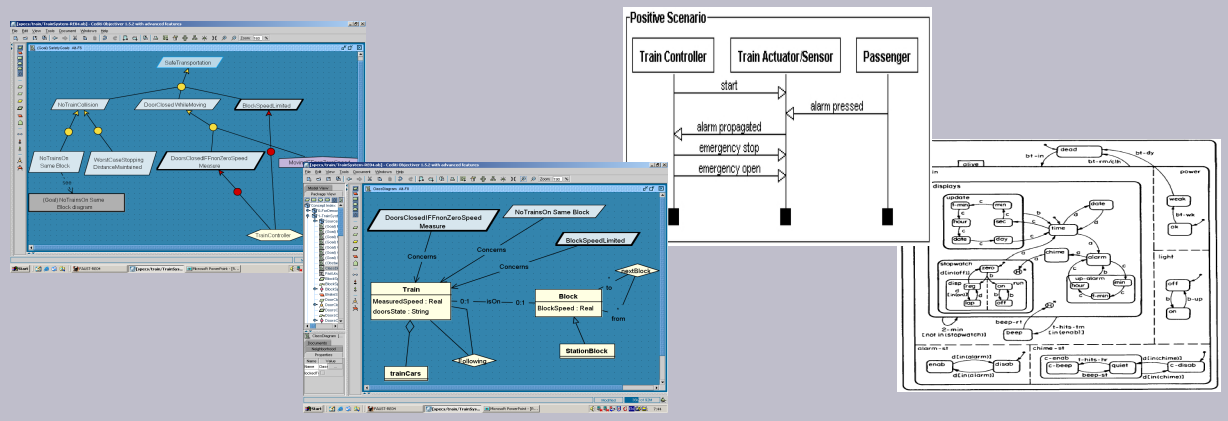
- Models for RE
  - Why model, model what?
- Goal-oriented RE
  - Goal-based model building
  - Goal-oriented model analysis for RE

Models as interfaces among RE tasks



Model-based RE

- To focus on **key aspects**
- To support early **analysis** & fix of critical errors
- To support explanation to stakeholders
- To support decisions among multiple options



Model building at RE time should be **goal-oriented**

To enable ...

- satisfaction arguments *Specs, Assumptions* ⊢ *Goal*
- completeness & pertinence of the model
- early, incremental analysis
- model refinement & synthesis (deductive, inductive)
- reasoning about alternative options
- validation by stakeholders
- backward traceability
- generation of ...
  - requirements document
  - architectural fragments
  - runtime reqirements monitors

Goals as declarative abstractions for system modeling at RE time

- **Goal**
  - **prescriptive** statement of intent about *system*

    "Trains shall stop at stop signals"

- **Domain property**
  - **descriptive** statement about *environment*

    "Train doors are either open or closed"

- Both used for model building
  - Goals may be refined, negotiated, weakened, prioritized ... ***unlike*** domain properties

Goals are formulated at different levels of abstraction

- **Higher-level** goals

  strategic, coarse-grained

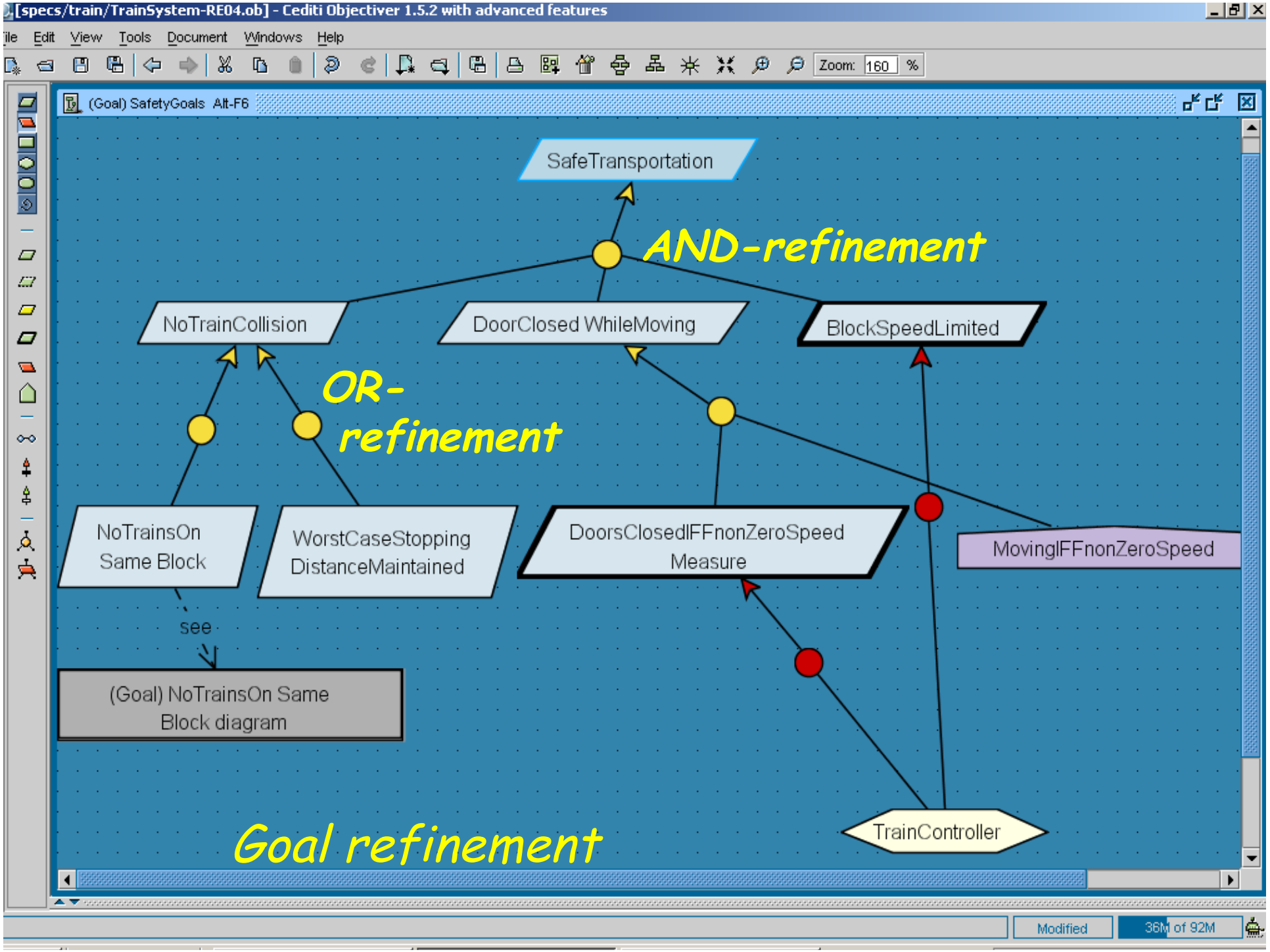
  "50% increase of transportation capacity"

- **Lower-level** goals

  technical, fine-grained

  "Acceleration command sent every 3 secs"

- Linked together by AND/OR refinement links


[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
AND-refinement
NoTrainCollision
DoorClosed WhileMoving
BlockSpeedLimited
OR-refinement
NoTrainsOn Same Block
WorstCaseStopping DistanceMaintained
DoorsClosedIFFnonZeroSpeed Measure
MovingIFFnonZeroSpeed
see
(Goal) NoTrainsOn Same Block diagram
TrainController
Goal refinement
Modified
36M of 92M

Goal satisfaction requires agent cooperation

- **Agent**: active component, controls behaviors

  software-to-be, device, human role, existing sw

  TrainController, Passenger, SpeedSensor, TrackingSystem

The more fine-grained a goal,

the fewer agents required for its satisfaction

SafeTransportation vs. DoorsClosedWhileMoving

Goal satisfaction requires agent cooperation

- **Agent**: active component, controls behaviors
  software-to-be, device, human role, existing sw
  TrainController, Passenger, SpeedSensor, TrackingSystem

  The more fine-grained a goal,
  the fewer agents required for its satisfaction
  SafeTransportation vs. DoorsClosedWhileMoving

- **Requirement**: goal assigned to single software agent
  Train.measuredSpeed ≠ 0 → Train.DoorsState = "closed"

- **Expectation**: goal assigned to single environment agent
  (prescriptive assumption)
  Train.measuredSpeed ≠ 0 iff Train.Speed ≠ 0

Goal types & categories

- Two *types* of goals
  - **Behavioral goals:** prescribe intended behaviors
    can be satisfied in clear-cut sense

    used for deriving operational models & risk analysis

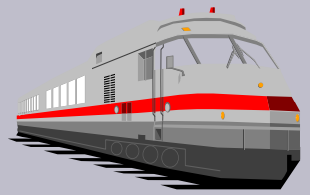
Soft goals prescribe preferred behaviors

Two categories

functional, non-functional goals

Behavioral goals prescribe sets of behaviors declaratively

DoorsClosed WhileMoving



moving closed → stopped closed → stopped open → stopped closed → moving closed

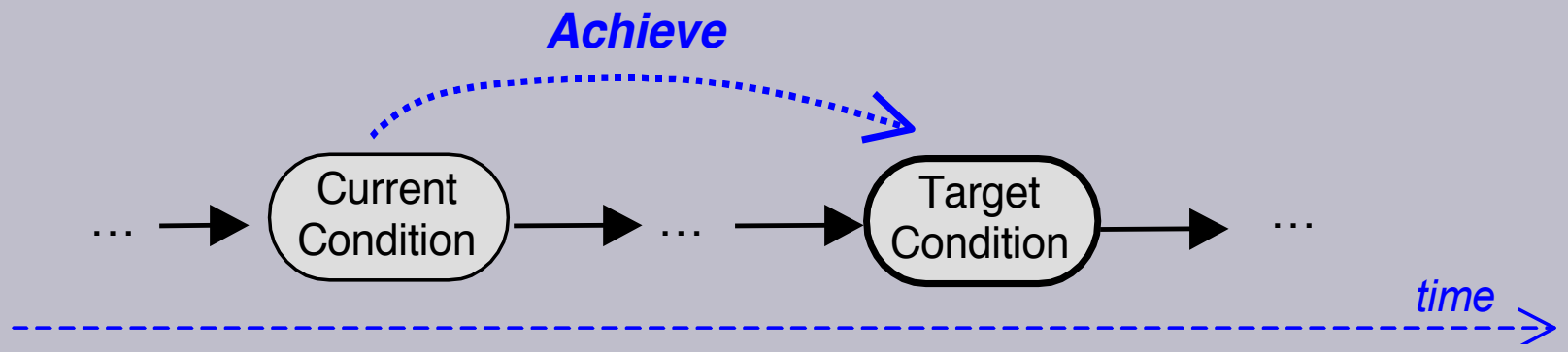
Behavioral goals: subtypes and specification patterns

- ***Achieve*** [TargetCondition]:

  [if CurrentCondition **then]** **sooner-or-later** TargetCondition

Achieve [FastJourney]: 

if train is at some platform **then within 5 minutes** it is at next platform



Behavioral goals: subtypes and specification patterns (2)

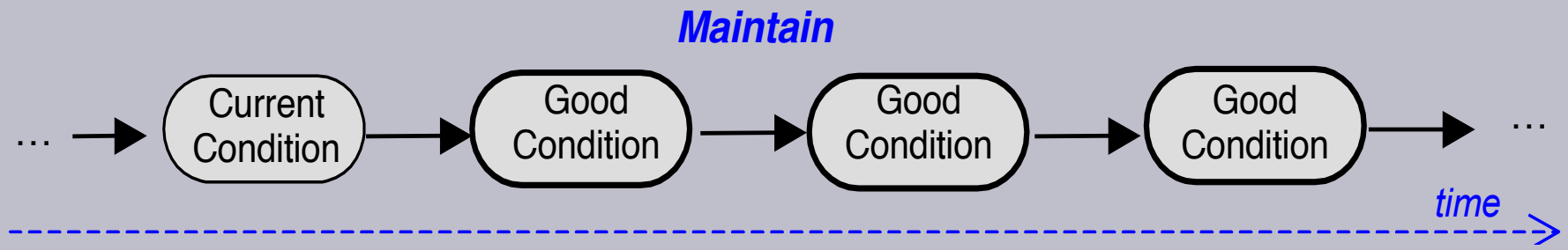
- ***Maintain*** [GoodCondition]:

  [if CurrentCondition **then] always** GoodCondition

  always (**if** CurrentCondition **then** GoodCondition)

Maintain *[DoorsClosedWhileMoving]:*

always (**if** a train is moving **then** its doors are closed)



Goal types & categories

- Two *types* of goals
  - Behavioral goals: prescribe intended behaviors
    can be satisfied in clear-cut sense
    used for deriving operational models & risk analysis
  - **Soft goals:** prescribe preferred behaviors
    cannot be satisfied in clear-cut sense
    used for comparing alternative options

"Stress conditions of air traffic controllers shall be reduced"

Goal types & categories

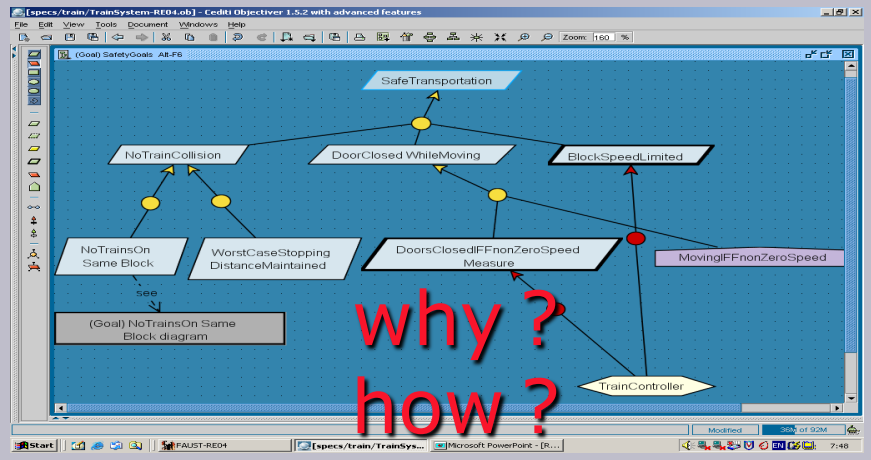
- Two *types* of goals
  - Behavioral goals: prescribe intended behaviors
    can be satisfied in clear-cut sense
    used for deriving operational models
  - Soft goals: prescribe preferred behaviors
    cannot be satisfied in clear-cut sense
    used for comparing alternative options
- Two *categories* of goals
  - **functional**: underlying operation, feature, service, task
  - **non-functional**: quality goals e.g. security, accuracy,...
    architectural goals, development goals,...

What kind of system model for RE ?

