

The KAOS approach to goal-oriented Requirements Engineering

Jann Sigrist, Michael Imhof, Raphael Schär

Content

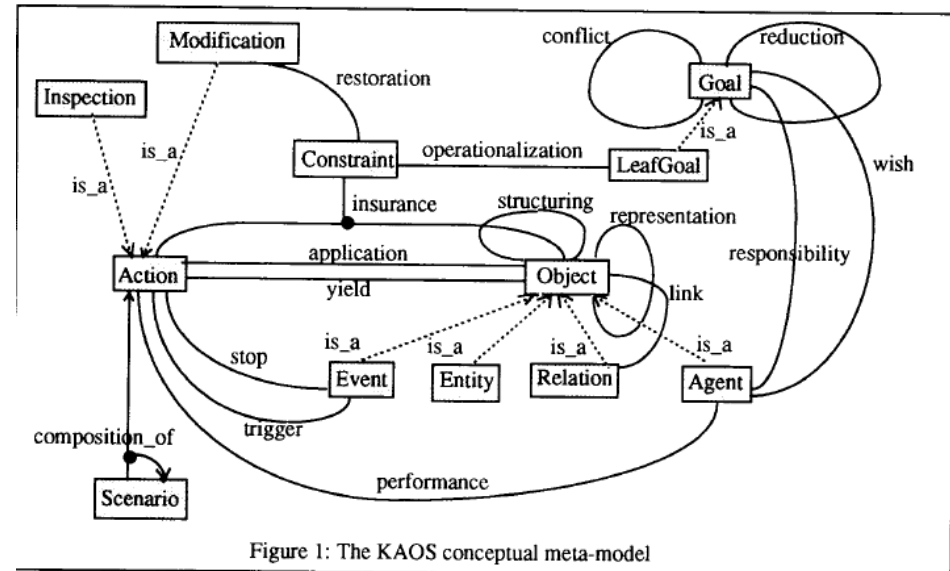
1. Overview
2. Goal modeling
3. Object modeling
4. Agent modeling
5. Operation modeling
6. Evaluation

Content

2. Goal modeling
3. Object modeling
4. Agent modeling
5. Operation modeling
6. Evaluation

Overview

- KAOS
- Knowledge acquisition in automated specification
- Three main phases:
 - Information gathering
 - **Modelling**
 - Document generation



Content

1. Overview

3. Object modeling

4. Agent modeling

5. Operation modeling

6. Evaluation

Goal modeling - I

- Goal identification from the initial document
- Rough overview
- +/- contributing or conflicting

