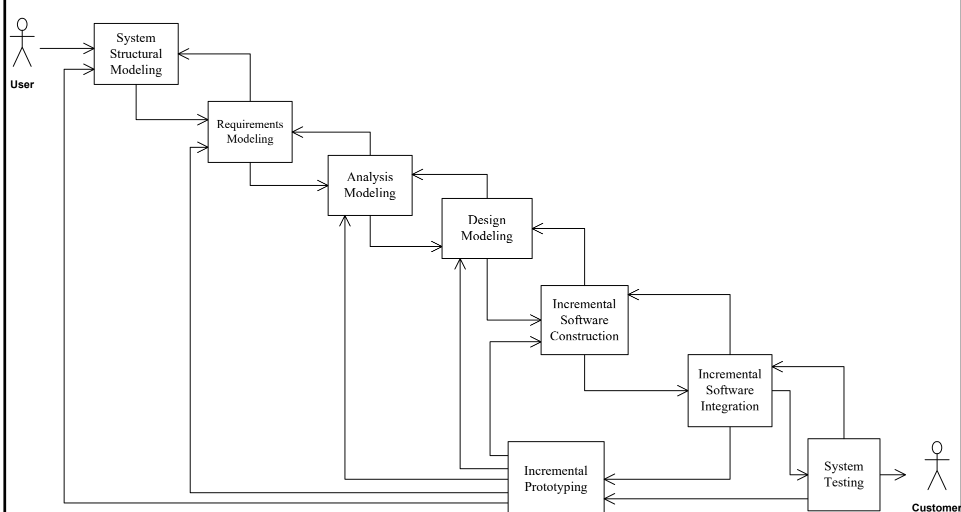
SWE 760

Lecture 3: Use Case Modeling for Real-Time Embedded Systems

Hassan Gomaa
Department of Computer Science
George Mason University
Email: hgomaa@gmu.edu

References:
H. Gomaa, Chapter 6 - *Real-Time Software Design for Embedded Systems*, Cambridge University Press, 2016

Figure 4.1 COMET/RTE life cycle model



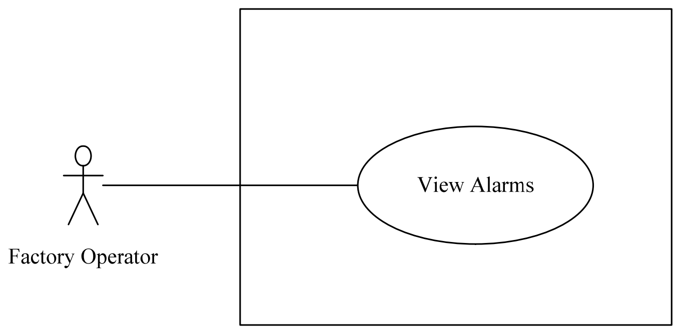
Requirements Modeling in COMET/RTE

- Develop use case model
  - Define system functional requirements in terms of actors and use cases
  - Two different perspectives for RTE use case model
    - Systems Engineering perspective
    - Software Engineering perspective
- Develop non-functional requirements
  - A.k.a. quality requirements

Use Case Modeling

- Define system functional requirements in terms of Actors and Use Cases
  - Each use case defined in terms of sequence of interactions between Actor and System
    - Narrative description
- Use case is a complete sequence of interactions initiated by an actor
  - Use case starts with input from an actor
  - Describes interactions between actor and system
  - Provides value to actor
  - Basic path
    - Most common sequence
  - Alternative branches
    - Variations of basic path
    - E.g., for error handling

Example of actor and use case
(Fig. 6.1)



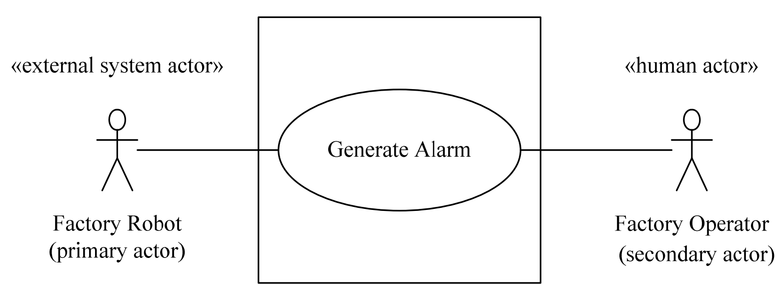
Actors

- Actor models external entities of system
- Actor initiates use case
- Actor interacts with the system
  - By providing inputs to the system
  - By responding to outputs from the system
- Actor must affect sequence of events in use case
- For real-time embedded systems
  - All external entities are actors