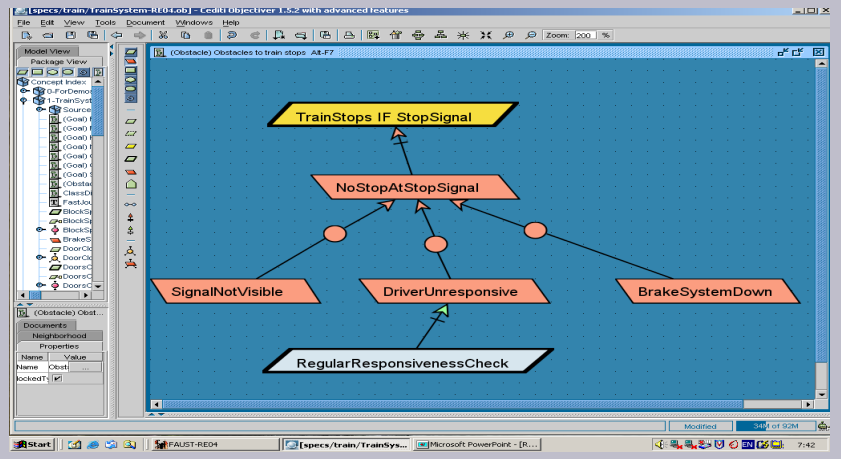
- **Multi-view**
  - Complementary facets, for model comprehensiveness
    intentional, structural, responsibility, operational, behavioral
  - Inter-view rules for structural consistency
- **Multi-formalism**
  - Diagrammatic
    Goal AND/OR refinement graphs
    UML subset: class, sequence, state diagrams
  - Formal (*when & where* needed): real-time temporal logic
  - Quantitative: propagation equations

What models for RE ?

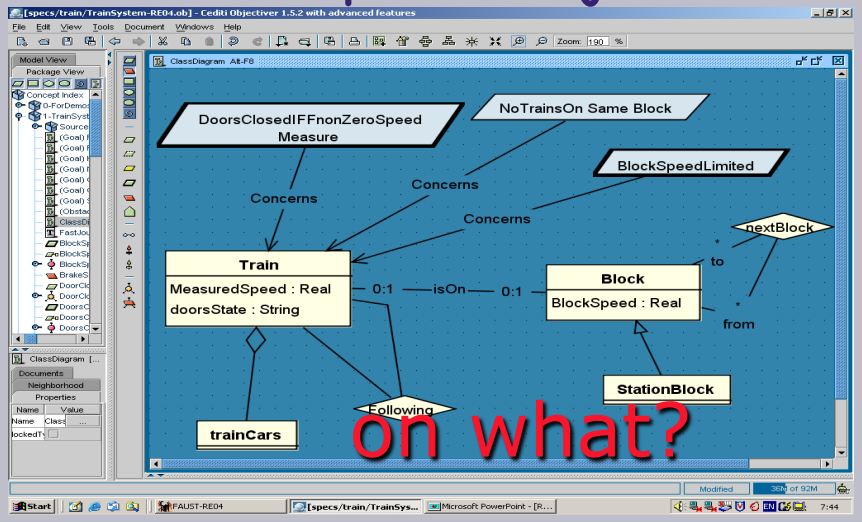
Goals



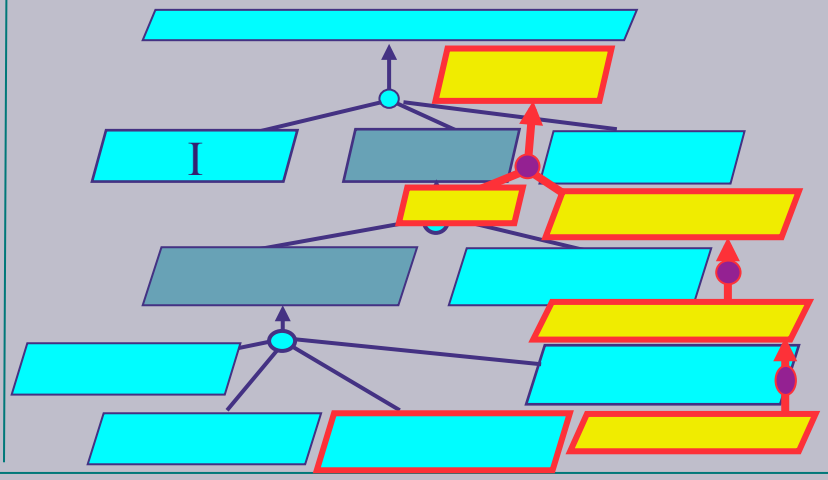
Risks



Conceptual objects

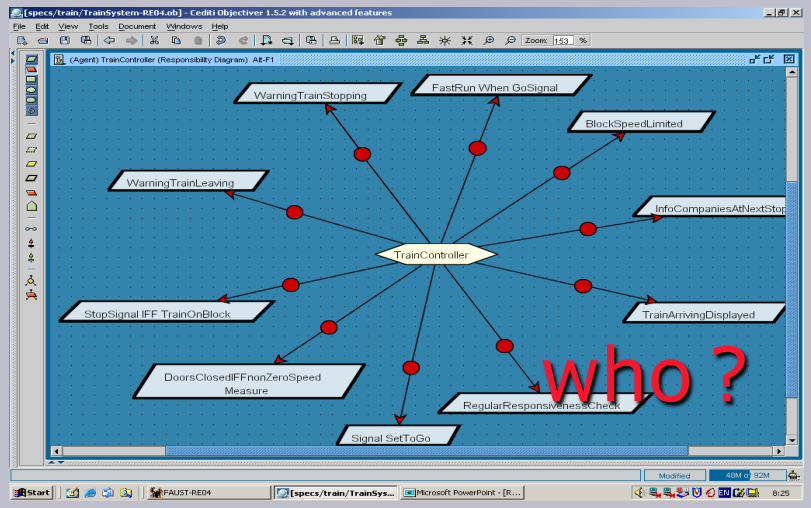


Threats

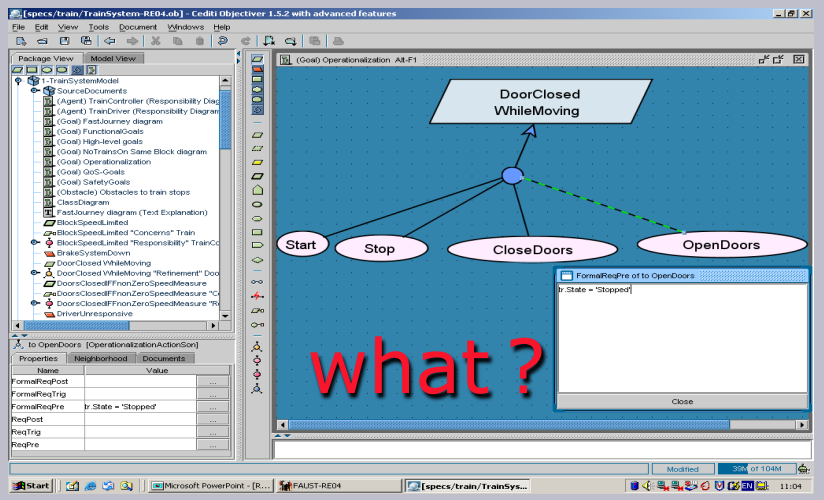


What models for RE ? (2)

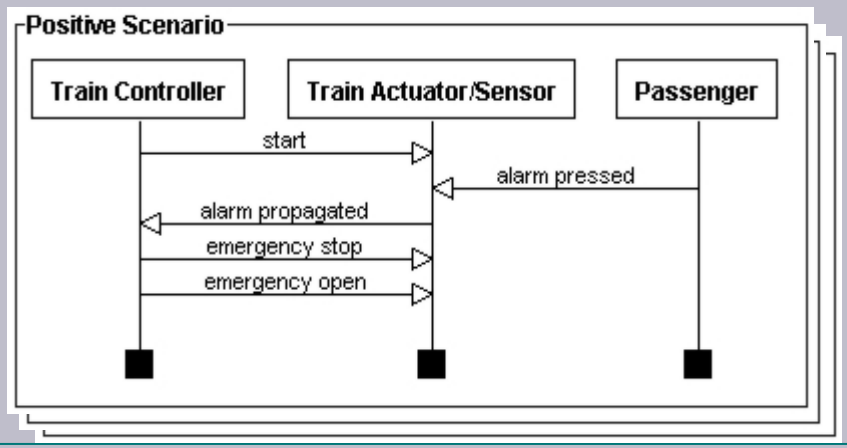
Agents



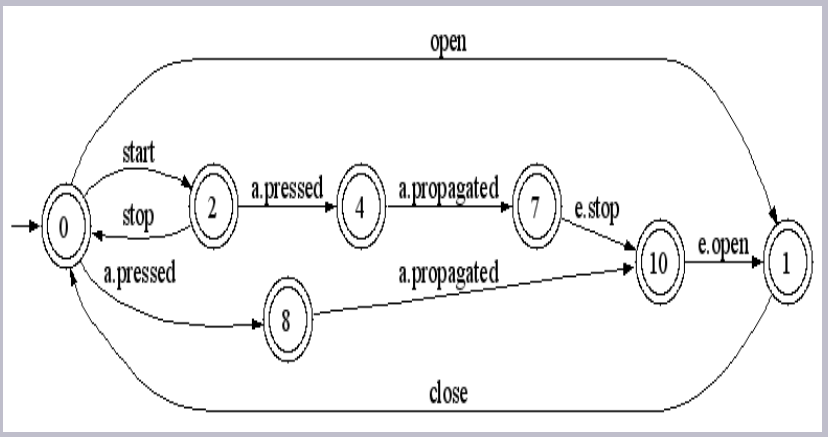
Operations



Interaction scenarios



Behaviors



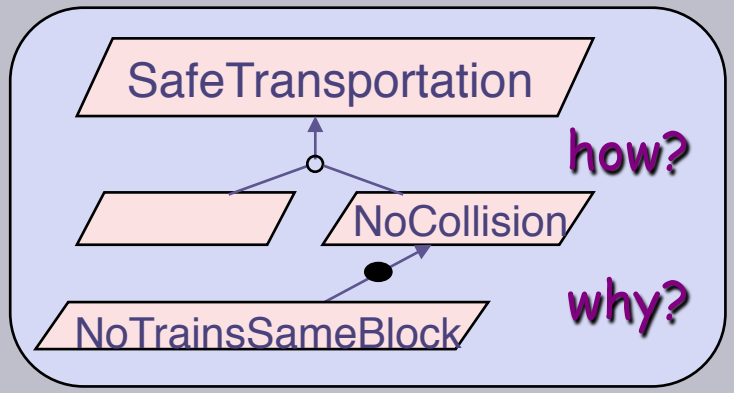
Outline

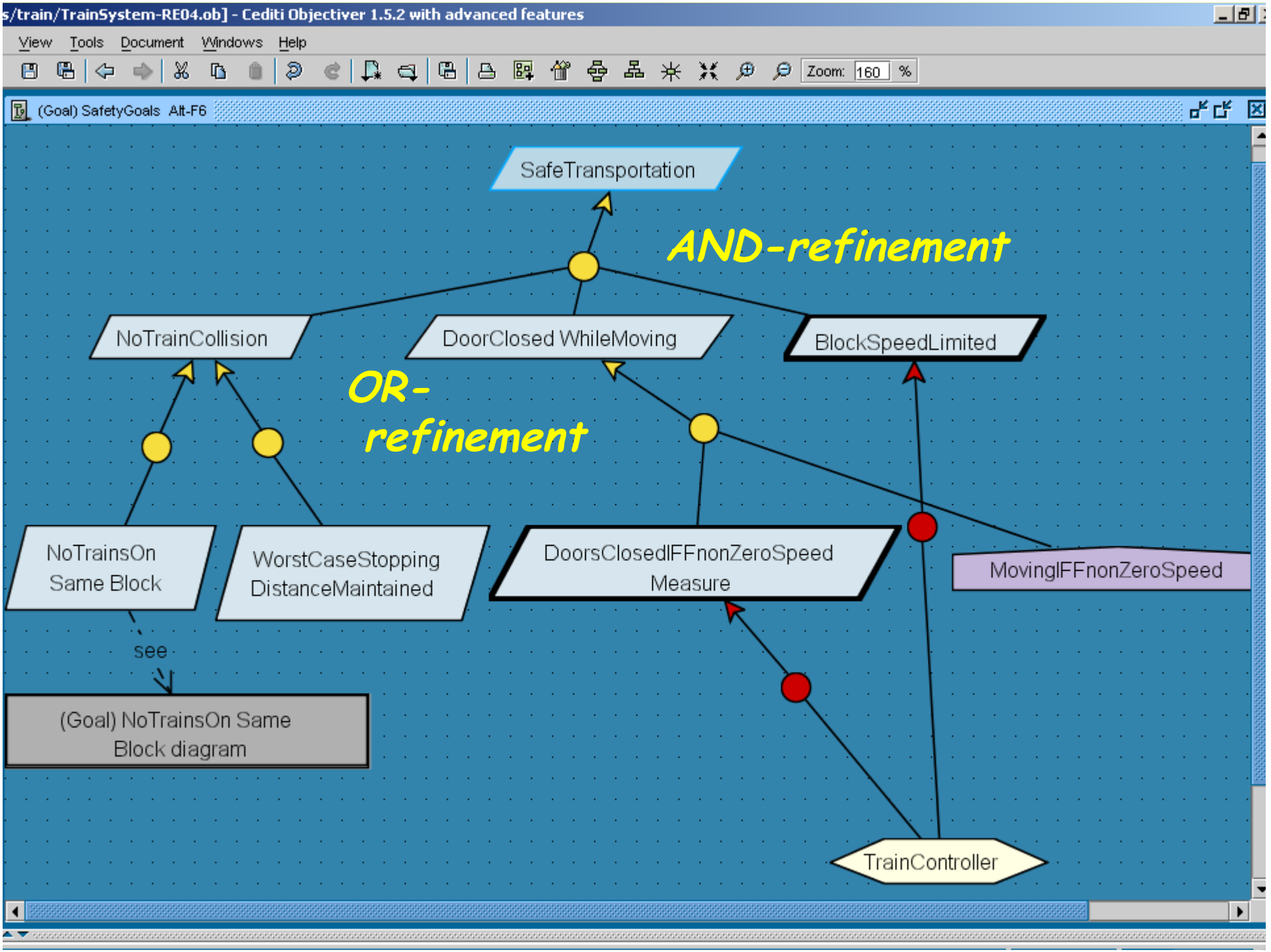
- Requirements engineering (RE)
  - What it is
  - Why it is difficult
  - Why it is important
- Models for RE
  - Why model, model what?