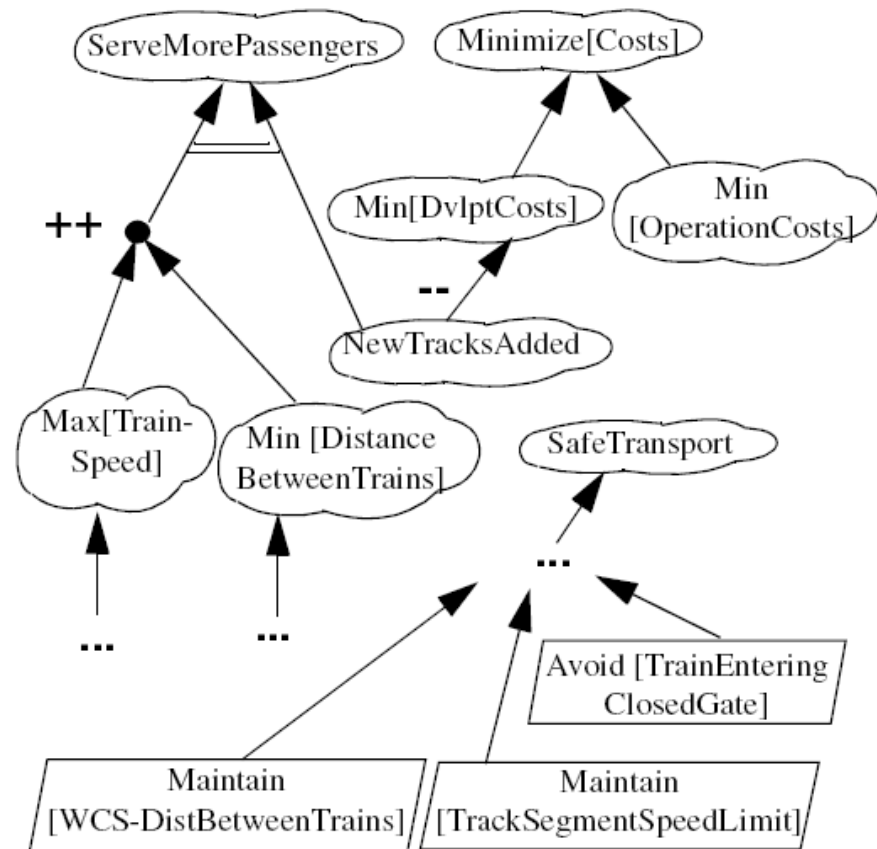


Figure 1 - Preliminary goal graph for the BART system

Clouds: Softgoals

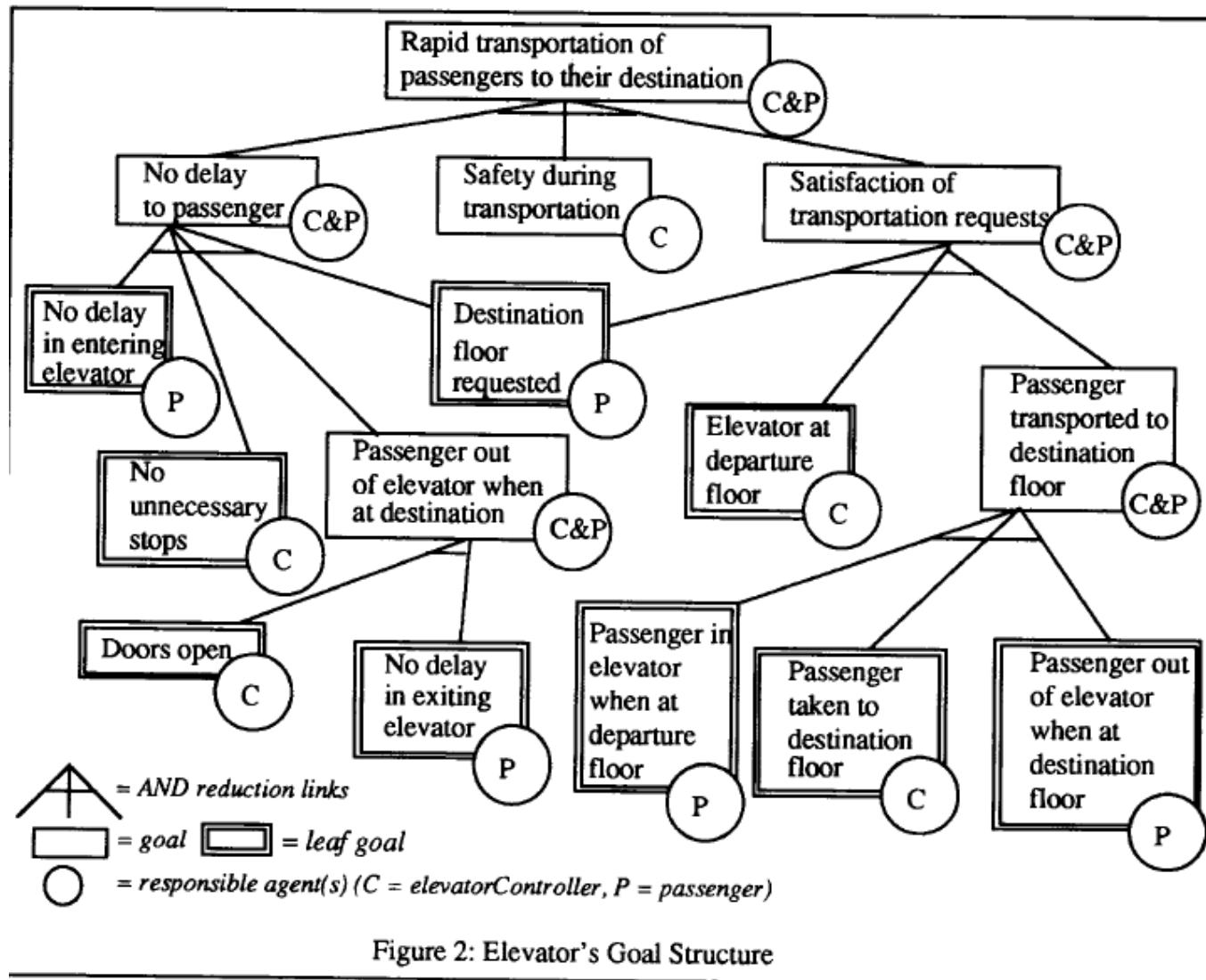
Parallelograms: formalizable goals

Arrows: subgoals

Goal modeling - II

- Refine goals until every goal belongs to one agent
- Check for conflicts between private and system goals!

Goal modeling - III



Content

1. Overview
2. Goal modeling

4. Agent modeling
5. Operation modeling
6. Evaluation

Object modeling

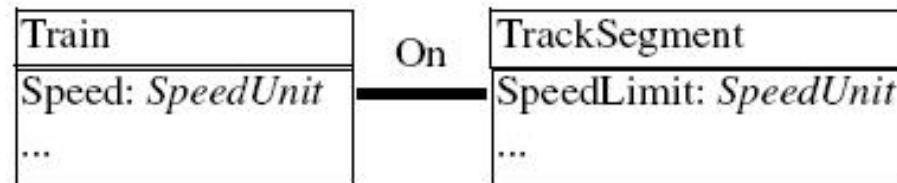
- principle: identify objects, relationships and attributes from goal specifications

From Goal model to object model

Goal Maintain[TrackSegmentSpeedLimit]

InformalDef *A train should stay below the maximum speed the track segment can handle.*

FormalDef $\forall tr: \text{Train}, s: \text{TrackSegment} :$
 $\text{On}(tr, s) \Rightarrow tr.\text{Speed} \leq s.\text{SpeedLimit}$



Content

1. Overview
2. Goal modeling
3. Object modeling

4. Use case modeling
5. Operation modeling
6. Evaluation

Agents

- Agent:
 - Person
 - Software
 - System
 - ...
- Identify agents
- assign responsibilities during construction of goal hierarchy
- Might not be complete
- Identify private goals

Content

1. Overview
2. Goal modeling
3. Object modeling
4. Agent modeling

6. Evaluation

Operationalize goals

- Introduce constraints for goals
- Formal assertions
- Assure tracability back to a goal
- Connects goals with actions (action closeDoors)

EXAMPLE: The “safety during transportation” goal can be operationalized with the following constraint:

