Karlstad University
Department of Information Systems

Adapted for a textbook by Blaha M. and Rumbaugh J.

Object Oriented Modeling and Design

Pearson Prentice Hall, 2005

Modeling Concepts

Remigijus GUSTAS

Phone: +46-54 700 17 65

E-mail: Remigijus.Gustas@kau.se

http://www.cs.kau.se/~gustas/

Object-Oriented System Design

- ✓ **A. The theoretical part**
- ✓ **B. The practical part (mandatory)**
  - Class diagrams
  - State-transition diagrams
  - Interaction diagrams (Use case, Activity and Sequence diagrams)

  - Note: the final report must be finalised prior to the examination date!

Course Literature:

- ✓ M Blaha, J Rumbaugh, Object-Oriented Modelling and Design with UML, second edition, Pearson, 2005.

Complementary Literature:

- ✓ J Martin, J J Odell. Object-Oriented Methods: A Foundation. Second UML edition, Prentice-Hall, New Jersey, 1998.
- ✓ L A Maciaszek. Requirements Analysis and System Design: Developing Information Systems with UML, Addison-Wesley, Harlow, 2001.
- ✓ G Booch, J Rumbaugh, I Jacobson. The Unified Modelling Language User Guide. Addison-Wesley, 1999.

What is Object Orientation (OO) ?

- ✓ OO means organizing software as well as viewing analysis and design models as a collection of discrete objects that incorporate both data structure and behavior.
- ✓ In the previous programming approaches, structure and behavior are loosely connected.
- ✓ UML is the industry standard for specifying, visualizing, constructing and documenting the artifacts of information/software systems.

Features of Object-oriented approach

- Classification
  - Objects with the same structure and behavior are grouped into classes
  - Encapsulation of attributes and operations
- Polymorphism
  - Same operation may behave differently for different classes
- Inheritance
  - Operations and attributes can be inherited from other classes

Objects

- ✓ Object: an entity with a well-defined role in an application
- ✓ Each object has:
  - State: defined by using the attributes, their values, and associations with other objects
  - Behavior: represents how an object acts and reacts
  - Identity (internal and external): uniqueness, no two objects are the same

Classes

- ✓Class: a logical grouping of objects with similar attributes and behavior
- ✓Operation: a function or service provided by all instances of a class
- ✓Encapsulation: the technique of hiding internal implementation details of an object from external view

Types of Operations

- ✓ Constructor/Destructor
  - Creates (Removes) a new instance of a class
- ✓ Query
  - Accesses the state of an object
- ✓ Update
  - Alters the state of an object
- ✓ Scope
  - Applies to a full class rather than an individual instance