Actors and Use Cases

- Primary Actor
  - Starts the use case by providing input to the system
- Secondary Actor(s)
  - Participates in use case
  - Can be Primary Actor of a different use case
- Identifying use cases from actors
  - Consider each major function an actor needs to perform
  - Provides value to actor

Example of primary and secondary actors
(Fig. 6.2)



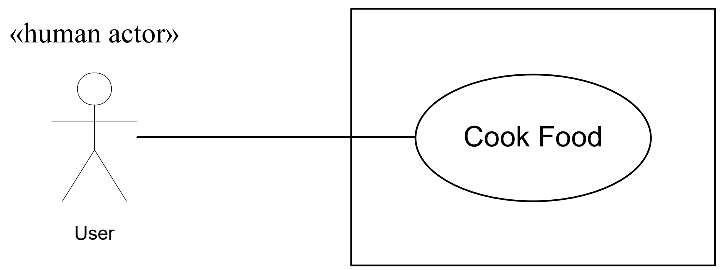
Use Cases for Systems Engineering and Software Engineering

- Systems Engineering and Software Engineering perspectives on use cases
- Little or no difference for information or web-based systems
  - Actors are mainly human users, possibly external systems
  - No difference between system and software context diagrams
- RT embedded systems
  - Big difference in Systems and Software Engineering perspectives on use cases
  - Many actors are non-human

Modeling Actors from a Systems Engineering Perspective

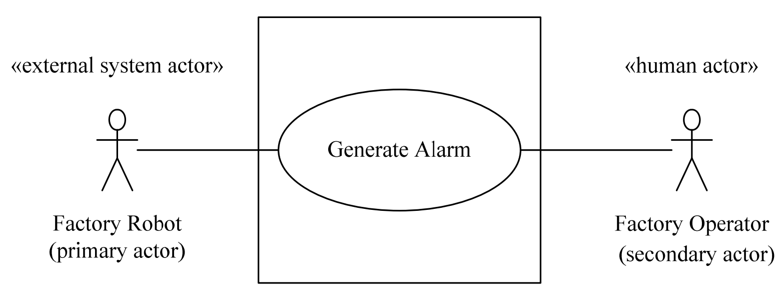
- Actors in Systems Engineering use cases
  - Human user
    - e.g., Microwave User
  - External system
    - E.g., Factory Robot
  - Physical entity
    - E.g., Train, Vehicle

Systems Engineering Perspective
- Human User as Actor (Fig. 6.3)



11

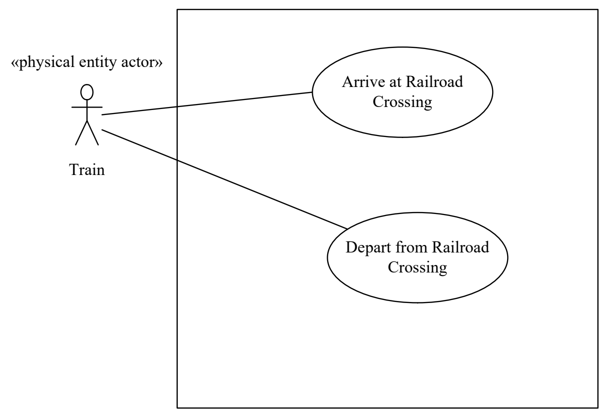
Systems Engineering Perspective
- External system as actor (Fig. 6.2)



