Design - use case realization

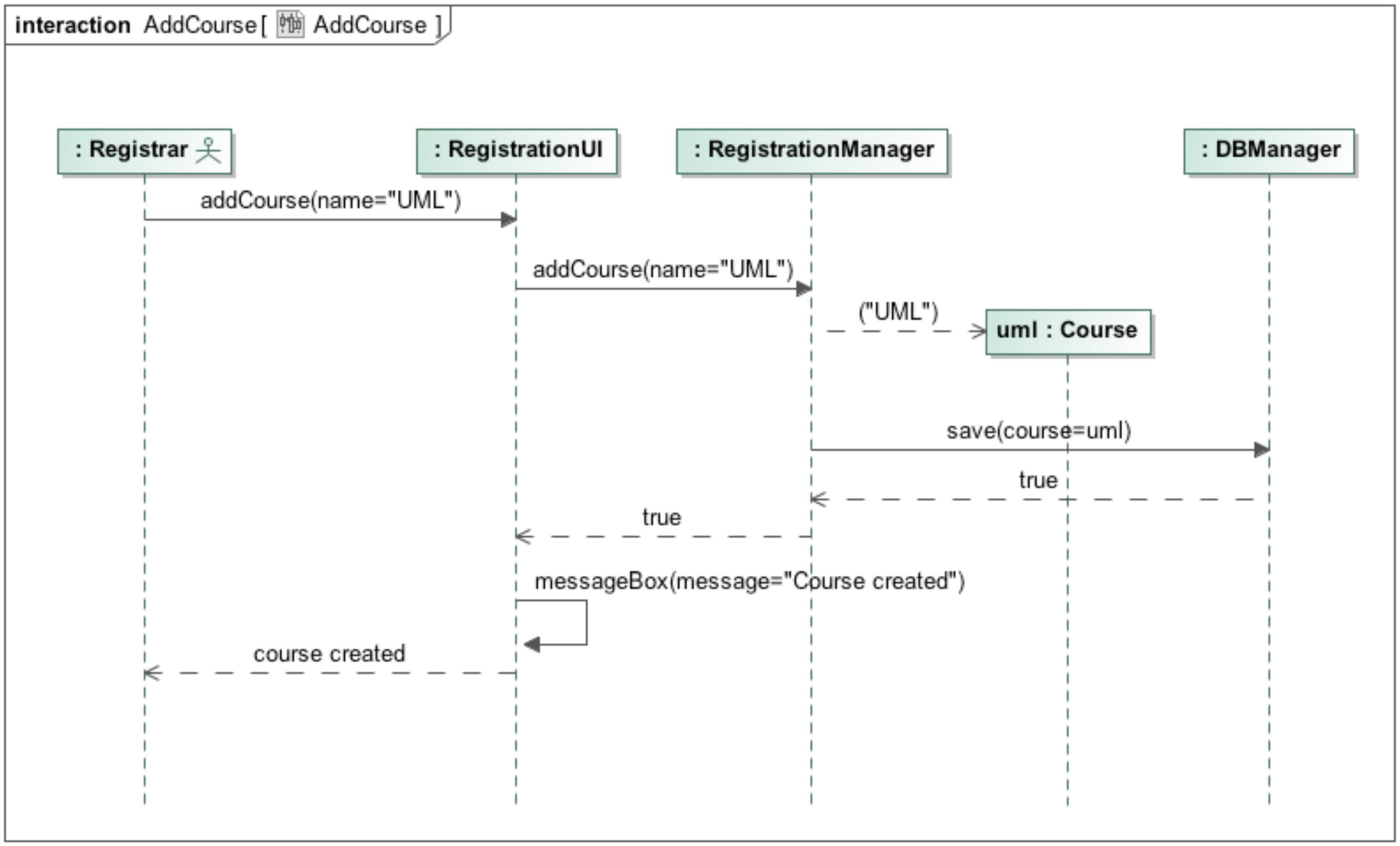
Use case realization - design

- A collaboration of design objects and classes that realize a use case comprising:
  - Interaction diagrams
  - Links to class diagrams containing the participating design classes
  - An explanatory text (flow)
- There is a trace between an analysis use case realization and a design use case realization. The content is similar, but the design version contains implementation details - it specifies implementation decisions and implements the non-functional requirements