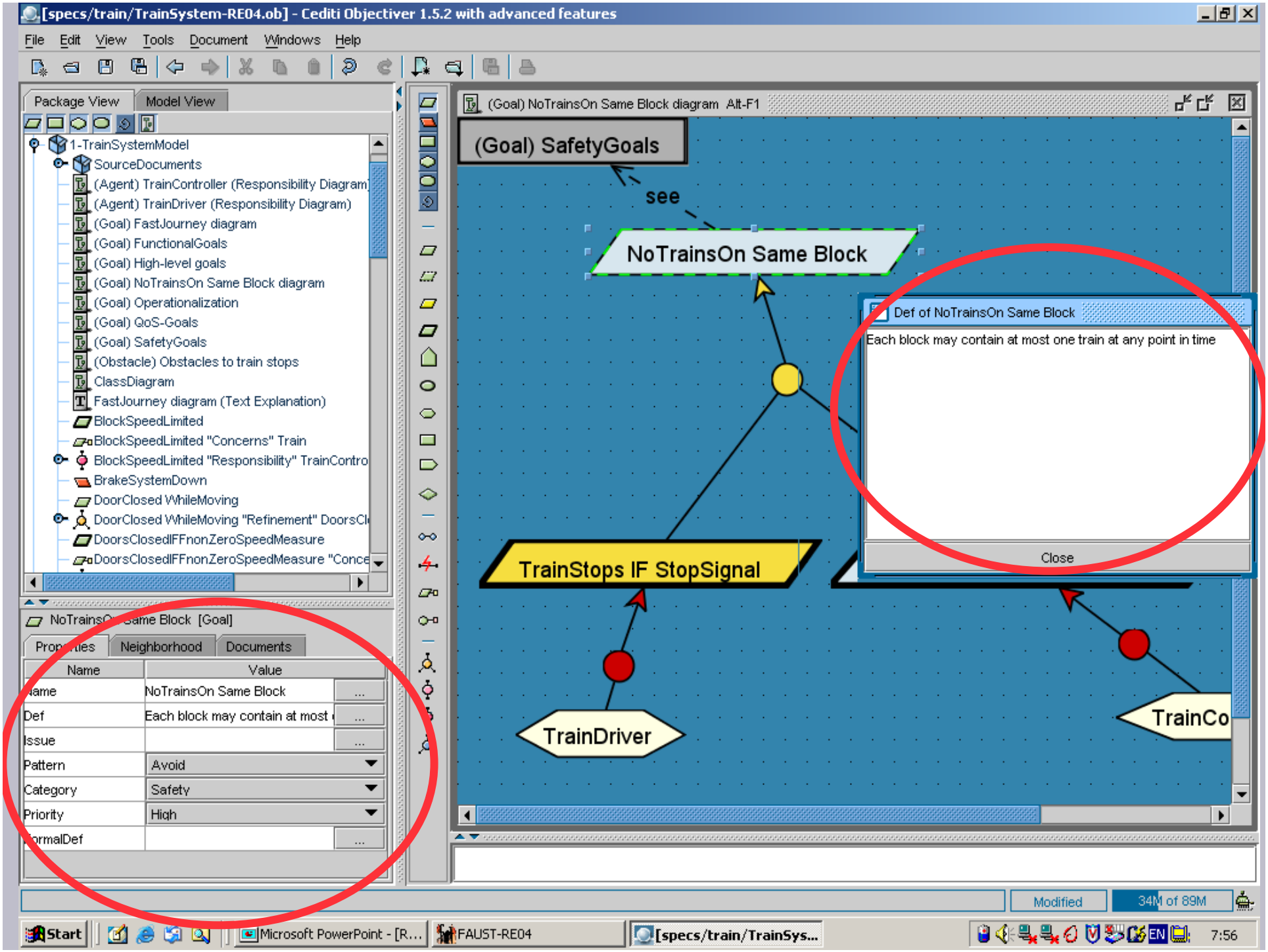
- Goal-oriented RE
  - Goal-based model building
  - Goal-oriented model analysis for RE

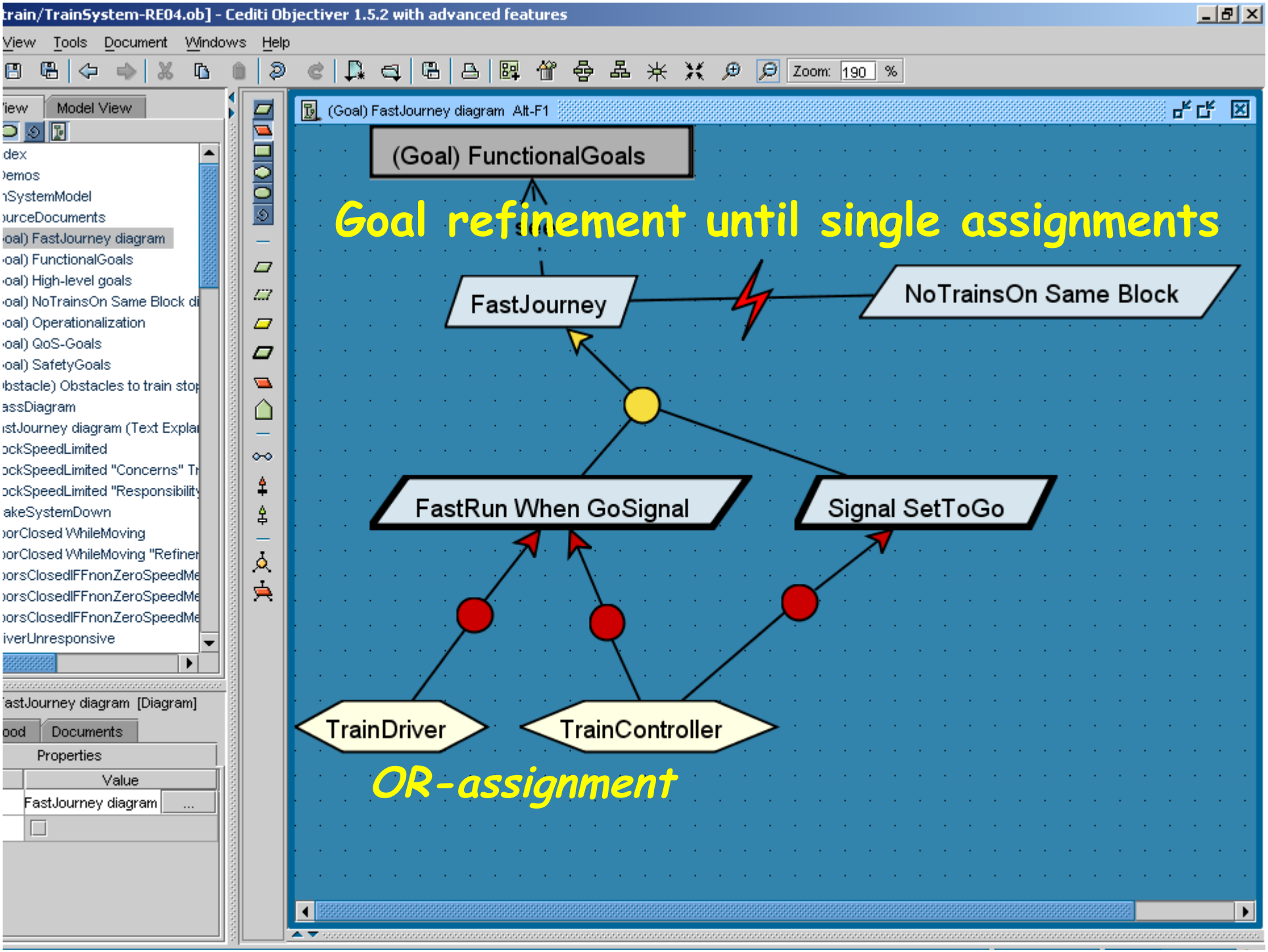
Goal-oriented model building

1. Domain analysis: refine/abstract goals



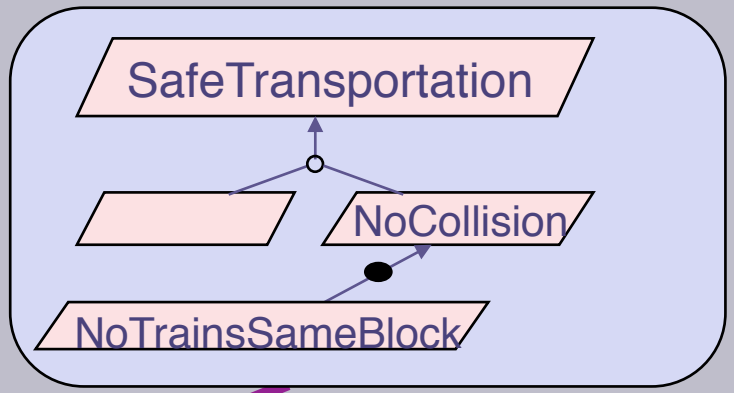

s/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
View Tools Document Windows Help
Zoom: 160 %
(Goal) SafetyGoals Alt-F6
SafeTransportation
AND-refinement
NoTrainCollision
DoorClosed WhileMoving
BlockSpeedLimited
OR-refinement
NoTrainsOn Same Block
WorstCaseStopping DistanceMaintained
DoorsClosedIFFnonZeroSpeed Measure
MovingIFFnonZeroSpeed
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...
7:56

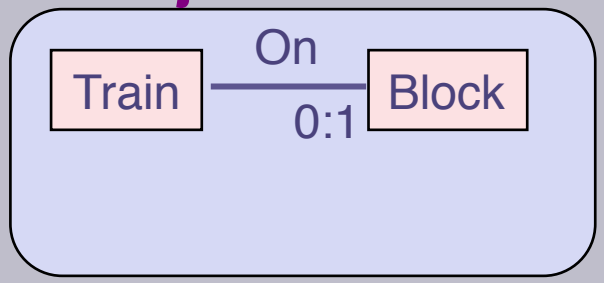

train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
View Tools Document Windows Help
Zoom: 190 %
Model View
(Goal) FastJourney diagram Alt-F1
(Goal) FunctionalGoals
Goal refinement until single assignments
FastJourney
NoTrainsOn Same Block
FastRun When GoSignal
Signal SetToGo
TrainDriver
TrainController
OR-assignment
Properties
Value
FastJourney diagram

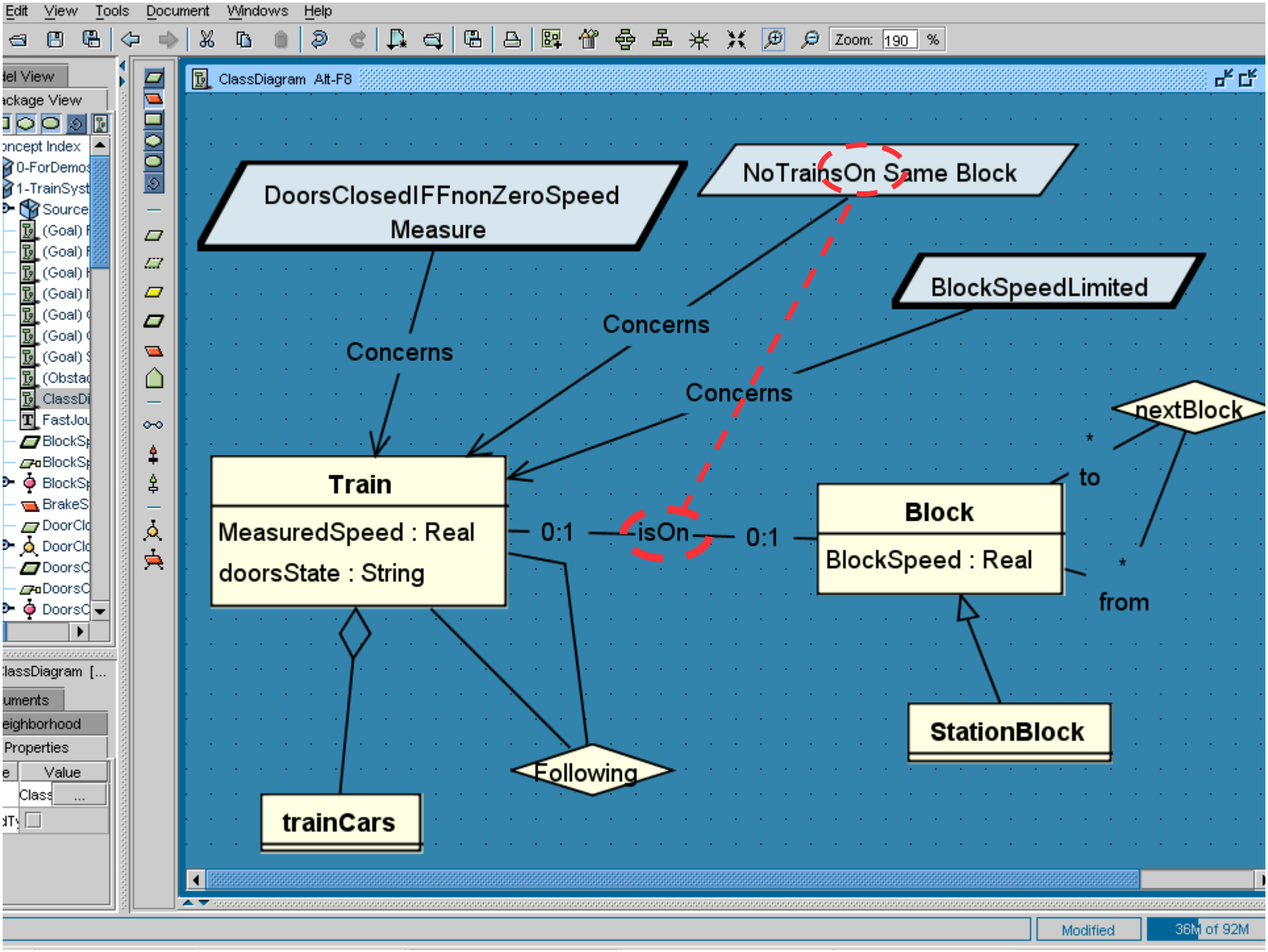
Goal-oriented model building

1. Domain analysis: refine/abstract goals



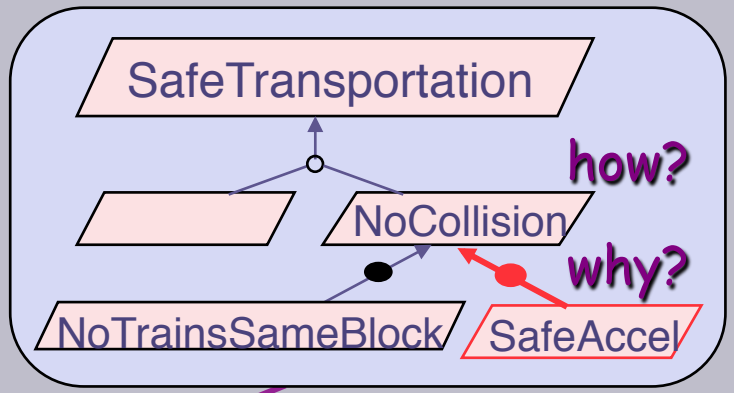
2. Domain analysis: derive/structure objects




