

KAOS

By,
Cynthia Aludogbu
Bhargav Bhatt

Topics :

Summary

Basic terms for KAOS

Examples

Discussion

Summary :

KAOS is a methodology for requirements engineering enabling analysts to build requirements models and to derive requirements documents from KAOS models.

The KAOS methodology is aimed at supporting the whole process of requirements elaboration.

The methodology comprises a specification language, an elaboration method, and meta-level knowledge used for local guidance during method enactment.

A constructive approach to goal refinement and operationalization.

The KAOS Language

The specification language provides constructs for capturing various kinds of concepts that appear during requirements elaboration, namely, goals, constraints, agents, entities, relationships, events, actions, views, and scenarios.

There is one construct for each type of concept.

The Underlying Ontology :

- Object
- Action
- Agent
- Goal
- Constrain

The KAOS Acquisition Process :

- Identify Goals, and their Concerned Objects.
- Identify potential Agents and their Capabilities.
- Operationalize Goals into Constraints.
- Refine Objects and Actions.
- Derive strengthened Objects and Actions to Ensure Constraints.
- Identify alternative Responsibilities.
- Assign Actions to responsible Agents.

A simple example :

Goal *Achieve* [ParticipantsConstraintsKnown]

InstanceOf InformationGoal

Concerns Meeting, Participant, Scheduler, ...

RefinedTo ConstraintRequested, ConstraintProvided

InformalDef *A meeting scheduler should know the constraints of the various participants invited to the meeting within C days after appointment*

FormalDef $\forall m: \text{Meeting}, p: \text{Participant}, s: \text{Scheduler}$

$\text{Invited}(p, m) \wedge \text{Scheduling}(s, m)$

$\Rightarrow \Diamond_{\leq C_d} \text{Knows}(s, p.\text{Constraints})$

Refining Goals into subgoals :

Refinement patterns Definition:- A refinement pattern is a one-level AND-tree of abstract goal assertions such that the set of leaf assertions is a complete refinement of the root assertion.

Useful for :

- Hidden formal reasoning
- Detecting incomplete refinement
- Explicit refinement

Example :

Goal Achieve [TrainProgress]

FormalDef $(\forall t: \text{Train}, b: \text{Block}) [\text{At}(t, b) \Rightarrow \Diamond \text{At}(t, b+1)]$

Goal Achieve [ProgressWhenGoSignal]

FormalDef $\forall t: \text{Train}, b: \text{Block}$

$\text{At}(t, b) \wedge \text{Go}[b+1] \Rightarrow \Diamond \text{At}(t, b+1)$

Goal Achieve [SignalSetToGo]

FormalDef $\forall t: \text{Train}, b: \text{Block}$

$\text{At}(t, b) \Rightarrow \Diamond \text{Go}[b+1]$

Goal Maintain [TrainWaiting]

FormalDef $\forall t: \text{Train}, b: \text{Block}$

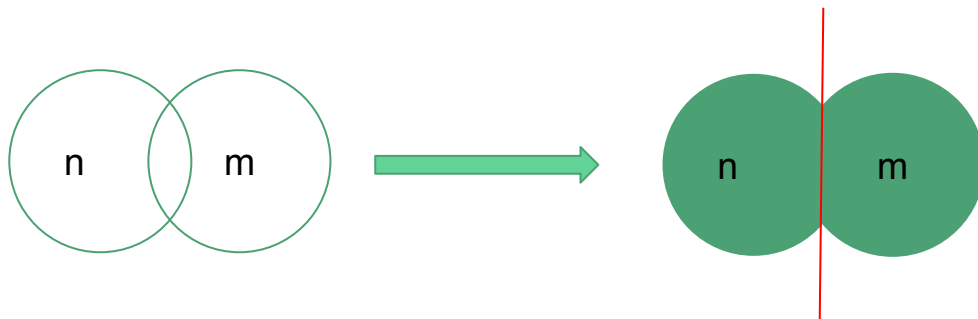
$\text{At}(t, b) \Rightarrow \text{At}(t, b) \mathcal{W} \text{At}(t, b+1)$

Conflicts

A conflict between assertions $A_1, \dots, A_n$ occurs within a domain Dom iff the following conditions hold:

1) $\{\text{Dom}, \bigwedge_{1 \leq i \leq n} A_i\} \models \text{false}$ (*logical inconsistency*)

2) for every i : $\{\text{Dom}, \bigwedge_{j \neq i} A_j\} \not\models \text{false}$ (*minimality*)



Discussion

- Are we making it more complex or more efficient?
- I* vs KAOS?