12

Systems Engineering Perspective
- Physical entity as Actor

Figure 6.4 Example of physical entity actor



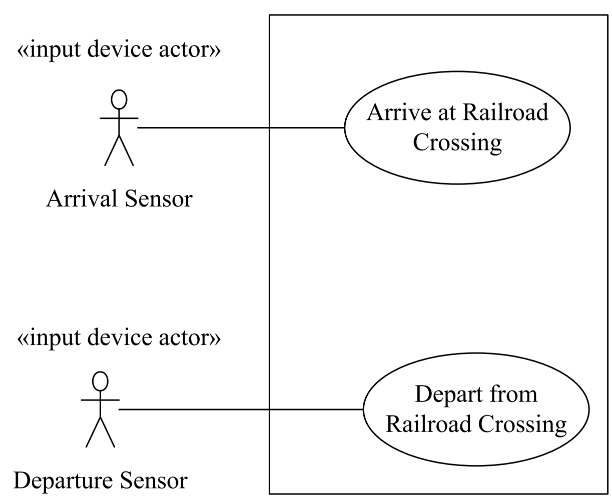
Actors in RT Embedded Systems

- Many non-human actors
- E.g., Vehicle arrival and departure detected by System
- Systems engineering perspective
  - Vehicle is an actor
  - External to the system
- Software engineering perspective
  - Vehicle does not interact directly with the system
  - Vehicle arrival and departure detected by sensors
  - Arrival and departure sensors
    - Internal to total hardware/software system
    - External to software system
  - Arrival and departure sensors are actors

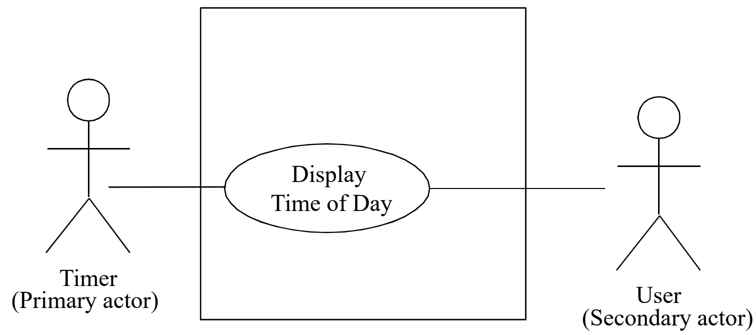
Modeling Actors from a Software Engineering Perspective

- Actors in Software Engineering use cases
  - Same actors as in Systems Engineering use cases
    - Human user
    - External system
- Actors ONLY in Software Engineering use cases
  - Actors that represent objects that are
  - Part of total hardware/software system
  - External to software system
    - Input device
    - Output device
    - I/O device
    - Timer

Software Engineering Perspective
Example of input device actors (Fig. 6.5)



Software Engineering Perspective
Example of timer actor (Fig. 6.6)



Non-functional Requirements

- Quality of service goal that system must fulfill
  - Also called Quality Attribute
- E.g. of non-functional requirements:
- **Performance requirement**: System shall respond to timer inputs within 100 milliseconds.
- **Safety requirement**: System shall switch off the microwave oven if temperature exceeds a pre-specified hazard level.
- **Availability requirement**: System shall be operational 99.9% of required time.
- **Security requirement**: System shall encrypt operator id and password.

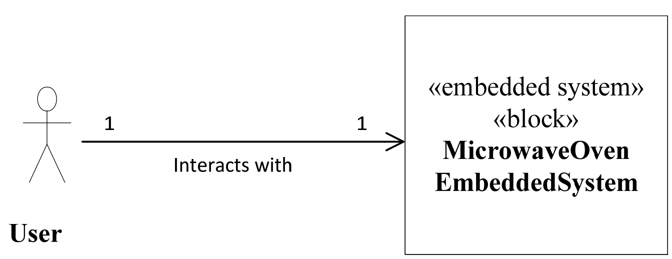
Documenting Use Cases

- Name
- Summary - Short description of use case
- Dependency (on other use cases)
- Actors – primary, secondary
- Preconditions
  - Conditions that are true at start of use case
- Description of main sequence
  - Narrative and textual description
- Description of alternative sequences
  - Textual description of alternative branches
- Nonfunctional requirements
  - E.g., performance and security requirements
- Postcondition
  - Condition that is true at end of use case

19

System Context Diagram for Microwave Oven System

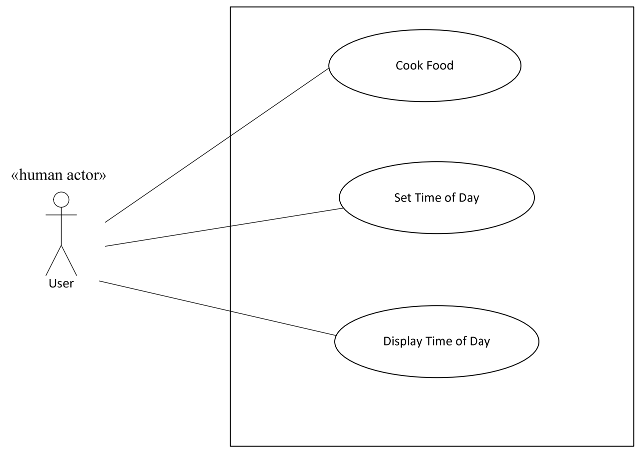
Figure 19.2: Microwave Oven System context diagram