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
36M of 92M

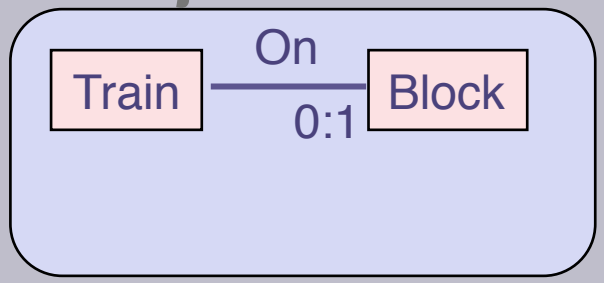
Goal-oriented model building

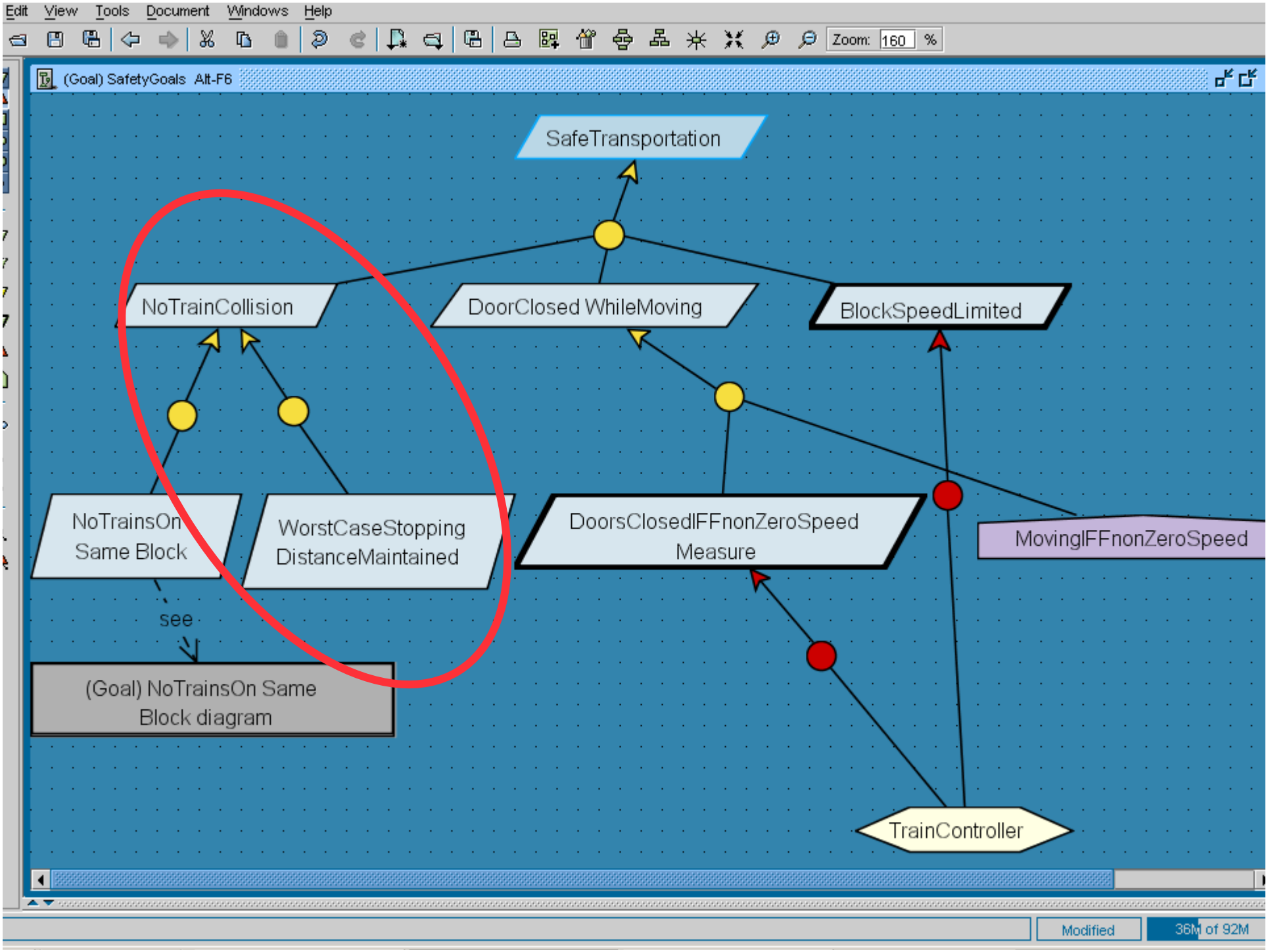
1. Domain analysis: refine/abstract goals



3. System-to-be: enriched goals (alternatives)

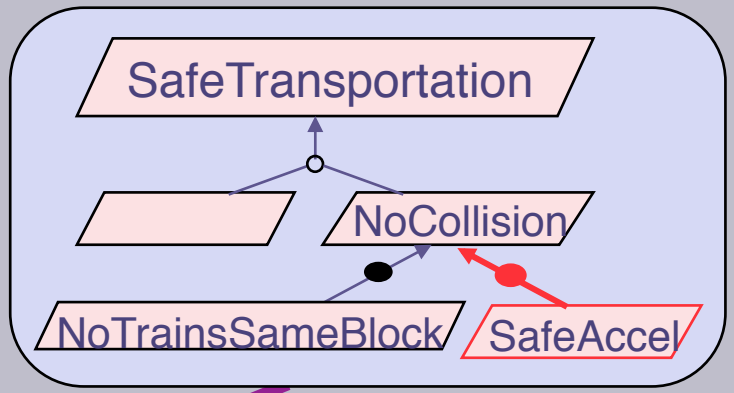
2. Domain analysis: derive/structure objects




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

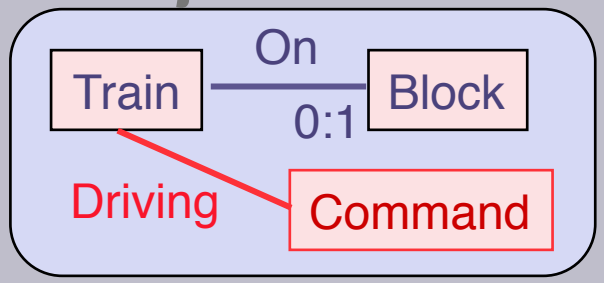
Goal-oriented model building

1. Domain analysis: refine/abstract goals



3. S2B analysis: enriched goals (alternatives)

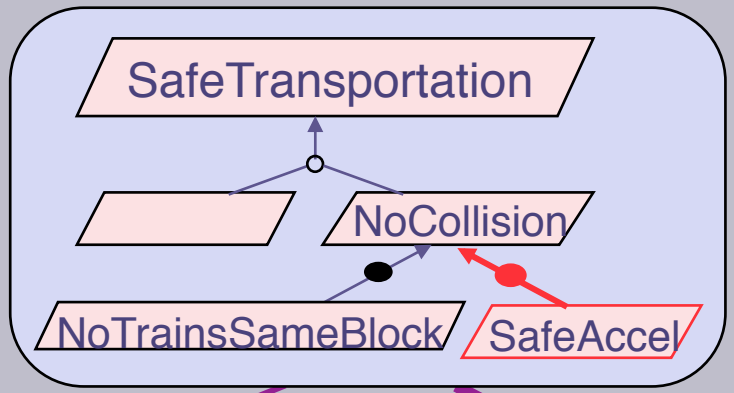
2. Domain analysis: derive/structure objects



4. System-to-be: enriched objects from new goals

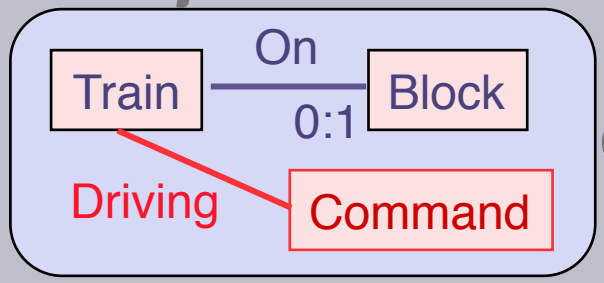
Goal-oriented model building

1. Domain analysis: refine/abstract goals

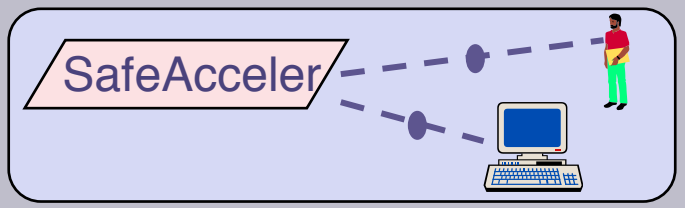


3. S2B analysis: enriched goals (alternatives)

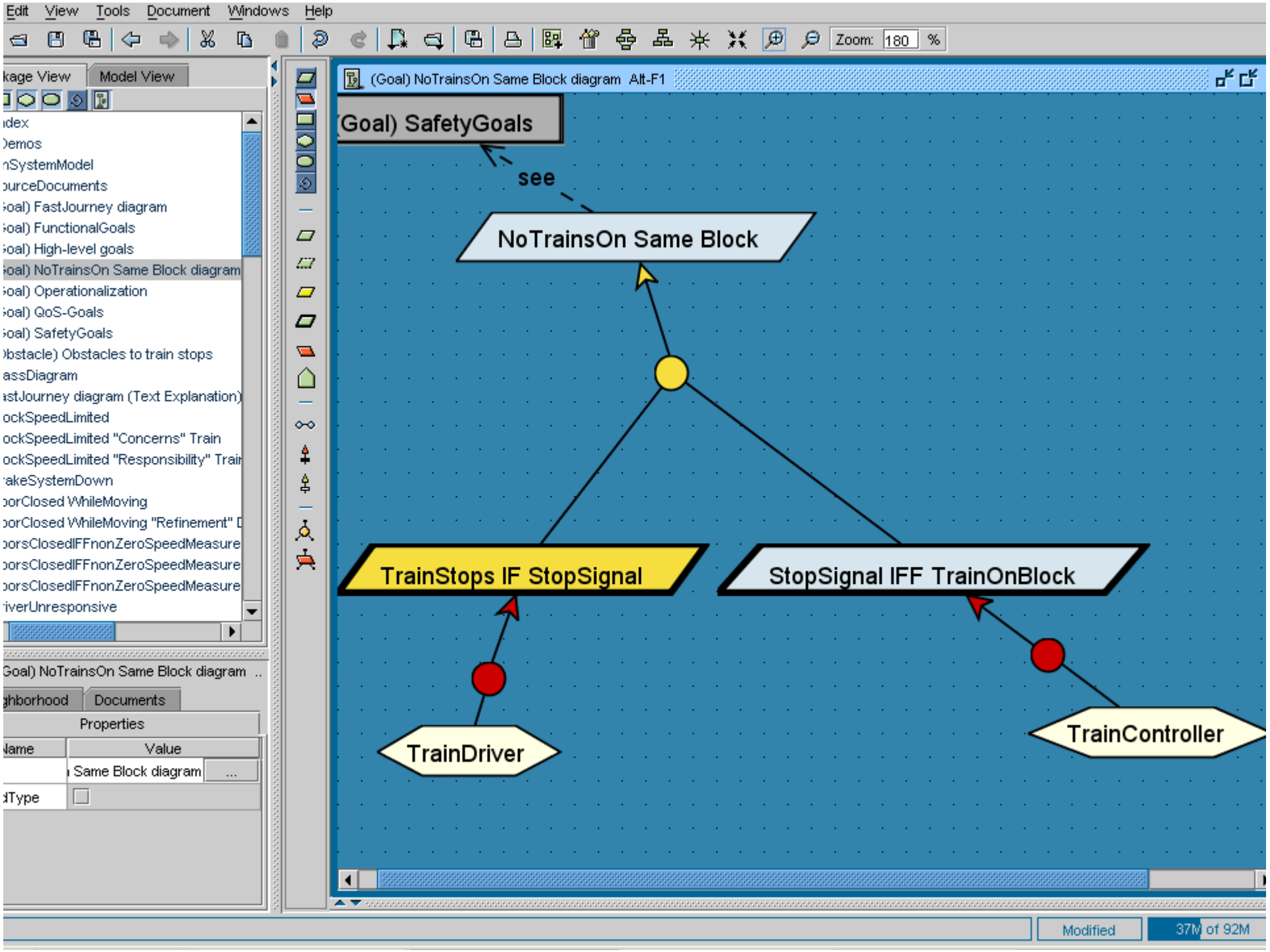
2. Domain analysis: derive/structure objects



4. S2B analysis: enriched objects from new goals

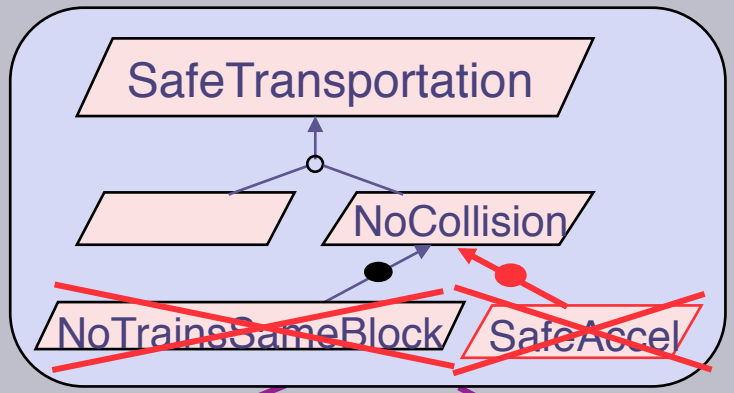


5. Responsibility analysis: agent assignment


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

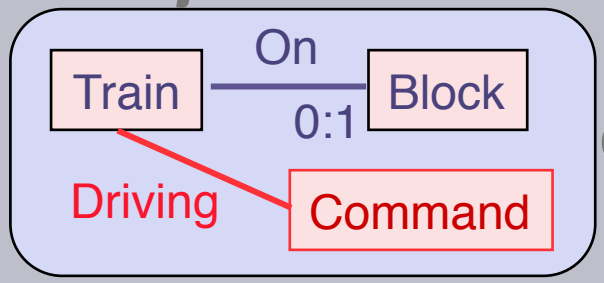
Goal-oriented model building

1. Domain analysis: refine/abstract goals



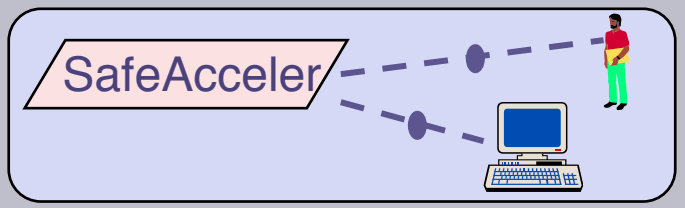
3. S2B analysis: enriched goals (alternatives)

