Introduction to KAOS Goal Modeling

Acknowledgements: Erik Fredericks

Agenda

• What is goal modeling?
• Goal modeling with KAOS
• Models at run time
What is goal modeling?

• What is a goal?
  – Representation of stakeholder objectives

• Who are the stakeholders?
  – Anybody interested in system
    • Developers, customers, maintainers, etc.

• What is a goal model?
  – Hierarchical arrangement of goals
  – Demonstrates relationships between goals

What are goals?

• Examples of goals:
  – Camera sensor must have 180 degree field of view
  – Radar sensor is always on
  – All sensors must provide reliable data

• Examples of non-goals:
  – Camera software implemented in C
  – Radar housing painted red
Goals can be decomposed from high-level objectives to low-level requirements.

Each goal refined with sub-goals that define how it can be satisfied.

Leaf-level goals are considered to be requirements.
Running Example

- Autonomous vehicle
  - Different types of sensors
    - E.g., camera and radar
  - Main objective: keep vehicle within lane

High-level vs. low-level goals

- Order these goals from high level to low level
  - Camera sensor provides data to processing unit
  - Vehicle must always stay within lane markings
  - Camera sensor must always be ON
  - Camera sensor detects objects within 10 meters
High-level vs. low-level goals

• Order these goals from high level to low level
  – Camera sensor provides data to processing unit
  – (1) Vehicle must always stay within lane markings
  – Camera sensor must always be ON
  – Camera sensor detects objects within 10 meters

High-level vs. low-level goals

• Order these goals from high level to low level
  – (2) Camera sensor provides data to processing unit
  – (1) Vehicle must always stay within lane markings
  – Camera sensor must always be ON
  – Camera sensor detects objects within 10 meters
High-level vs. low-level goals

- Order these goals from high level to low level
  - (2) Camera sensor provides data to processing unit
  - (1) Vehicle must always stay within lane markings
  - Camera sensor must always be ON
  - (3) Camera sensor detects objects within 10 meters

High-level vs. low-level goals

- Order these goals from high level to low level
  - (2) Camera sensor provides data to processing unit
  - (1) Vehicle must always stay within lane markings
  - (4) Camera sensor must always be ON
  - (3) Camera sensor detects objects within 10 meters
**High-level vs. low-level goals**

- Order these goals from high level to low level
  - Vehicle must always stay within lane markings
  - Camera sensor provides data to processing unit
  - Camera sensor detects objects within 10 meters
  - Camera sensor must always be ON

**Types of goals**

- Functional goals
  - “Hard” goals
  - Functions that system will perform
  - Well-defined criteria for satisfaction
  - E.g., vehicle always stays within lane markings

- Non-functional goals
  - “Soft” goals
  - Desired system qualities
  - Hard to define and quantify
  - Reliability
  - Quality
  - E.g., automatic stop is not jarring to passenger
Types of goals

- Safety goals
  - Ensure system consistently runs safely
  - Does not endanger people or system itself
  - E.g., sensor automatically shuts off if voltage exceeds maximum

- Failsafe goals
  - Provide safe fallback state in case of error
  - E.g., system shuts off if camera sensor is damaged

Goal exercise

- **Identify the goals** in the following paragraph:
  Company X is designing a new autonomous vehicle. Their autonomous vehicle system comprises at least two sensors: a camera and a radar. Both the camera and radar are responsible for sensing objects at a minimum distance of 10 meters. These sensors can communicate to a CPU via a secure CAN bus, at which point the CPU parses the incoming data. For safety purposes, at least one sensor must be active at all times.
Goal exercise

- Identify the **goals** in the following paragraph:

  Company X is designing a new autonomous vehicle. Their autonomous vehicle system comprises **at least two sensors**: a camera and a radar. Both the camera and radar are responsible for **sensing objects at a minimum distance of 10 meters**. These sensors can **communicate to a CPU** via a secure CAN bus, at which point the CPU **parses the incoming data**. For safety purposes, **at least one sensor must be active at all times**.

Why do we use goal models?