SysML Block Definition Diagram

20

Systems Engineering Perspective
Use Case Model for Microwave Oven System
(Fig. 6.8)



Systems Engineering Perspective
Use Case Model for Microwave Oven System

Use case name: Cook Food.

Summary: User puts food in oven, and microwave oven cooks food.

Actors: User

Precondition: Microwave oven is idle.

Main Sequence:

1. **User opens the door**
2. **System switches the oven light on.**
3. **User puts food in the oven and closes the door.**
4. **System switches the oven light off.**
5. **User presses the Cooking Time button.**
6. **System prompts for cooking time.**
7. **User enters cooking time on the numeric keypad and presses Start.**
8. **System starts cooking the food and switches the oven light on.**
9. **System continually displays the cooking time remaining.**

Systems Engineering Perspective
Use Case Model for Microwave Oven System

10. System timer detects that the cooking time has elapsed.

11. System stops cooking the food, switches off the light, stops the turntable, sounds the beeper, and displays the end message.

12. User opens the door.

13. System switches on the oven light.

14. User removes the food from the oven and closes the door.

15. System switches off the oven light and clears the display.

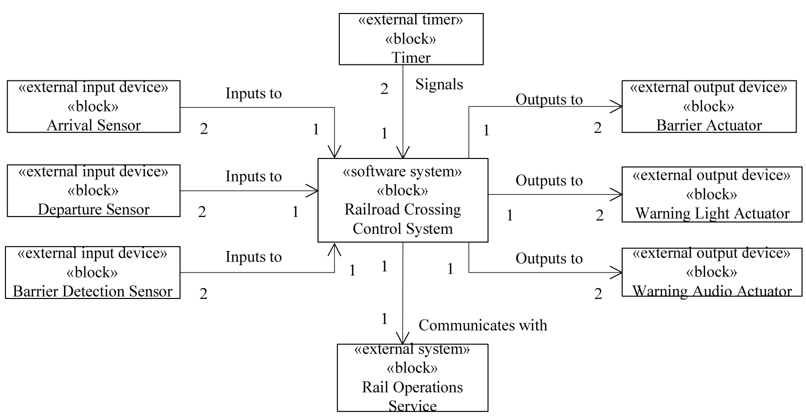
Configuration requirement:

Name: Display Language.

Description: There is a choice of language for displaying messages, which is set during system configuration. Default language is English.

Postcondition: Microwave oven has cooked the food.

Fig. 5.10: Software System Context Diagram for Railroad Crossing Control System

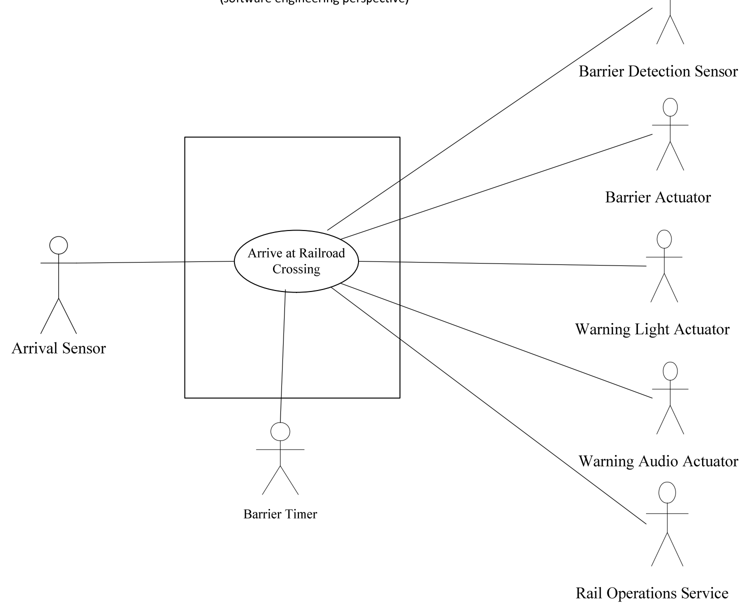


Actors in Railroad Crossing Embedded System

- Systems engineering perspective
  - Train is an actor
  - External to the system
- Software engineering perspective
  - Train does not interact directly with the system
  - Train arrival and departure detected by sensors
  - Arrival and departure sensors
    - Internal to total hardware/software system
    - External to software system
  - Arrival Sensor and Departure Sensors are actors