2. Domain analysis: derive/structure objects

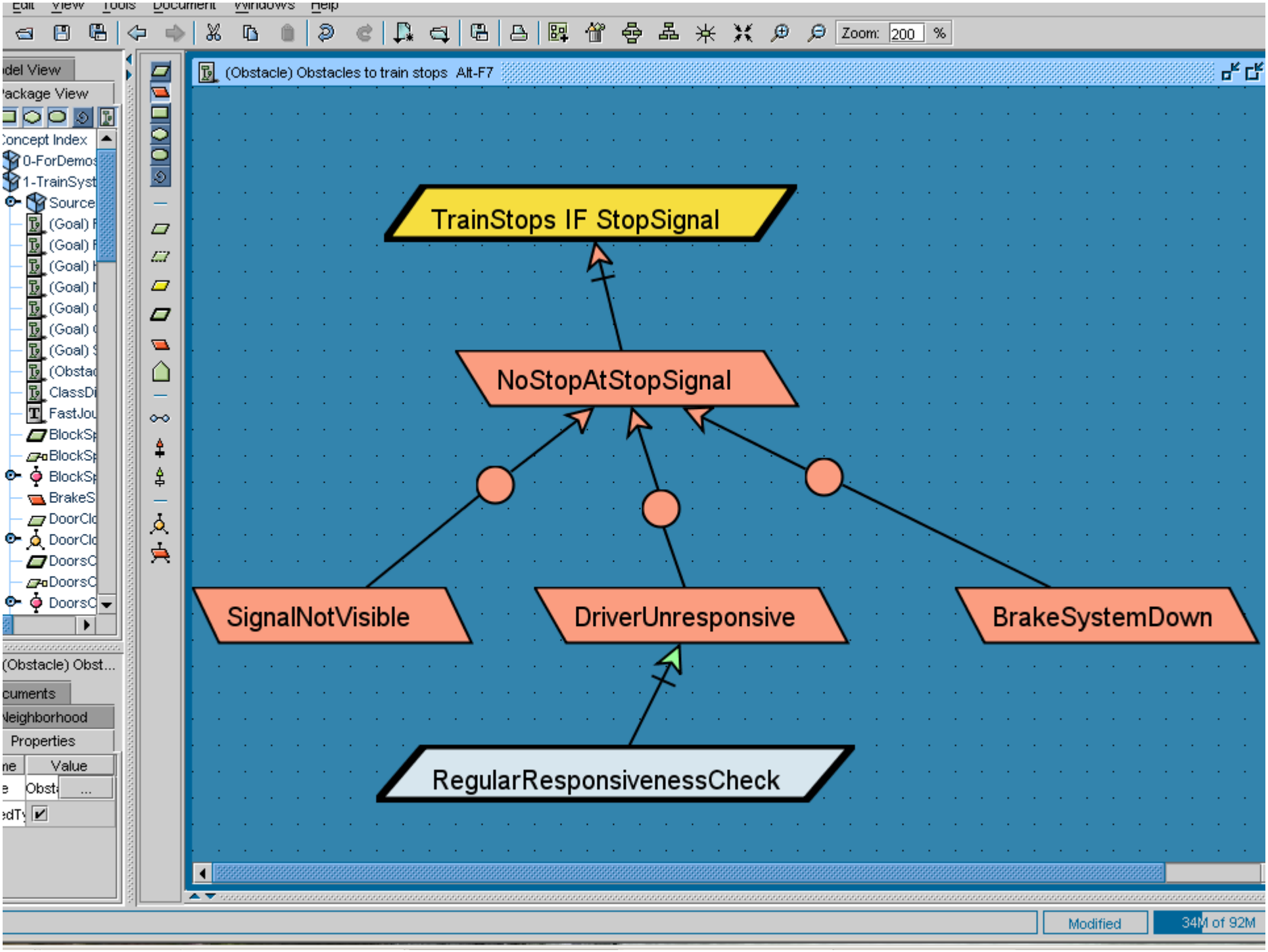


4. S2B analysis: enriched objects from new goals

1-5 // Risk & conflict analysis

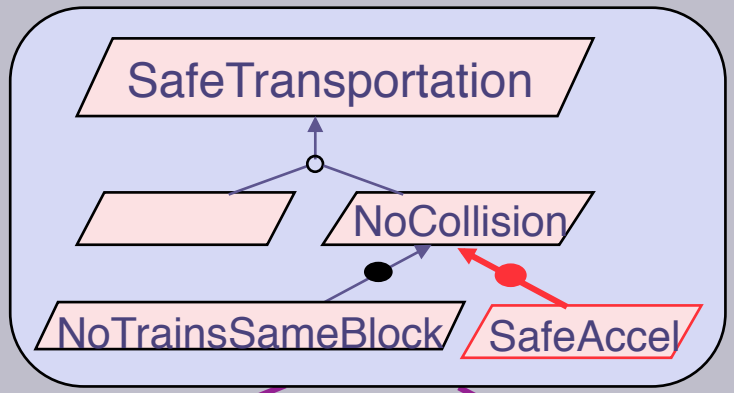


5. Responsibility analysis: agent assignment


(Obstacle) Obstacles to train stops Alt-F7
TrainStops IF StopSignal
NoStopAtStopSignal
SignalNotVisible
DriverUnresponsive
BrakeSystemDown
RegularResponsivenessCheck
Zoom: 200 %
Modified
34M of 92M

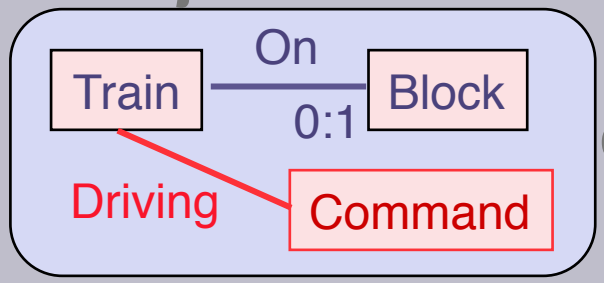
Goal-oriented model building

1. Domain analysis: refine/abstract goals



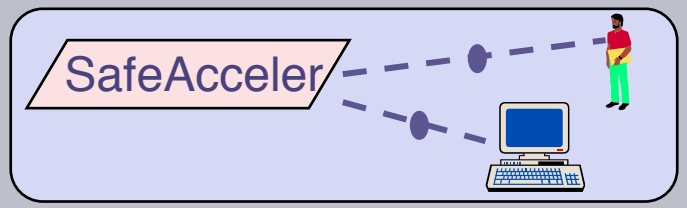
3. S2B analysis: enriched goals (alternatives)

2. Domain analysis: derive/structure objects

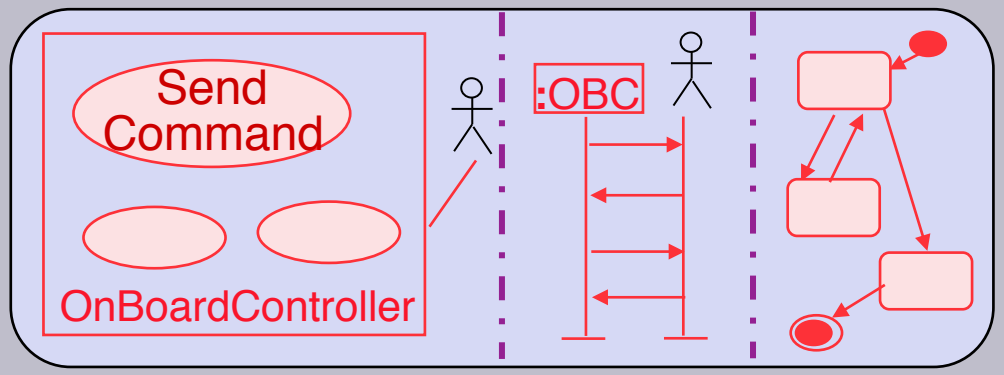


4. S2B analysis: enriched objects from new goals

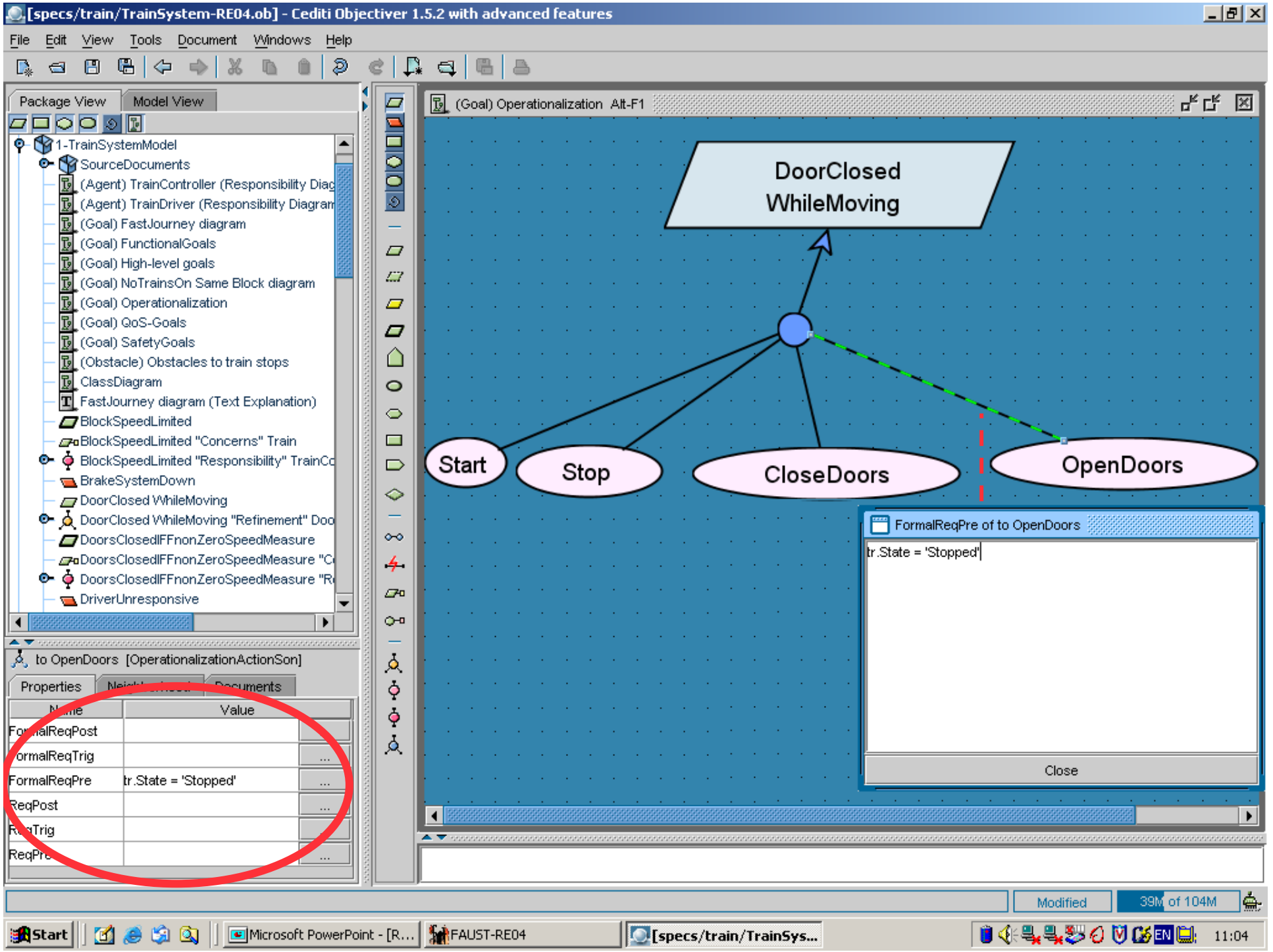
1-5 // Risk & conflict analysis

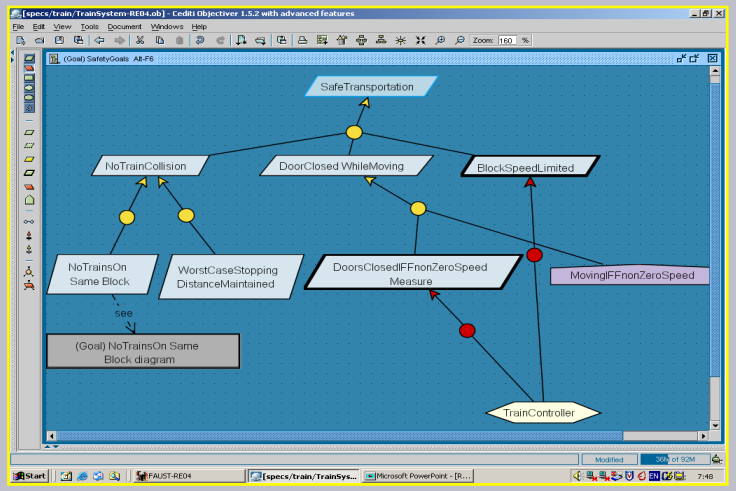


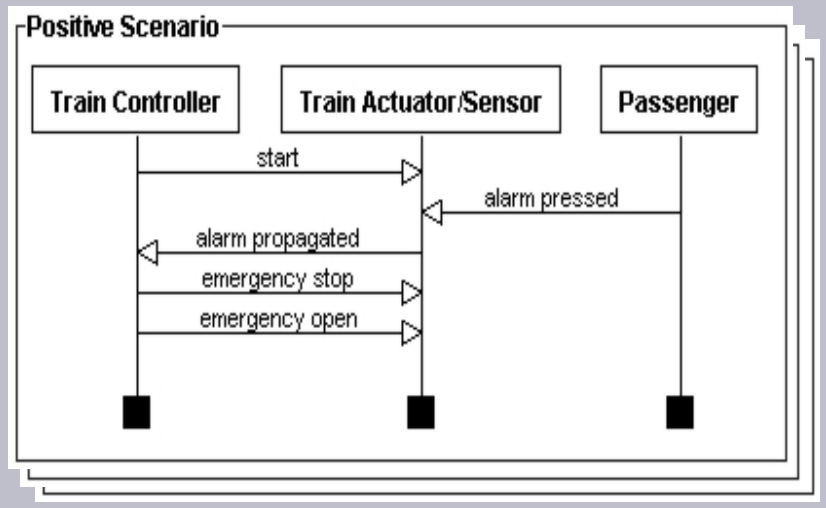
5. Responsibility analysis: agent assignment



6. Operationalization & behavior analysis

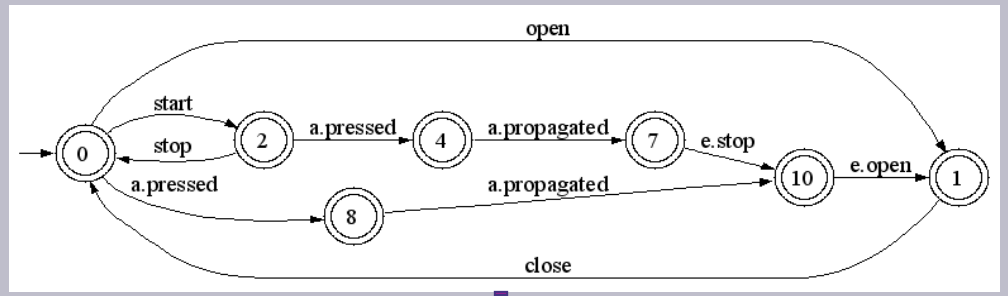

[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Package View
Model View
1-TrainSystemModel
SourceDocuments
(Agent) TrainController (Responsibility Diag
(Agent) TrainDriver (Responsibility Diagram
(Goal) FastJourney diagram
(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
to OpenDoors [OperationalizationActionSon]
Properties
Documents
Value
FormalReqPre
tr.State = 'Stopped'
(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...
11:04