Interaction diagrams in design

- Only produce design interaction diagrams where they add value to the project:
  - A refinement of the analysis interaction diagrams to illustrate design issues
  - New diagrams to illustrate technical issues
  - New diagrams to illustrate central mechanisms
- In design, sequence diagrams are used much more than communication diagrams
- Timing diagrams may be used to capture timing constraints

Design sequence diagram

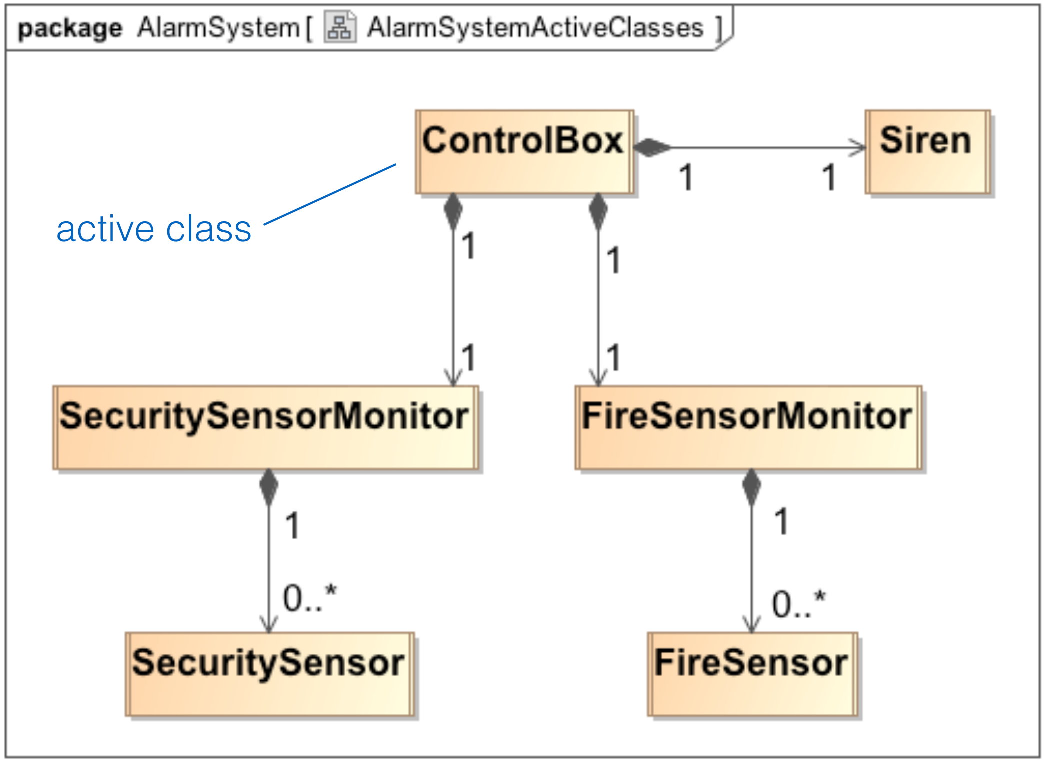


- Show implementation details such as UIs and databases

Concurrency - active classes

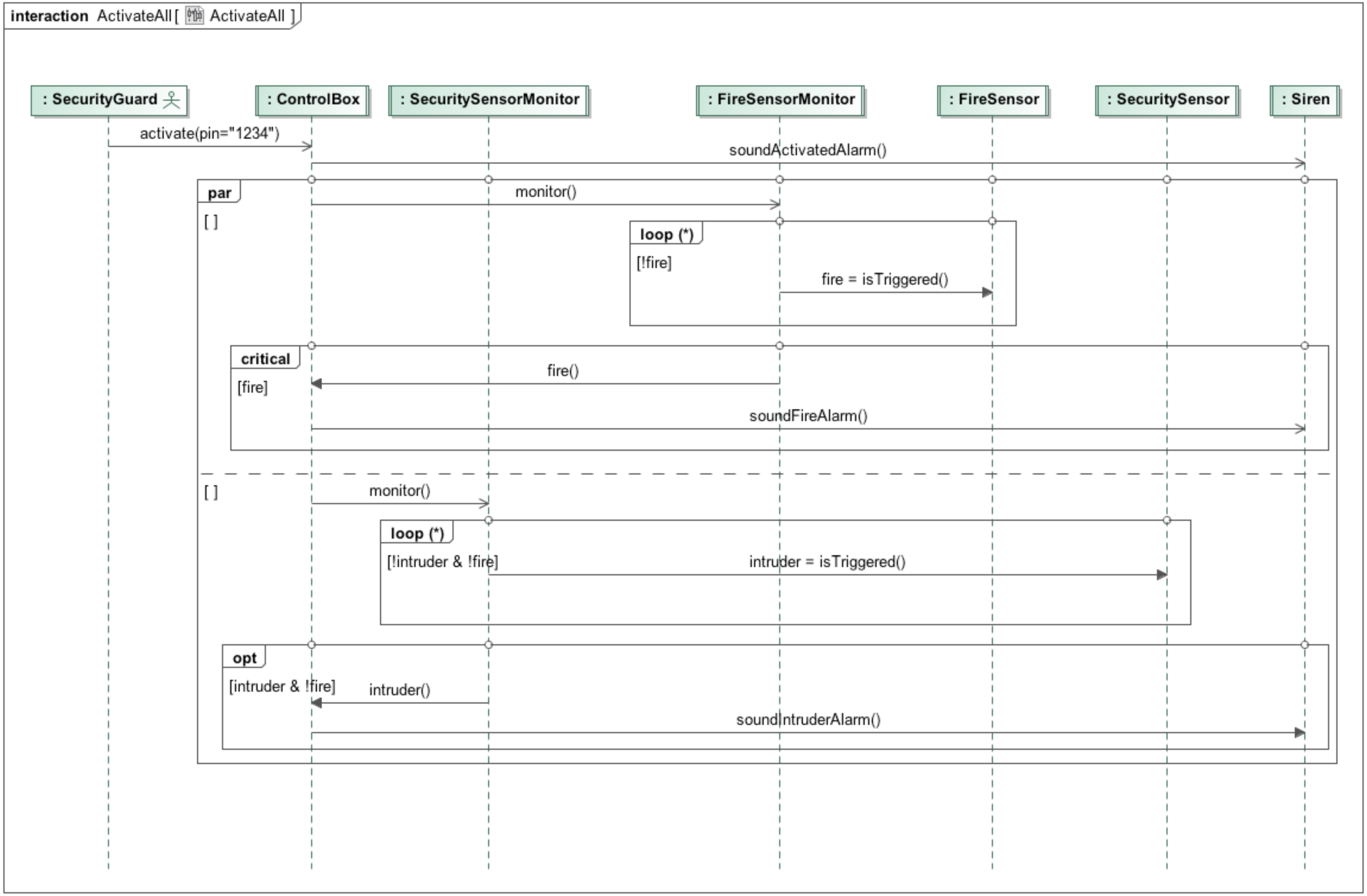
- Active classes are classes whose instances are active objects that have their own threads of control

- Concurrency is best modeled with sequence and timing diagrams. You can also model it with communication diagrams by labelling threads of execution, but it is messy