- Provide rationale for requirements
- Identify stable information in system objectives
- Guide requirements elaboration / elicitation
- Provide visual depiction of relationships and dependencies between objectives
When to use goal models

- Early in requirements engineering process
  - Identify problems
  - Explore solutions and alternatives
  - Performed prior to UML modeling
  - Continually refine goal model as new requirements or obstacles surface

When to use goal models

Goal modeling with KAOS

- Different approaches exist for creating goal models
  - KAOS, i*, UML (use cases)

- We will be using KAOS in this class for goal modeling
KAOS notation

- Refine goals into requirements
- Objects in KAOS goal model
  - Goal
  - Agent
  - Requirement / Expectation
  - Refinements

KAOS notation

```
(A) Vehicle remains within lane markings

(B) Send image data to processing unit
(C) Detect objects within 10m

(D) Camera feed is always valid
(E) Camera always ON
```

Camera
• Goals

(A) Vehicle remains within lane markings

(B) Send image data to processing unit

(C) Detect objects within 10m

(D) Camera feed is always valid

(E) Camera always ON

Camera

• Agent
  – Agent can be human or sensor

(A) Vehicle remains within lane markings

(B) Send image data to processing unit

(C) Detect objects within 10m

(D) Camera feed is always valid

(E) Camera always ON

Camera
• Requirement / Expectation
  • Requirement:
    – System component responsible
  • Expectation:
    – Environment responsible

(A) Vehicle remains within lane markings
(B) Send image data to processing unit
(C) Detect objects within 10m
(D) Camera feed is always valid
(E) Camera always ON

KAOS notation

AND-refinement

(A) Vehicle remains within lane markings
(B) Send image data to processing unit
(C) Detect objects within 10m
(D) Camera feed is always valid
(E) Camera always ON

Camera
KAOS notation

(A) Vehicle remains within lane markings

(B) Send image data to processing unit

(C) Detect objects within 10m

OR-refinement

(D) Camera feed is always valid

(E) Camera always ON

Goal categories

• Invariant
  – Must **ALWAYS** be satisfied
  – Safety, failsafe, or system invariants
  – E.g., vehicle can never collide with an obstacle

• Non-invariant
  – Temporarily **tolerate** unsatisfied goal
  – Transient conditions
  – E.g., sensor temporarily occluded
Goal examples

(A) Vehicle remains within lane markings

(B) Send image data to processing unit

(C) Detect objects within 10m

(D) Camera feed is always valid

(E) Camera always ON

Goal examples

(A) Vehicle remains within lane markings

Invariant goal

(A) Maintain [Vehicle remains within lane markings]

(Avoid also considered invariant)
E.g., Avoid [Collision]
Goal examples

(A) Maintain [Vehicle remains within lane markings]

(B) Send image data to processing unit

(C) Detect objects within 10m

(D) Camera feed is always valid

(E) Camera always ON

Goal examples

(D) Detect objects within 10m

Non-invariant goal

(D) Achieve [Detect objects within 10m]
Goal examples

- All have been functional so far… (A)

- What about non-functional?
  - Soft / fuzzy goals?

Goal examples

Maintain [Vehicle remains within lane markings] (A)

Achieve [Send image data to processing unit] (B)

Achieve [Detect objects within 10m] (C)

Achieve [Camera feed is always valid] (D)

Achieve [Camera always ON] (E)

Camera
Goal examples

Maintain
[Vehicle remains within lane markings]

Achieve
[Vehicle drives smoothly]

Achieve
[Send image data to processing unit]

Achieve
[Detect objects within 10m]

Achieve
[Camera feed is always valid]

Achieve
[Camera always ON]

No difference between functional and non-functional goal in KAOS!
Using models at run time

- Goal models can provide measure of system performance
- Is system satisfying its invariants
- How well is system satisfying its non-invariants?

Utility functions

- Derived mathematical function for each goal
- Provides degree of satisfaction for each goal (satisficement)
- Normalized on [0.0, 1.0]
  - 0.0 ➔ goal is violated
  - 1.0 ➔ goal is satisfied
  - (0.0, 1.0) ➔ goal is satisfied to some degree
Goal examples

Maintain [Vehicle remains within lane markings]

(A) Send image data to processing unit

(B) Detect objects within 10m

Invariants are typically Boolean functions

Goal (A) = 1.0 if vehicle has never crossed lane boundary
Goal (A) = 0.0 otherwise
Also, Goal (A) = 0.0 if (Goal (B) = 0.0) OR (Goal (C) = 0.0)

Goal examples

Maintain [Vehicle remains within lane markings]

(A) Send image data to processing unit

(B) Detect objects within 10m

Non-invariants are typically real functions

Goal (C) = 1.0 – |distance_{detected} – distance_{max}|

Goal (C) = 1.0 - |8.0-10.0| = 10.0
Goal (C) = 0.8
What can we do with a utility value?

- Determine if an objective has been violated
  - Or, determine if an objective is not being satisfied enough

- Refactor goal model
- Introduce a bug fix or patch
- Reconfigure system

In-class Assignment Feb. 14, 16

- Derive goal model based on requirements specification for your backup rollover assignment
- Must have at minimum:
  - (2) Invariant goals
  - (6) Non-invariant goals
  - (3) System agents
  - (2) AND- and (2) OR- refinements