Example of Use Case Model - Software Engineering Perspective

Figure 6.9 Use Case model for Railroad Crossing Control System (software engineering perspective)



Use Cases - Railroad Crossing Control System

Use case name: Arrive at Railroad Crossing

Summary: Train approaches railroad crossing. The system lowers the barrier, switches on the warning lights, and switches on the audio warning alarm.

Actors:

- Primary actor: Arrival Sensor
- Secondary actors: Barrier Detection Sensor, Barrier Actuator, Warning Light Actuator, Warning Audio Actuator, Rail Operations Service, Barrier Timer.

Precondition: The system is operational and there is either no train or one train in the railroad crossing.

Main Sequence:

1. Arrival Sensor detects the train arrival and informs the system.
2. System commands each Barrier Actuator to lower a barrier, each Warning Light Actuator to switch on the flashing lights, and each Warning Audio Actuator to switch on the audio warning.
3. Barrier Detection Sensor detects that a barrier has been lowered and informs the system.
4. System sends a train arrival message to Rail Operations Service.

Alternative Sequences:

Step 2: If there is another train already at the railroad crossing, skip steps 2 and 3.

Step 3: If Barrier Timer notifies the system that the lowering timer has timed out, the system sends a safety warning message to the Rail Operations Service.

Use Cases - Railroad Crossing Control System

Non-functional Requirements:

a) **Safety requirements**:

- Barrier lowering time shall not exceed a pre-specified time. If timer times out, the system shall notify Rail Operations Service.
- System shall keep track of the number of trains at the railroad crossing, such that the barrier is lowered when the first train arrives and only raised after the last train departs.

b) **Performance requirement:**

- The elapsed time from the detection of train arrival to sending the command to the barrier actuator shall not exceed a pre-specified response time.

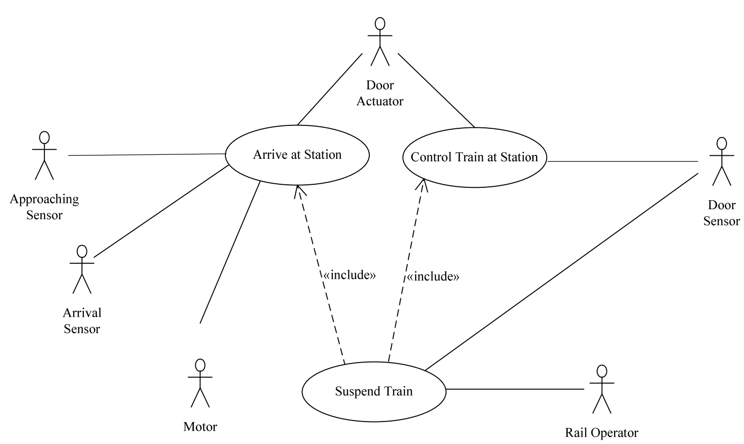
Postcondition: The barrier has been closed, the warning lights are flashing and the audio warning is sounding.

Use Case Relationships

- **Include** relationship
  - Identify common sequences of interactions in several use cases
    - Extract common sequence into **inclusion use case**
    - Base use cases **includes** inclusion use case
- Example
  - Suspend Train use case
    includes
  - Arrive at Station use case

29

Example of inclusion use cases and include relationship (Fig. 6.10)



30

Example of inclusion use cases and include relationship

Use case: Control Train at Station.

Actors: Door Sensor (primary), Door Actuator.

Precondition: Train is stopped at station with doors opening.

Main sequence:

1) Door Sensor sends Doors Opened message.
2) After time interval, System sends Close Doors command to the Door Actuator.

Postcondition: Train is stopped at station with doors closing.

Use case: Arrive at Station.

Actors: Approaching Sensor (primary), Arrival Sensor, Motor, Door Actuator.

