SWE 760
Real-Time Software Design

Lecture 6 – Dynamic Interaction Modeling for Real-Time Embedded Systems


Hassan Gomaa
Dept of Computer Science
George Mason University
Fairfax, VA

Figure 4.1 COMET/RTE life cycle model
Analysis Modeling

• Static Modeling
  – Define entity classes and relationships
• Dynamic State Machine Modeling
  – Real-time systems are highly state dependent
  – Actions depend on input event AND current state
• Object Structuring
  – Determine objects that realize each use case
• Dynamic Interaction Modeling
  – Determine sequence of interactions among objects

Dynamic Interaction Modeling

• Use cases realized in Dynamic Model
  – Show objects participating in each use case
• Determine how objects participate in use case
  – Use object structuring criteria to determine objects
    • Stereotype for each object structuring criterion
  – Shows sequence of object interactions in use case
    • Depict on
      – communication diagram or
      – sequence diagram
• State dependent control objects
  – Modeled using statecharts
• Dynamic Modeling
  – Approach to determine how objects interact with each other to support use case
Dynamic Modeling

- Determine how objects interact with each other to support use case
  - Start with external event from actor
  - Determine objects needed to support use case
  - Determine sequence of internal events following external event
  - Depict on communication diagram or sequence diagram
- Stateless (non state-dependent) Dynamic Modeling
- State dependent Dynamic Modeling

Stateless Dynamic Modeling

- Start with use case
- Determine boundary objects
  - Receives external events from actor
- Determine internal objects
  - Receive messages from boundary objects
- Determine object interactions
  - Sequence of messages passed
- Develop main interaction sequence (scenario)
- Develop alternative sequences
  - For alternative branches of use case
  - E.g, for error handling or less frequently occurring conditions
Example of Stateless Dynamic Modeling

Figure 9.4 Send Vehicle Status use case

![Diagram showing the relationship between actors (timer, human), entities (VehicleData, Output), and actions (Timer Event, read(out location, out speed), Vehicle Status)]

Fig 9.5: Sequence diagram for Send Vehicle Status use case
State Dependent Dynamic Modeling

- Object interaction controlled by statechart(s)
  - Control object
    - Executes statechart
    - Activates/deactivates other objects
  - For each use case
    - Determine objects participating in use case
    - Determine sequence of object communication
    - Develop statechart for control object
  - For each event that arrives at control object
    - Determine state transition from current state to next state
    - Determine actions and activities to be executed
    - Determine objects required to perform actions and activities
Figure 21.2 Light Rail Control System context block diagram

1. External User: RailOperator
   - Interacts with:
     - LightRailEmbeddedSystem
     - LightRailControlSystem
     - StationDisplay
     - StationAudioDevice
     - WaysideMonitoringSystem
     - RailroadCrossingControl
     - WakeupObserver

2. Embedded System: LightRailControl
   - Outputs to:
     - MotorActuator
     - DoorActuator
     - TrainDisplay
     - TrainAudioDevice
     - StationDisplay
     - WakeupObserver

3. External System: RailroadCrossingControl
   - Communicates with:
     - WakeupObserver

4. External System: WaysideMonitoringSystem
   - Communicates with:
     - WakeupObserver

5. External System: RailOperationsDisplay
   - Communicates with:
     - WakeupObserver

6. External System: RailOperationsObserver
   - Communicates with:
     - WakeupObserver

7. External User: RailOperator
   - Interacts with:
     - ArrivalSensor
     - ApproachingSensor
     - DepartureSensor
     - ProximitySensor
     - LocationSensor
     - SpeedSensor

8. External Input Device: DoorSensor
   - Inputs to:
     - WakeupObserver

Figure 21.3 Light Rail Control Software system context block diagram

1. External User: RailOperator
   - Interacts with:
     - Software System: LightRailControl
     - WaysideMonitoringSystem
     - RailroadCrossingControl
     - WakeupObserver

2. External System: LightRailControl
   - Communicates with:
     - WakeupObserver

3. External System: WaysideMonitoringSystem
   - Communicates with:
     - WakeupObserver

4. External System: RailroadCrossingControl
   - Communicates with:
     - WakeupObserver

5. WakeupObserver
   - Communicates with:
     - WakeupObserver

Copyright 2016 H. Gomaa
Arrival Sensor

Control Train Operation

Departure Sensor

Approaching Sensor

Arrival Sensor

Figure 21.4 Light Rail Control System actors and use cases: Light Rail Operations use case package

Figure 21.5 Light Rail Control System: Train Dispatch and Suspend use case package
Figure 21.6 Light Rail Control System: Railroad Hazard Detection

Figure 21.9 State machine for Arrive at Station use case
Figure 21.10 State machine for Control Train at Station use case

Figure 21.11 State machine for Depart from Station use case
Figure 21.12 State machine for Suspend Train use case

Figure 21.13 State machine for Dispatch Train use case
Figure 21.16 Flat state machine for Control Train

Figure 21.17 Hierarchical state machine for Control Train
Figure 21.18 Light Rail Control Software subsystems

Figure 24.19 Input and output classes for Train Control Subsystem
Dynamic Interaction Modeling of External Objects with Software System

- Starting with
  - Use case model
  - System context model
- From software system context diagram
  - Model software system as one aggregate object
  - Model instances of external objects from
- From use case model
  - Follow sequence of interactions described in use case
  - Depict interaction with external objects
Figure 21.26 Sequence diagram for Detect Obstacle Presence use case

Copyright 2016 H. Gomaa

Figure 21.27 Sequence diagram for Detect Obstacle Removal use case

Copyright 2016 H. Gomaa
Figure 4.1 COMET/RTE life cycle model

User

- System Structural Modeling
  - Requirements Modeling
    - Analysis Modeling
      - Design Modeling
        - Incremental Software Construction
        - Incremental Software Integration
        - Incremental Prototyping
      - System Testing

Customer
Analysis Modeling

- **Static Modeling**
  - Define entity classes and relationships
- **Dynamic State Machine Modeling**
  - Real-time systems are highly state dependent
  - Actions depend on input event AND current state
- **Object Structuring**
  - Determine objects that realize each use case
- **Dynamic Interaction Modeling**
  - Determine sequence of interactions among objects

**Figure 4.1 COMET/RTE life cycle model**