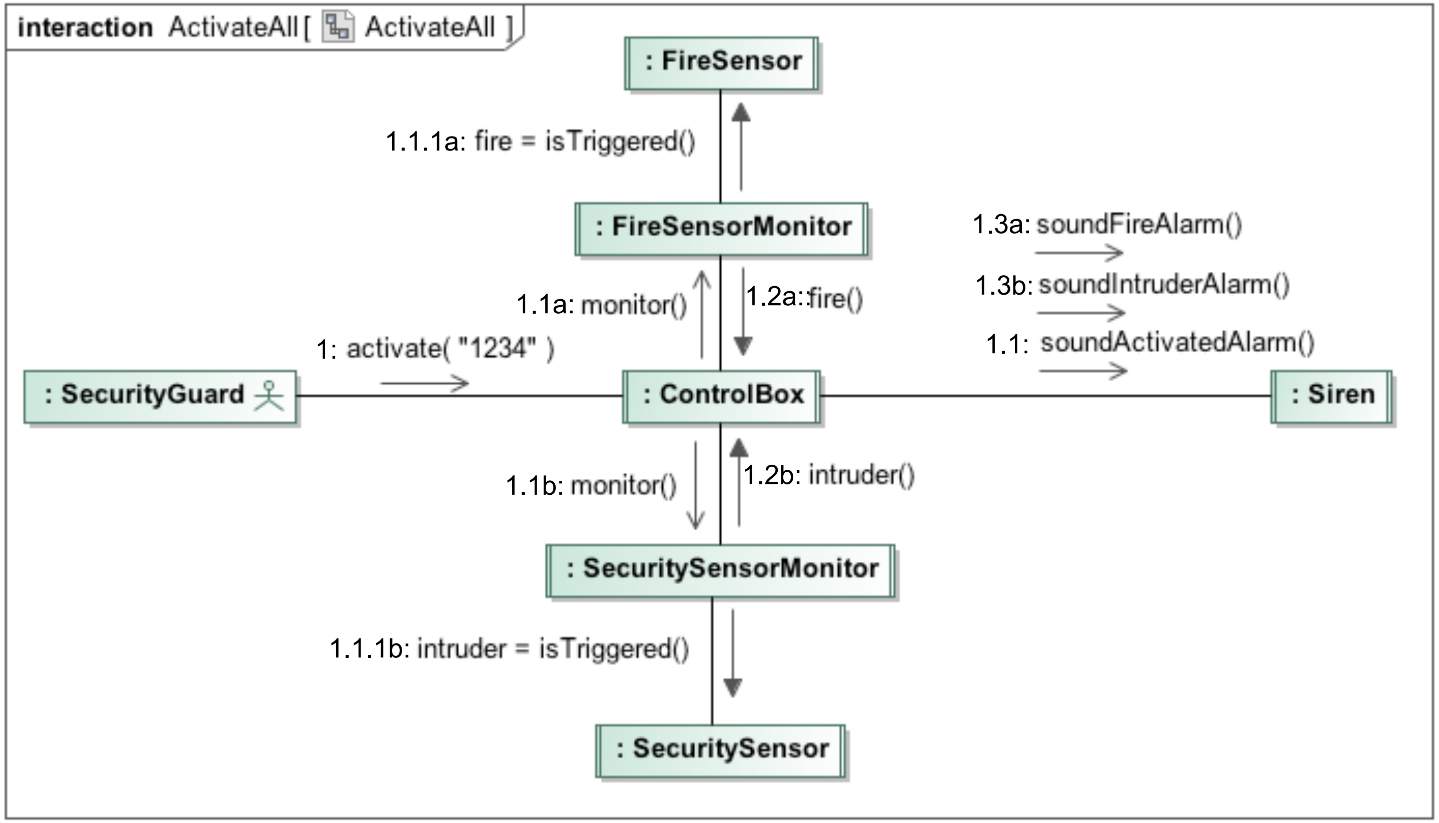


Each of these active classes has objects that have their own threads of control

Concurrency with par



Concurrency with active objects



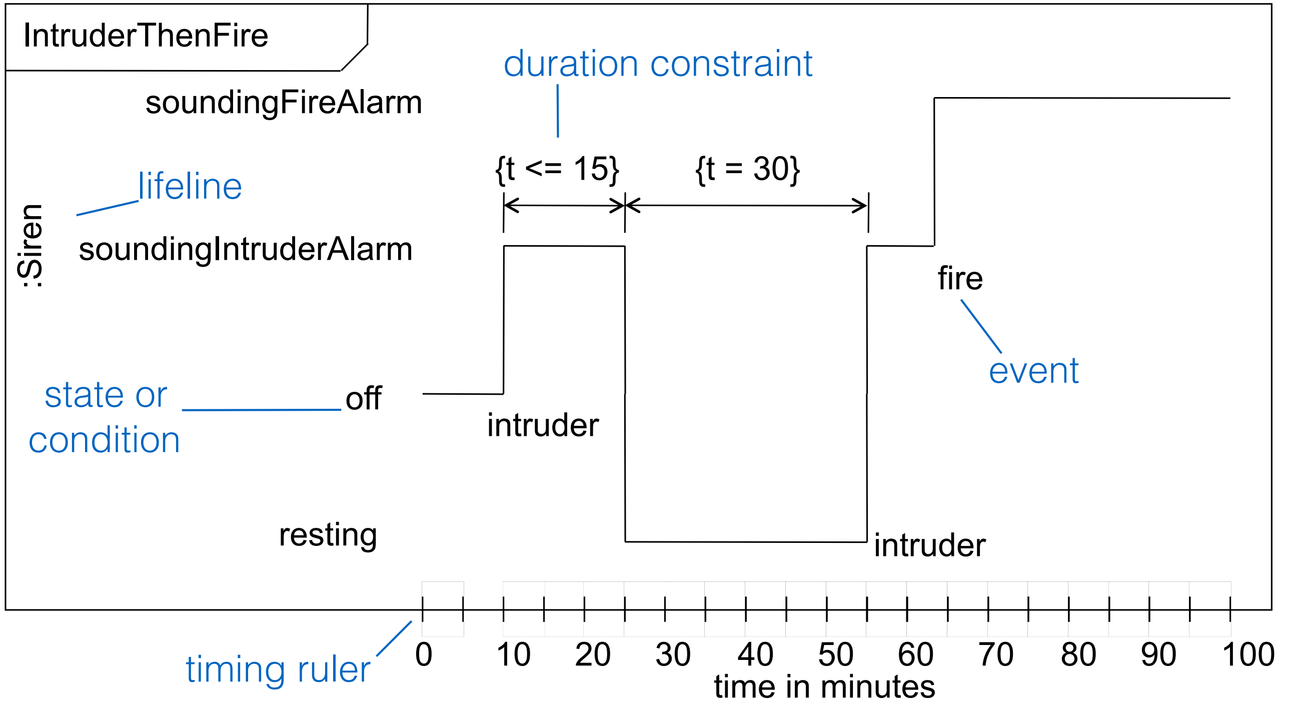
- Each separate thread of execution is given its own label so that messages with different labels execute concurrently, e.g. 1.1a executes concurrently to 1.1b