Precondition: Train is moving toward next station.

Main sequence:

1) Approaching Sensor signals that train is approaching station
2) System sends Decelerate command to Motor.
3) Arrival Sensor signals that train is entering station
4) System sends Stop Motor command to Motor.
5) Motor responds that train has stopped.
6) System sends Open Doors command to the Door Actuator.

Postcondition: Train has stopped at station with doors opening.

Example of base use case and include relationship

Use case: Suspend Train.

Actor: Rail Operator (primary), Door Sensor.

Dependency: Includes Arrive at Station, Control Train at Station use cases.

Precondition: Train is operational and moving toward next station.

Main sequence:

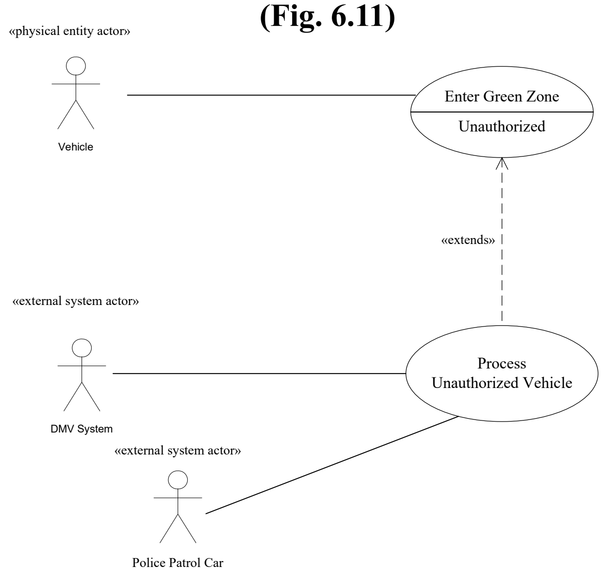
1) Rail Operator sends suspend train operation command to System.
2) Include Arrive at Station use case.
3) Include Control Train at Station use case.
4) Door Sensor sends Doors Closed message to System.

Postcondition: Train is stationary at station and out of service.

Use Case Relationships

- **Extend** relationship
  - Use case A is an extension of use case B
  - Under certain conditions use case B will be extended by description given in use case A
  - Same use case can be extended in different ways
- When to use **extend**
  - Show conditional parts of use case
  - Model complex or alternative paths
- Example of extension use cases and **Extend** relationship
  - Store Part, Retrieve Part
  - Extend Flexibly Manufacture Part

Example of extension use case and Extend relationship (Fig. 6.11)



Example of base use case and Extend relationship

Use case: Enter Green Zone.

Summary: Vehicle enters restricted Green Zone; System starts tracking the vehicle.

Actor: Vehicle.

Precondition: Green Zone entry point is clear.

Main Sequence:

1. Vehicle approaches green zone entry point.
2. System detects vehicle entering the green zone.
3. System reads vehicle permit number RFID.
4. System checks that permit number is valid.
5. System stores the following information: permit number, entry time/date, entry location.

Alternative Sequence:

Step 4: Unauthorized (i.e., unrecognized or missing permit number): Extend with *Process Unauthorized Vehicle* use case.

Postcondition:

Vehicle has entered green zone.

Example of extension use case and Extend relationship

Use case: Process Unauthorized Vehicle.

Summary: The license number of the unauthorized vehicle is detected, decoded, and sent to the police.

Actor: Vehicle (primary), Police Patrol Car (secondary), DMV System (secondary).

Dependency: Extends *Enter Green Zone* use case.

Precondition: Vehicle has an invalid or nonexistent permit number.

Description of insertion segment:

1. System takes photograph of vehicle license plate.
2. System uses an image processing algorithm to analyze the photograph and extract the state name and vehicle license number.
3. System sends a message to the DMV system of the vehicle's state containing the vehicle license number and requesting owner name and address.
4. The DMV system sends a message to the system containing the name and address of the vehicle owner.
5. The system issues and prints a fine to be sent by mail to the vehicle owner.

Postcondition: The unauthorized vehicle has been detected and a fine has been issued.

Alternative sequence:

Step 2: License plate cannot be decoded (because of bad photograph, bad weather, covered license plate); System sends alert message to the Police Patrol Car.

Figure 4.1 COMET/RTE life cycle model