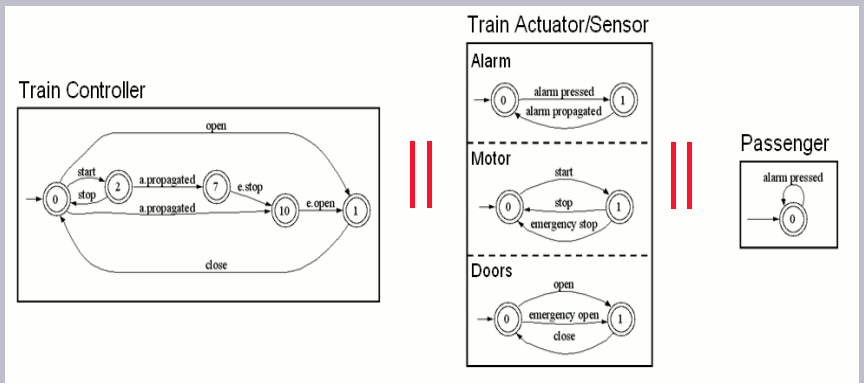


Behavior model synthesis

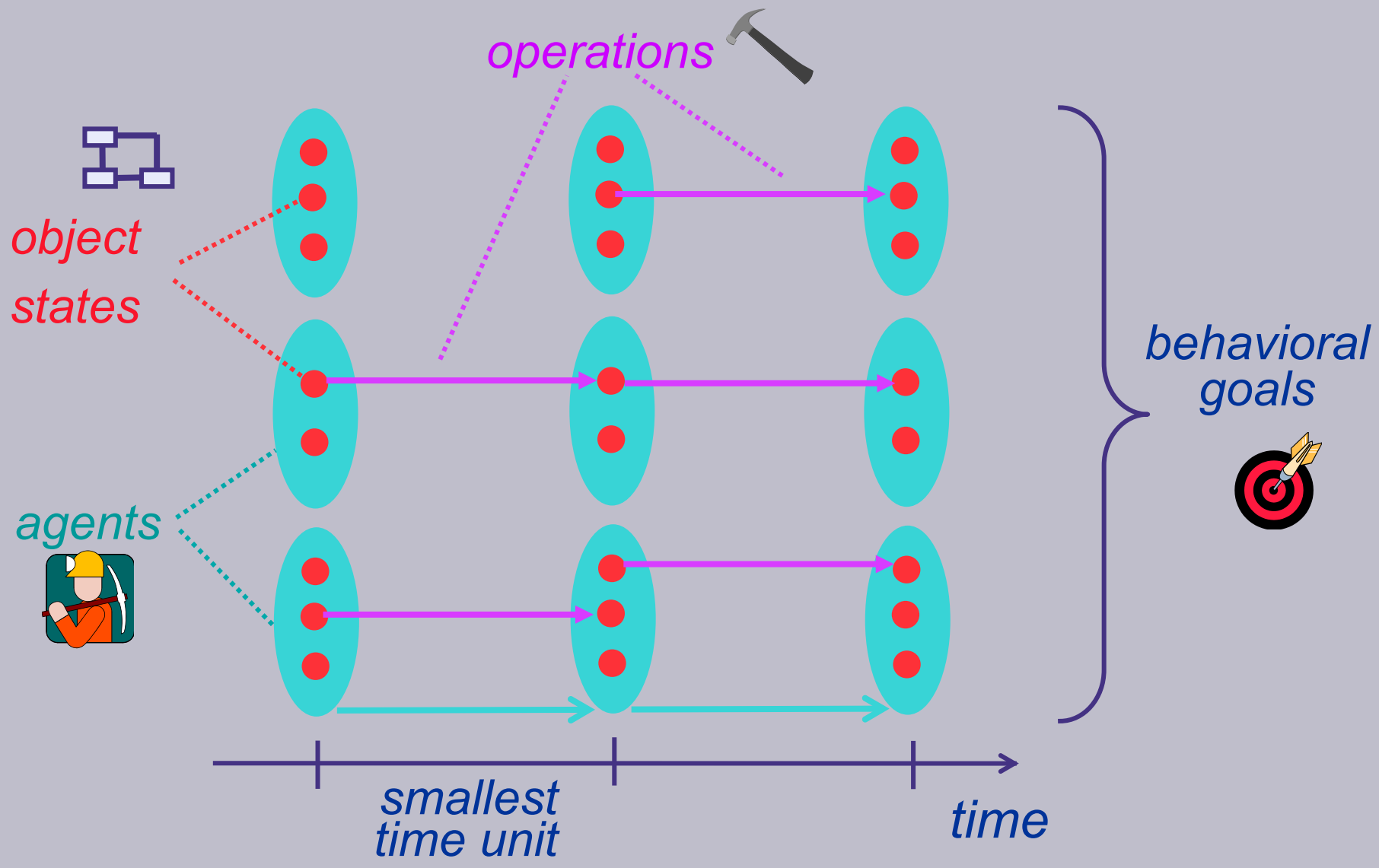
learn



decompose



Goals, objects, agents, operations: the semantic picture



Outline

- Requirements engineering (RE)
  - What it is
  - Why it is difficult
  - Why it is important
- Models for RE
  - Why model, model what?
- Goal-oriented RE
  - Goal-based model building
  - Goal-oriented model analysis for RE

Checking goal refinements

- Aim: show that refinements are correct & complete

  Subgoals, Assumptions, DomainProps |– ParentGoal

- (Approach 1: use theorem prover)

  heavyweight, non-constructive

- Approach 2: use refinement patterns

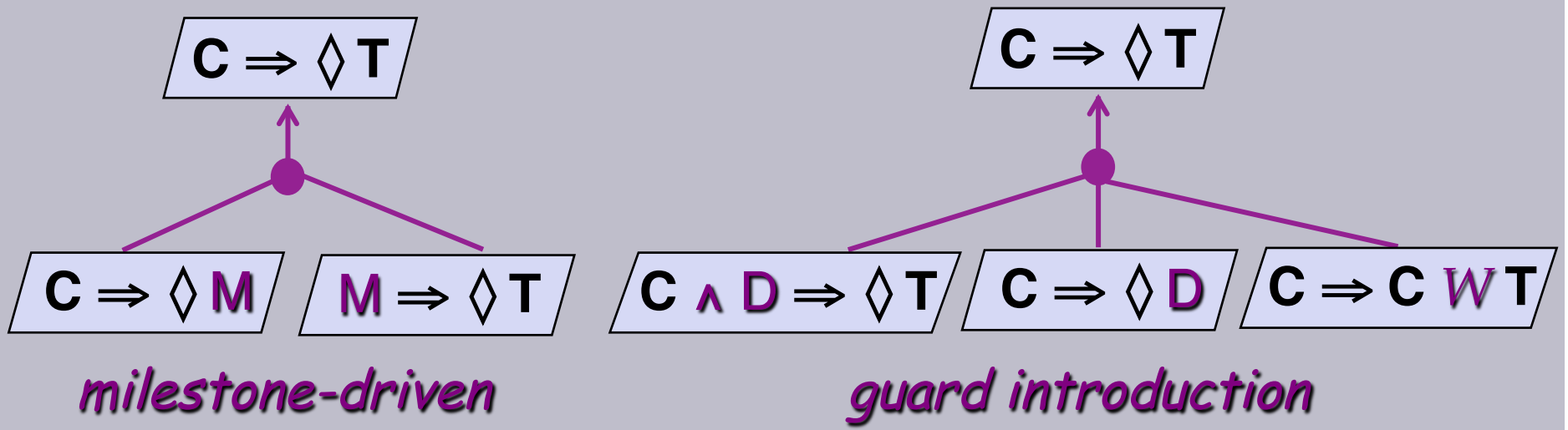
  lightweight, constructive:

  to *guide*, *complete*, *explore* refinements

Checking goal refinements with patterns

- Catalogue of patterns encoding refinement tactics
- Generic refinements proved formally, once for all
- Reuse through instantiation, in matching situation

Can be used informally (NL templates)



milestone-driven

guard introduction

Checking goal refinements with patterns (2)

Achieve [TrainProgress]
On (tr, b) ⇒ ◊ On (tr, next(b))

missing subgoal !!
detectable automatically

Achieve [ProgressWhenGo]
On (tr, b) ∧ Go [next(b)]
⇒ ◊ On (tr, next(b))

Achieve [SignalSetToGo]
On (tr, b) ⇒ ◊ Go [next(b)]

Checking goal refinements with patterns (2)

Achieve [TrainProgress]
$On(tr, b) \Rightarrow \Diamond On(tr, next(b))$

guard introduction

Achieve [ProgressWhenGo]
$On(tr, b) \wedge Go[next(b)] \Rightarrow \Diamond On(tr, next(b))$

Achieve [SignalSetToGo]
$On(tr, b) \Rightarrow \Diamond Go[next(b)]$

Maintain [TrainWaiting]
$On(tr, b) \Rightarrow On(tr, b)\ \mathcal{W}\ On(tr, next(b))$

mathematical proof hidden, reusable

Checking goal refinements

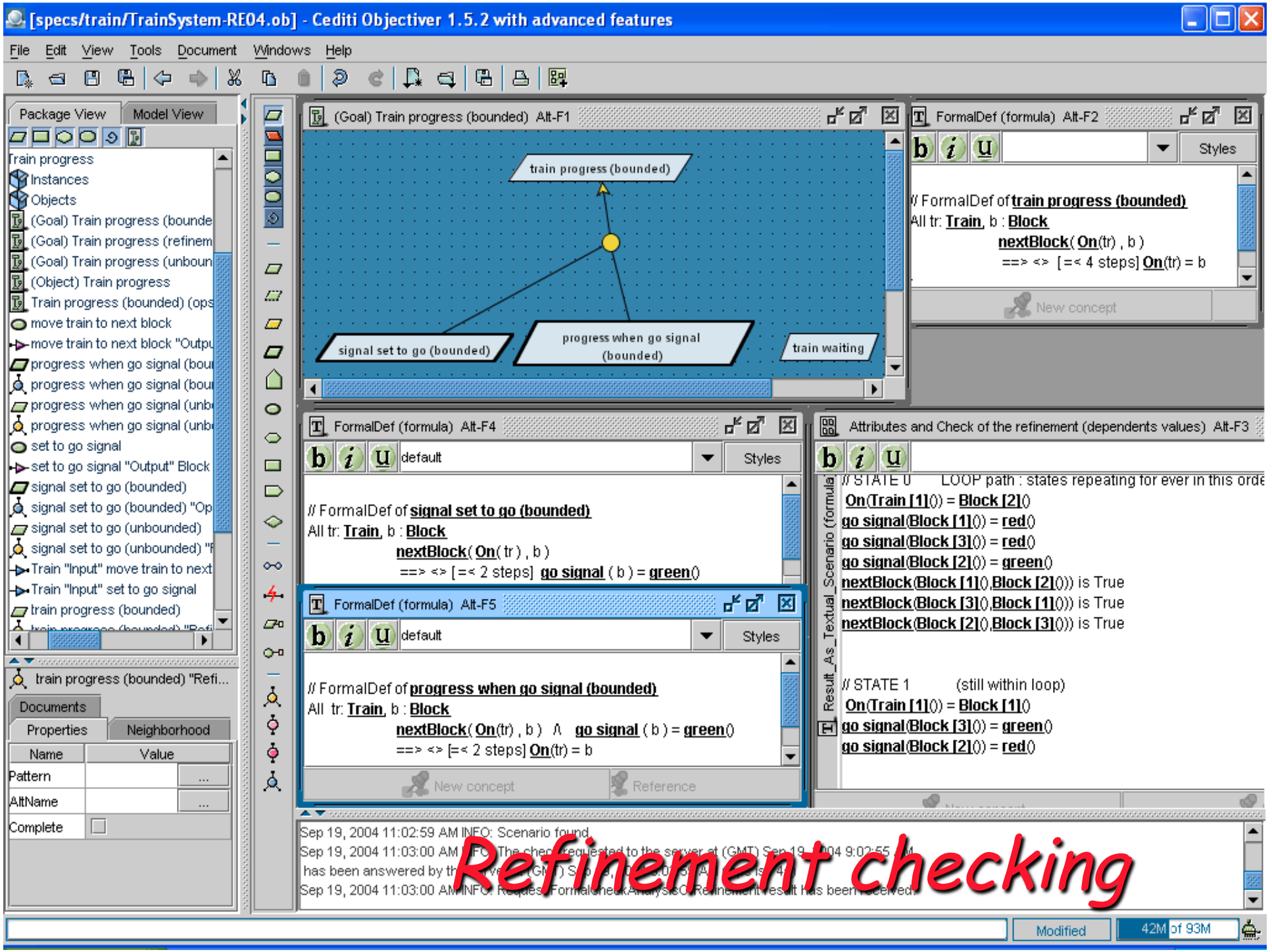
- Approach 3: front end to bounded SAT solver
  - Incremental check/debug of goal model fragments
  - On selected object instances (propositionalization)

Input: $SubG_1 \wedge \ldots \wedge SubG_n \wedge Dom \wedge \neg ParentGoal$

Ouput:

OK

KO + counter-example scenario


[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Package View
Model View
Train progress
Instances
Objects
(Goal) Train progress (bounde
(Goal) Train progress (refinem
(Goal) Train progress (unboun
(Object) Train progress
Train progress (bounded) (ops
move train to next block
move train to next block "Outpu
progress when go signal (bou
progress when go signal (bou
progress when go signal (unb
progress when go signal (unb
set to go signal
set to go signal "Output" Block
signal set to go (bounded)
signal set to go (bounded) "Op
signal set to go (unbounded)
signal set to go (unbounded) "F
Train "Input" move train to next
Train "Input" set to go signal
train progress (bounded)
train progress (bounded) "Refi...
Documents
Properties
Neighborhood
Name
Value
Pattern
AltName
Complete
(Goal) Train progress (bounded) Alt-F1
train progress (bounded)
signal set to go (bounded)
progress when go signal (bounded)
train waiting
FormalDef (formula) Alt-F2
Styles
// FormalDef of train progress (bounded)
All tr: Train, b : Block
nextBlock(On(tr) , b)
==> <> [=< 4 steps] On(tr) = b
New concept
FormalDef (formula) Alt-F4
default
Styles
// FormalDef of signal set to go (bounded)
All tr: Train, b : Block
nextBlock(On(tr) , b)
==> <> [=< 2 steps] go signal (b) = green()
FormalDef (formula) Alt-F5
default
Styles
// FormalDef of progress when go signal (bounded)
All tr: Train, b : Block
nextBlock(On(tr) , b) /\ go signal (b) = green()
==> <> [=< 2 steps] On(tr) = b
New concept
Reference
Attributes and Check of the refinement (dependents values) Alt-F3
Result_As_Textual_Scenario (formula)
// STATE 0 LOOP path : states repeating for ever in this orde
On(Train [1]()) = Block [2]()
go signal(Block [1]()) = red()
go signal(Block [3]()) = red()
go signal(Block [2]()) = green()
nextBlock(Block [1](),Block [2]())) is True
nextBlock(Block [3](),Block [1]())) is True
nextBlock(Block [2](),Block [3]())) is True
// STATE 1 (still within loop)
On(Train [1]()) = Block [1]()
go signal(Block [3]()) = green()
go signal(Block [2]()) = red()
Sep 19, 2004 11:02:59 AM INFO: Scenario found.
Sep 19, 2004 11:03:00 AM INFO: The check requested to the server at (GMT) Sep 19, 2004 9:02:55 AM
has been answered by the server (GMT) Sep 19, 2004 9:03:00 AM
Sep 19, 2004 11:03:00 AM INFO: Request FormalCheckAnalysisOfRefinement result has been received.
Refinement checking
Modified
42M of 93M

From declarative to operational models

- Derive *operations* or *tasks* through operationalization patterns

proved correct once for all

G: $C \Rightarrow \circ T$

Operation Op1
- DomPre $\neg T$
- DomPost T
- ReqTrig for G: C

Operation Op2
- DomPre T
- DomPost $\neg T$
- ReqPre for G: $\neg C$

- ... or check them with bounded SAT solver:

$[|\ ReqCond_1\ |] \wedge \ldots \wedge [|\ ReqCond_n\ |] \wedge Dom \wedge \neg Goal$

Operationalization pattern: example

HighWaterSignal = 'On' ⇒ o PumpSwitch = 'On'

C: HighWaterSignal = 'On'
T: PumpSwitch = 'On'

Operation Op1
- DomPre ¬ T
- DomPost T
- ReqTrig for *G*: *C*

Operation Op2
- DomPre T
- DomPost ¬ T
- ReqPre for *G*: ¬ *C*

Operationalization pattern: example

HighWaterSignal = 'On' ⇒ ○ PumpSwitch = 'On'

C: HighWaterSignal = 'On'
T: PumpSwitch = 'On'