$$\forall \text{ elevator: elevator.state='moving'} \Rightarrow \text{door.state='closed'}$$

Acquire action details

- Complete actions found through constraints:

- Links
- Attributes

EXAMPLE: The “safety during transportation” goal can be operationalized with the following constraint:

$\forall \text{ elevator: elevator.state='moving'}$
 $\Rightarrow \text{door.state='closed'}$.

- Two objects: Elevator and Door.
- Attributes: status.
 - moving bzw. closed
- Actions:
 - goToFloor
 - closeDoor
 - openDoor

Assign agents to actions

- Agents must perform actions in order to accomplish the goals they are responsible for in regard to the constraints.

EXAMPLE: The “safety during transportation” goal can be operationalized with the following constraint:

$$\forall \text{ elevator: elevator.state='moving'} \\ \Rightarrow \text{door.state='closed'}$$

- Action closeDoors was introduced.
- Two possible agents: elevatorController agent and passenger agent.
- elevatorController is responsible to reach the goal safety
à assign closeDoor action to elevatorController

Content

1. Overview
2. Goal modeling
3. Object modeling
4. Agent modeling
5. Operation modeling

Benefits

- Informal/formal for non-critical /critical components
- Systematic
- Requirements defined in KAOS
- Can be used until the late phases of requirements engineering

Disadvantages

- Complex for customers and beginners
- Experts needed (expensive)
- No textbooks, no education
- Only few tools