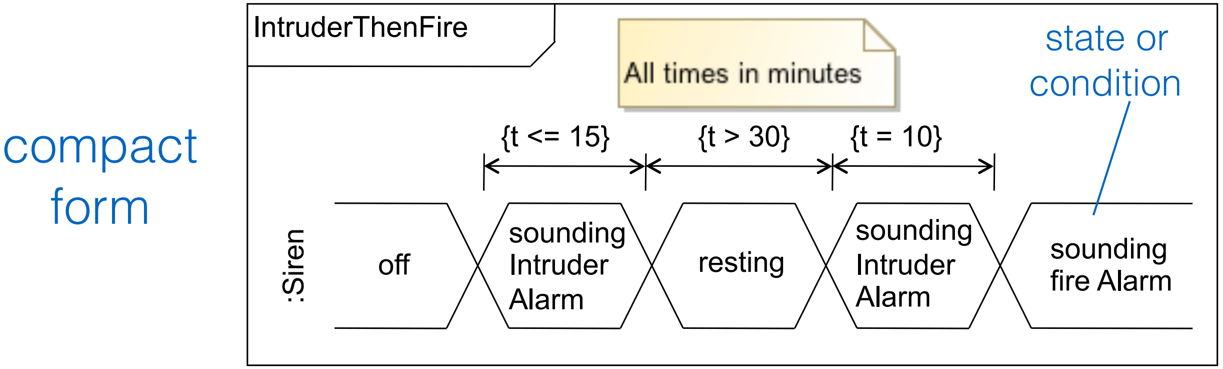
Subsystem interactions

- Sometimes it’s useful to model a use case realization as a high-level interaction between subsystems rather than between classes and interfaces
- Model the interactions of classes within each subsystem in separate interaction diagrams
- You can use interactions diagrams to model the behavior of any behaviored classifier