Operation SwitchPump***On***
DomPre PumpSwitch ≠ On
DomPost PumpSwitch = On
ReqTrig for *RootGoal*
HighWaterSignal = 'On'

Operation SwitchPump***Off***
DomPre PumpSwitch = On
DomPost PumpSwitch ≠ On
ReqPre for *RootGoal*
HighWaterSignal ≠ 'On'

Outline

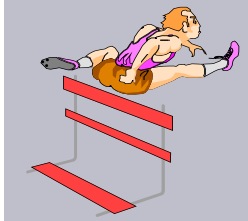
- Building models early: challenges
- Declarative abstractions for system modeling
- Goal-oriented model building
- Checking & deriving system models
  - Goal refinement, goal operationalization
  - Risk analysis (hazards, threats)
  - Behavior model synthesis

Requirements completeness is a major challenge

- Missing requirements = major cause of software failure
- Often result from **poor risk analysis**
  - lack of anticipation of what could go wrong

    => over-ideal system,

    no requirements on handling adverse events



Risk analysis for more robust system

- **Obstacle** = condition for goal obstruction

  $\{ O, Dom \} \models \neg G$ *obstruction*

  $Dom \not\models \neg O$ *domain consistency*

  (high-level exception)

Obstacle analysis

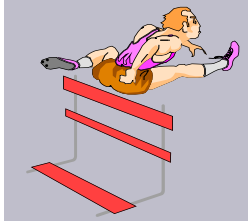
1. Identify obstacles (as many as possible)

   → formal calculus, obstruction patterns

2. Assess their likelihood and severity

3. Explore resolutions as new goals in the goal model

   → resolution operators (tactics)



Risk analysis for more robust system

- Obstacle = condition for goal obstruction

  $\{ O, Dom \} \models \neg G$ *obstruction*

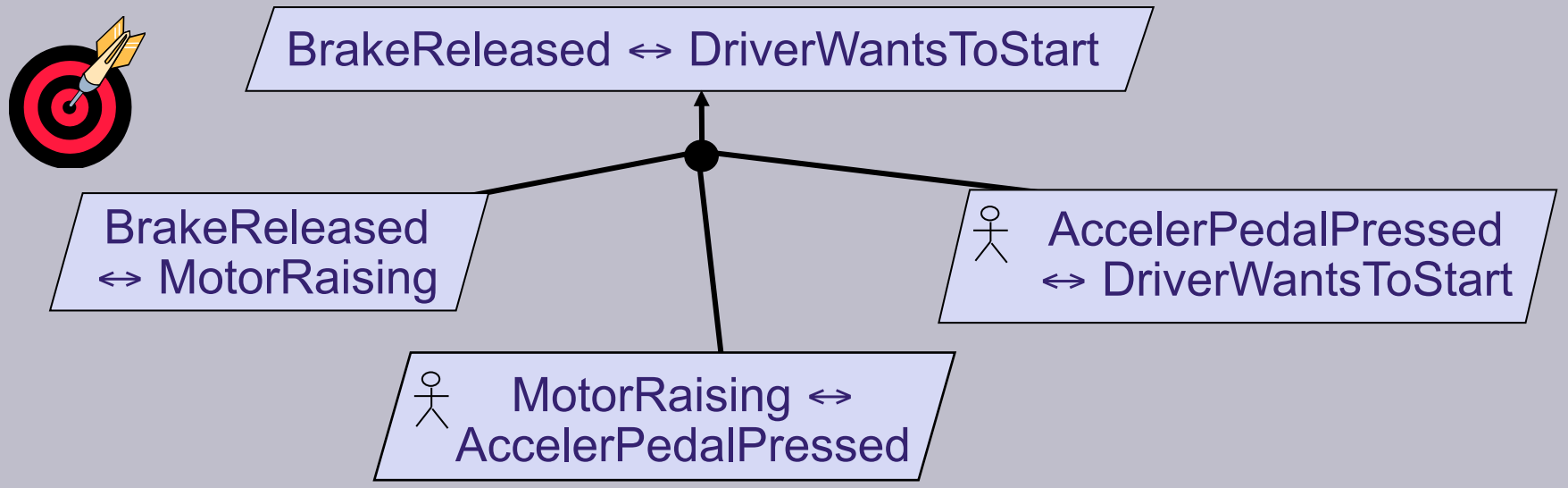
  $Dom \not\models \neg O$ *domain consistency*

  (high-level exception)

- **Obstacle analysis**

  1. **Identify** obstacles (as many as possible)
     → calculus *or* obstruction patterns *or* learning
  2. **Assess** their likelihood and severity
  3. Explore **resolutions** as new goals in the goal model
     → resolution operators (tactics)

Generating obstacles: obstacle trees as goal-anchored fault trees



Generating obstacles: obstacle trees as goal-anchored fault trees

BrakeReleased ↔ DriverWantsToStart

BrakeReleased ↔ MotorRaising

AccelerPedalPressed ↔ DriverWantsToStart

MotorRaising ↔ AccelerPedalPressed

...

...

MotorRaising **And Not** AccelerPedalPressed

AccelerPedalPressed **And Not** DriverWantsToStart

...

...

...

AirConditioningRaising

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

Threat analysis for more secure model

ItemOrderedByBuyer ⇒ $\Diamond_{\leq 7d}$ ItemReceivedByBuyer

ItemSent ⇒ ItemPaid

ItemOrdered ⇒ $\Diamond_{\leq 2d}$ ItemPaid

ItemPaid ⇒ $\Diamond_{\leq 2d}$ ItemSent

ItemSent ⇒ $\Diamond_{\leq 3d}$ ItemReceived

ShippingCo

ItemPaid ⇒ $\Diamond_{\leq 1d}$ $BELIEF_S$(ItemPaid)

$BELIEF_S$(ItemPaid) ⇒ $\Diamond_{\leq 1d}$ ItemSent

Seller

ItemPaid ⇒ $\Diamond_{\leq 8h}$ PaymentReceived

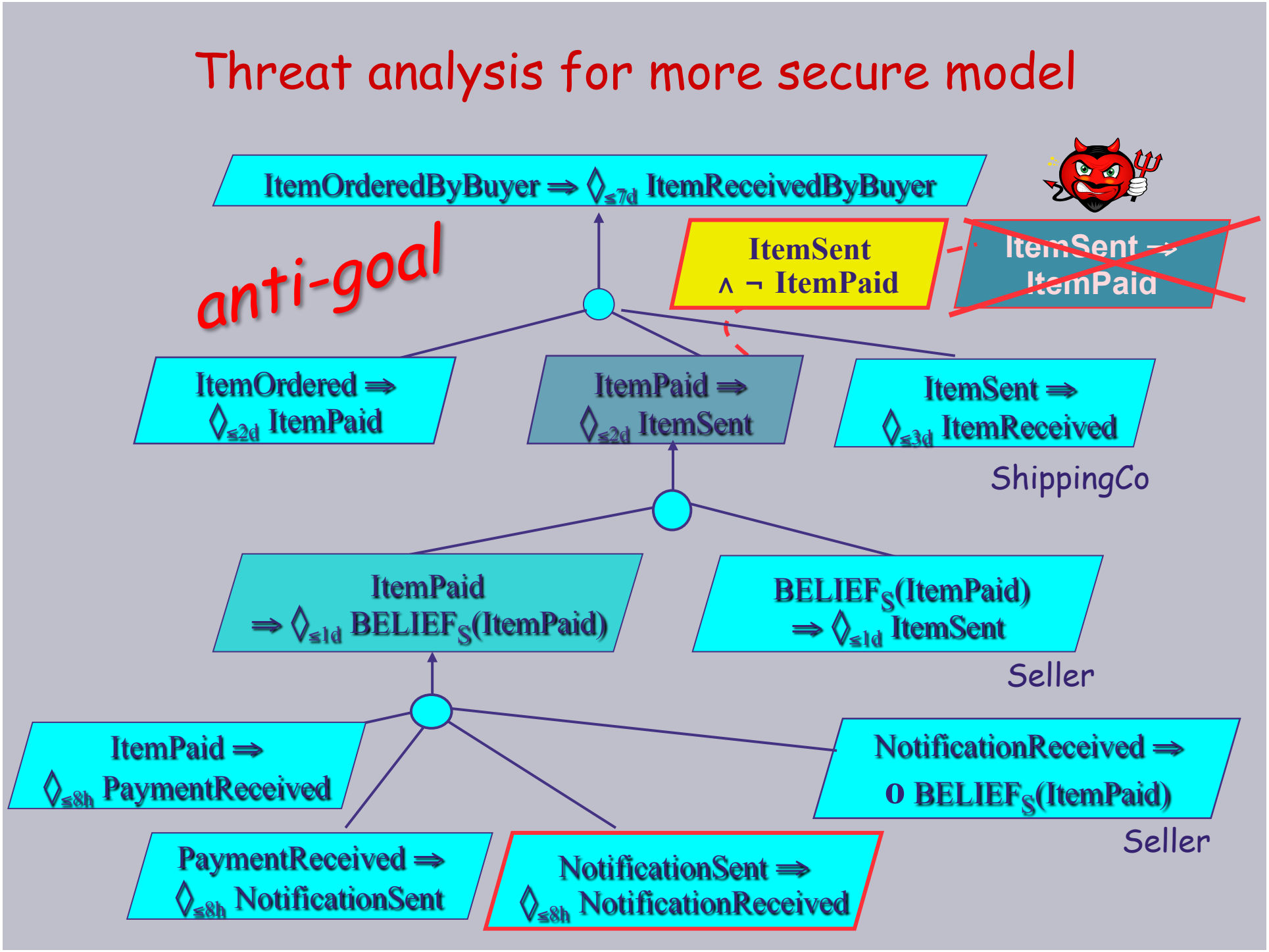
NotificationReceived ⇒ o $BELIEF_S$(ItemPaid)

Seller

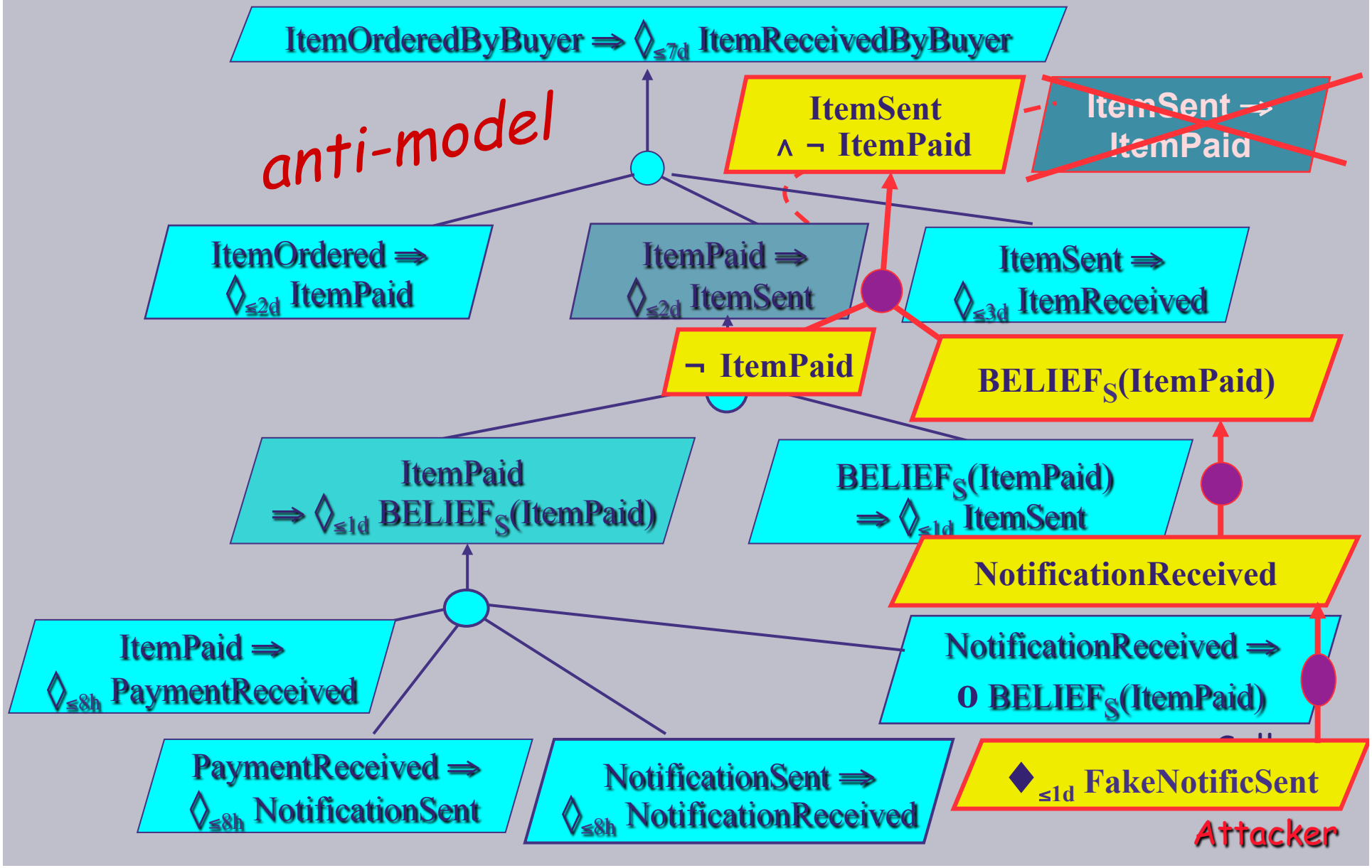
PaymentReceived ⇒ $\Diamond_{\leq 8h}$ NotificationSent

NotificationSent ⇒ $\Diamond_{\leq 8h}$ NotificationReceived

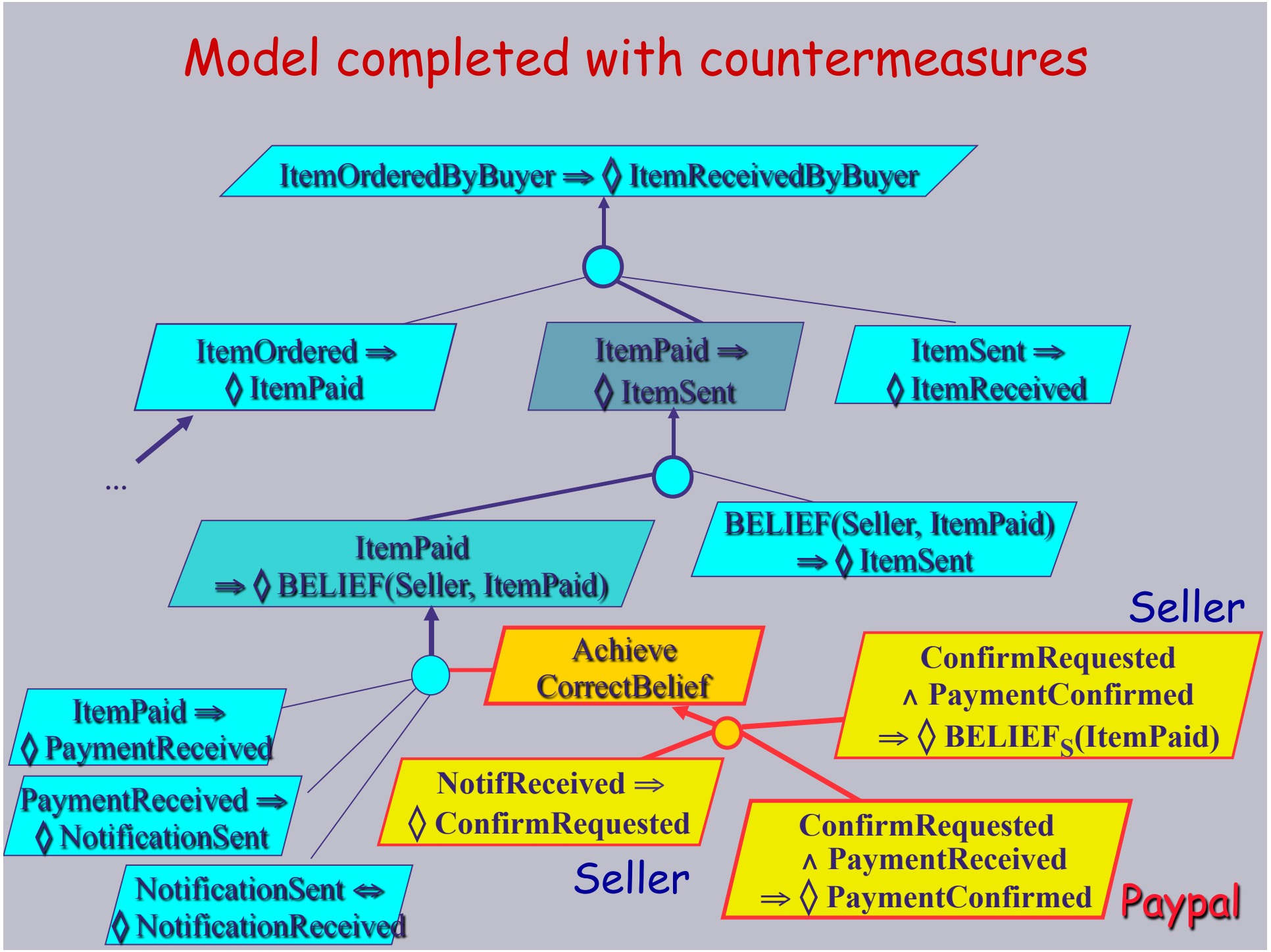
Threat analysis for more secure model



Threat analysis for more secure model



Model completed with countermeasures



Industrial application:

Security of Aircraft in the Future European Environment

Threats against crew & passengers

(External threats)

with Airbus, British Aerospace, SAGEM, Marconi, ...

Threats from baggage area

- Modeling terrorist threats (anti-goal model)
- For on-board detection & reaction system

Outline

- Building models early: challenges
- Declarative abstractions for system modeling
- Goal-oriented model building
- Checking & deriving system models
  - Goal refinement, goal operationalization
  - Risk analysis (hazards, threats)
  - Behavior model synthesis

State-based synthesis from operationalizations

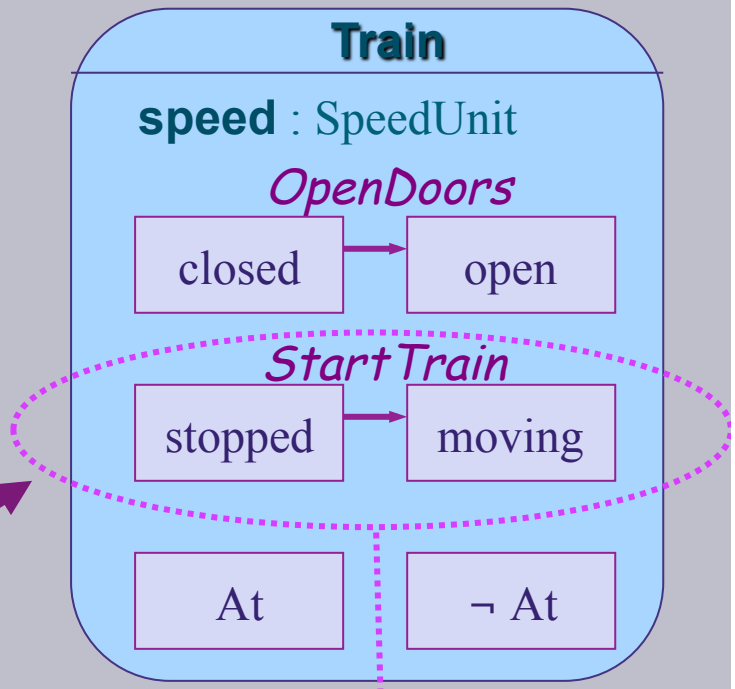
Goal Maintain[DoorsClosedWhileMoving]
  FormaDef ∀tr: Train, s: Station
    Moving (tr) ⇒ tr.Doors = 'closed'

Entity Train
  Has Speed: speedUnit

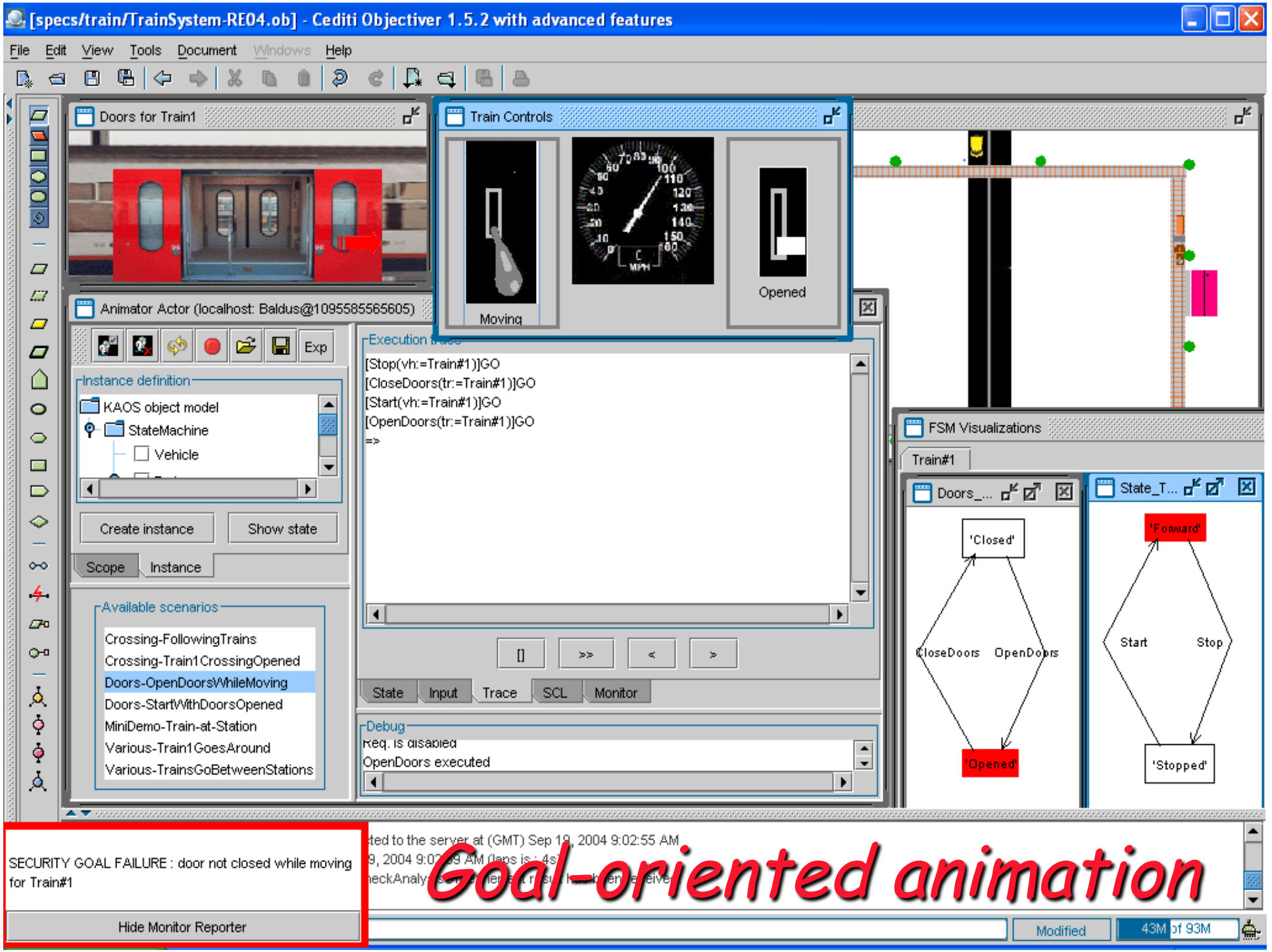
Relationship At
  Links Train {0:1}, Station {0:N}

Operation OpenDoors
  Input tr: Train; **Output** tr: Train
  DomPre tr.Doors = 'closed'
  DomPost tr.Doors = 'open'
  ReqPre for DoorsClosedWhileMoving
    ∃ s : Station At (tr,s)

Operation StartTrain
  Input tr: Train; **Output** tr: Train
  DomPre tr.Status = 'stopped'
  DomPost tr.Status = 'moving'
  ReqPre for DoorsClosedWhileMoving
    tr.Doors = 'closed'



Transition StartTrain
  SourceState stopped
  DestState moving
  Guard closed
  Actions null


[specs/train/TrainSystem-RE04.ob] - Cediti Objectiver 1.5.2 with advanced features
File Edit View Tools Document Windows Help
Doors for Train1
Train Controls
Moving
Opened
Animator Actor (localhost: Baldus@1095585565605)
Exp
Instance definition
KAOS object model
StateMachine
Vehicle
Create instance
Show state
Scope
Instance
Available scenarios
Crossing-FollowingTrains
Crossing-Train1CrossingOpened
Doors-OpenDoorsWhileMoving
Doors-StartWithDoorsOpened
MiniDemo-Train-at-Station
Various-Train1GoesAround
Various-TrainsGoBetweenStations
Execution trace
[Stop(vh:=Train#1)]GO
[CloseDoors(tr:=Train#1)]GO
[Start(vh:=Train#1)]GO
[OpenDoors(tr:=Train#1)]GO
=>
[]
>>
<
>
State
Input
Trace
SCL
Monitor
Debug
Req. is disabled
OpenDoors executed
FSM Visualizations
Train#1
Doors_...
'Closed'
CloseDoors
OpenDoors
'Opened'
State_T...
'Forward'
Start
Stop
'Stopped'
SECURITY GOAL FAILURE : door not closed while moving for Train#1
Hide Monitor Reporter
Modified
43M of 93M
Goal-oriented animation

One-slide summary:
model building should be goal-oriented

To enable ...

- satisfaction arguments *Specs, Assumptions* ⊢ *Goal*
- completeness & pertinence of the model
- early, incremental analysis (incl. risks, conflicts)
- model refinement & synthesis (deductive, inductive)
- reasoning about alternative options
- validation by stakeholders
- backward traceability
- generation of requirements document, architectural fragments, runtime requirements monitors, ...

Conclusion

- **Problem**-oriented abstractions, declarative specs needed
  - for ... communication with stakeholders
    - early, incremental analysis of partial models
    - model analysis and derivation
- **Systematic** techniques for model construction
  - from high-level goals to detailed operational specs
  - from detailed operational specs to high-level goals
- Appropriate mix of **deductive** & **inductive** techniques
- Importance of capturing assumptions + satisfaction args

Conclusion

- Anticipation of what could go wrong

  hazards, threats, conflicts, inadequate decisions, ...

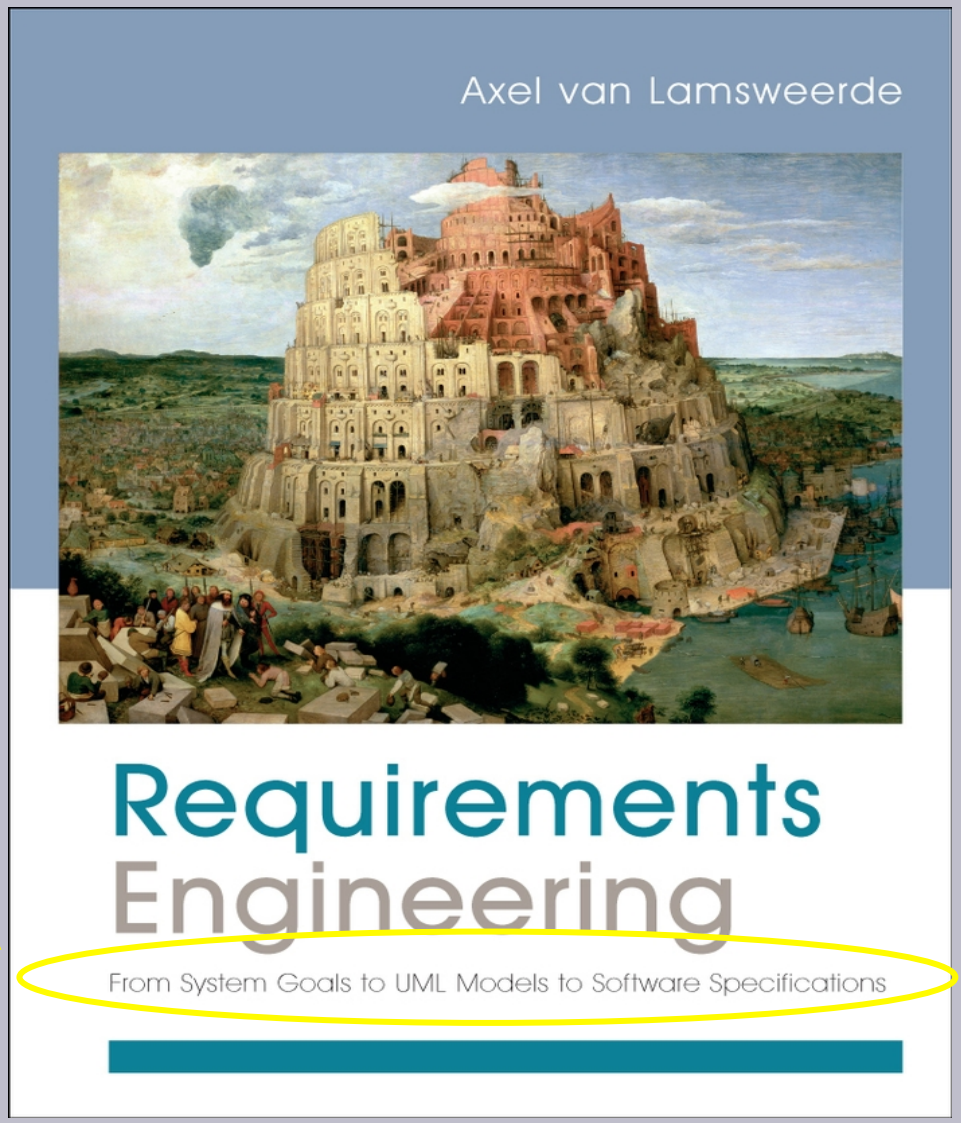
- Multi-button approach
  - Semi-formal

    for navigation, traceability ... and *accessibility*
  - Formal, when and where needed

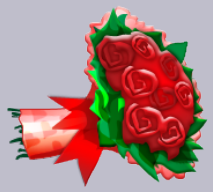
    for precise, automated reasoning on model pieces

Rigorous approaches needed !

Much, much more info in ...



Wiley, 2009



... and thanks to

Emmanuel Letier

Robert Darimont

Christophe Damas

Bernard Lambeau

Antoine Cailliau