Timing diagrams

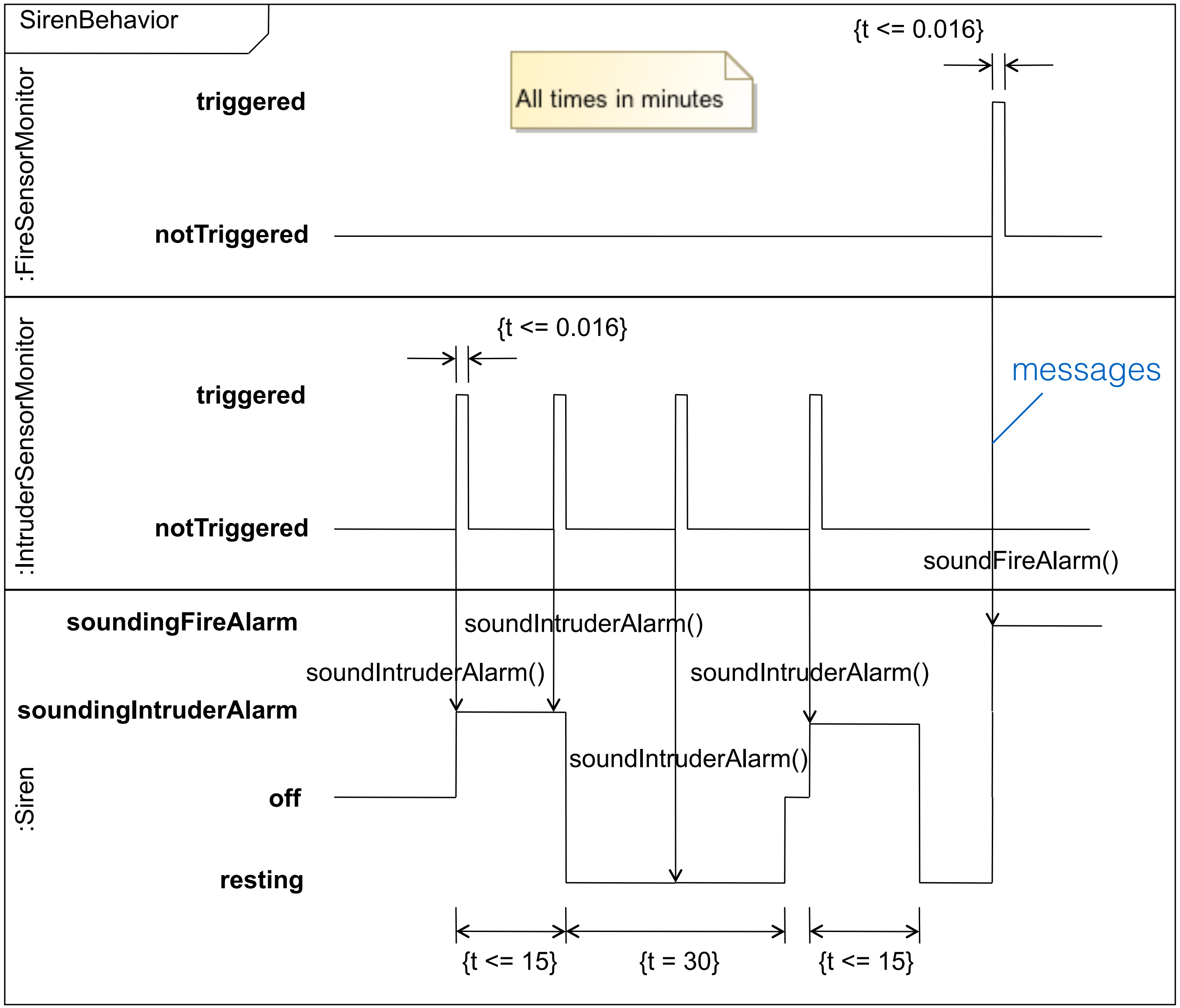
- Emphasize the real-time aspects of an interaction and are used to model timing constraints
- Lifelines, their states or conditions are drawn vertically, time horizontally
- It's important to state the time units you use in the timing diagram





Messages on timing diagrams

- You can show messages between lifelines on timing diagrams
- Each lifeline has its own partition



Example: use case realization - design

- The example is too big to fit on a slide - see Section 20.8 of “UML 2 and the Unified Process”

Summary

- We have looked at:
  - Design sequence diagrams
  - Concurrency in interaction diagrams
  - Timing diagrams