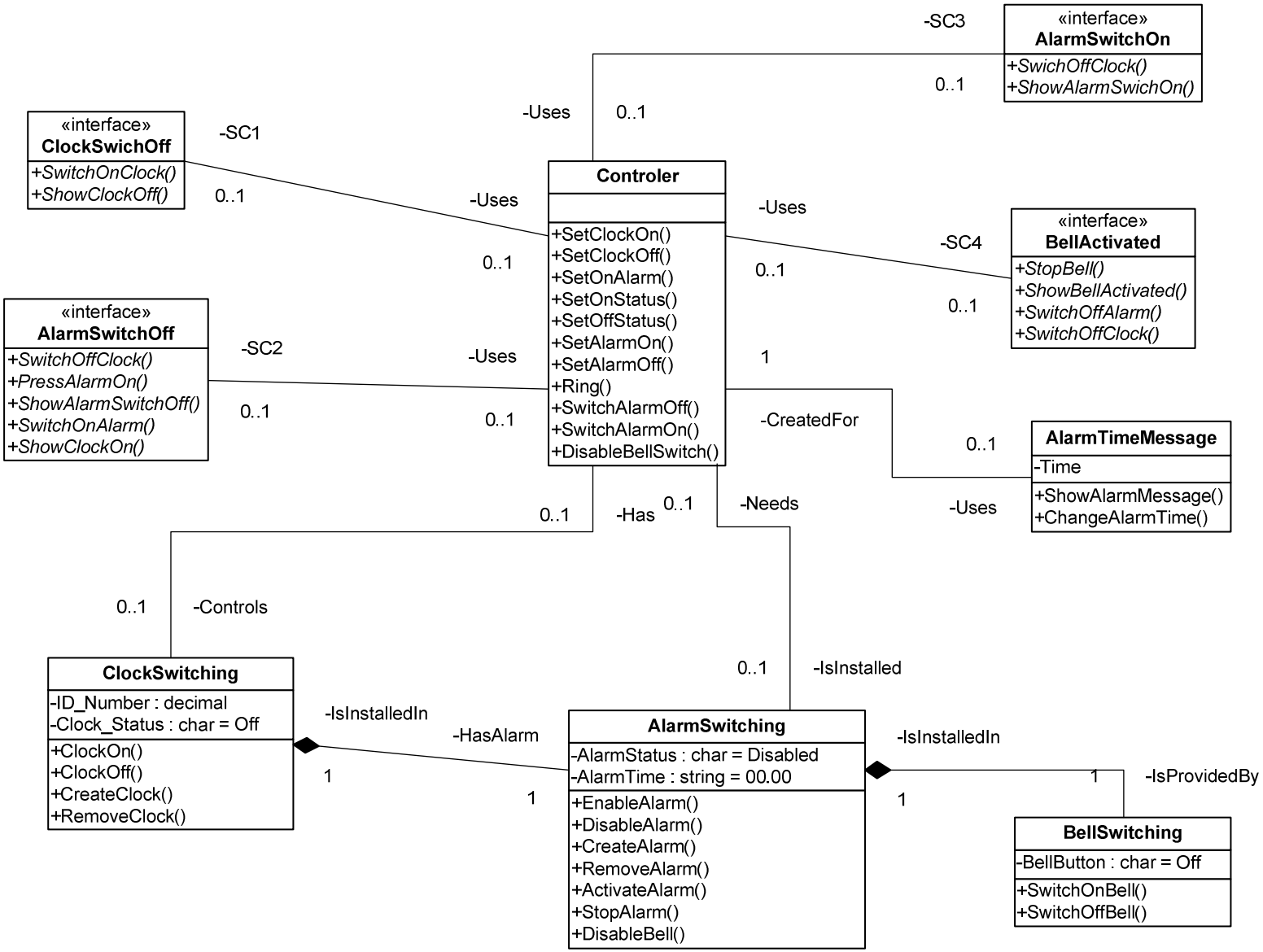
Modeling Concepts, not Implementation

- ✓ Most of OO literature emphasize **implementation** rather than **analysis and design**.
- ✓ Premature focus on implementation restricts design choices. The real payoff in system development comes from **conceptual modeling**.
- ✓ OO modeling is a conceptual process independent of programming language. It is a **way of thinking**, not programming.
- ✓ Greatest benefits come from helping system developers and customers to **communicate** system specification to each other.

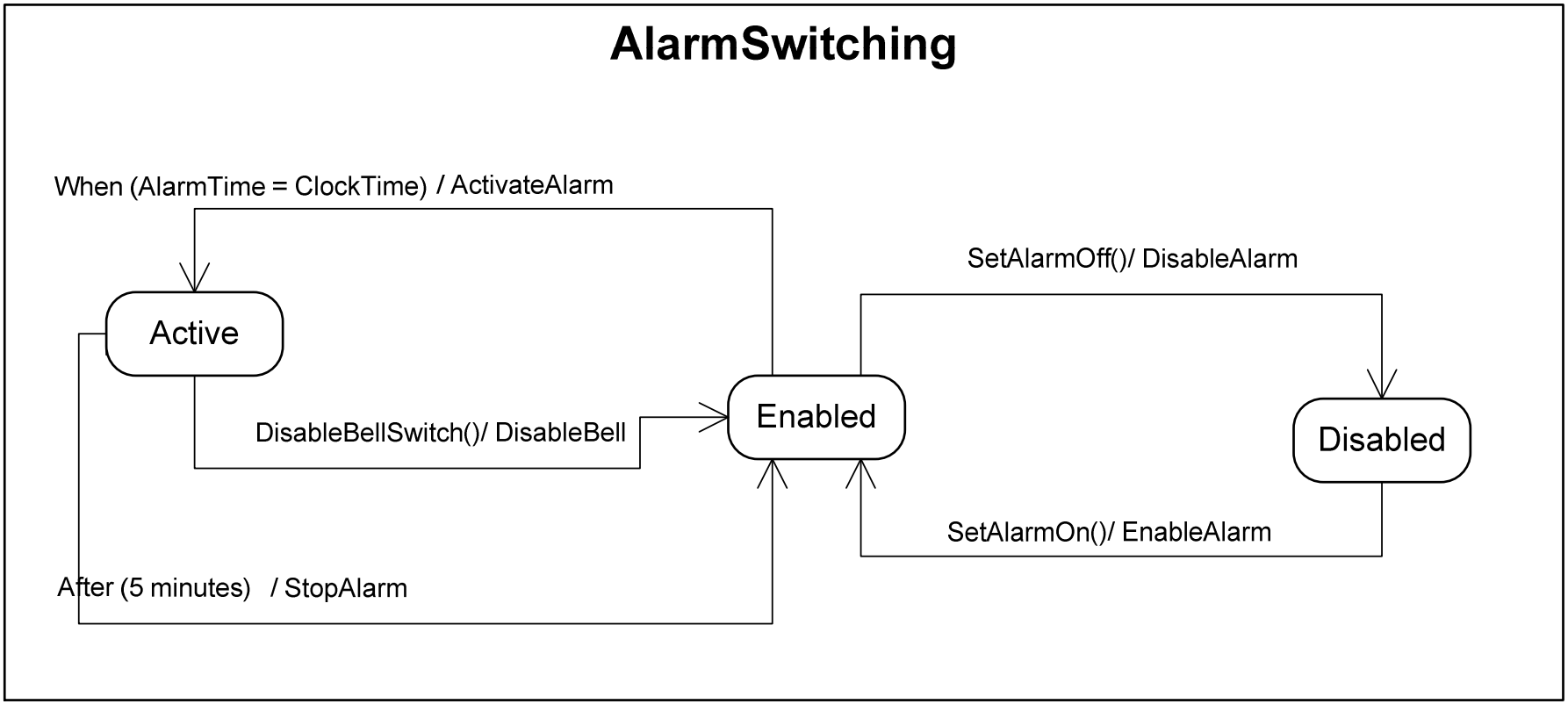
A complete description of system requires three different viewpoints:

- ✓ Class model describes the static structure of objects.
  - Class diagram
- ✓ State model describes the behavioral aspects of an object that change over time.
  - State diagram
- ✓ Interaction model describes how the objects in a system cooperate to achieve user goals.
  - Use case, activity and sequence diagrams

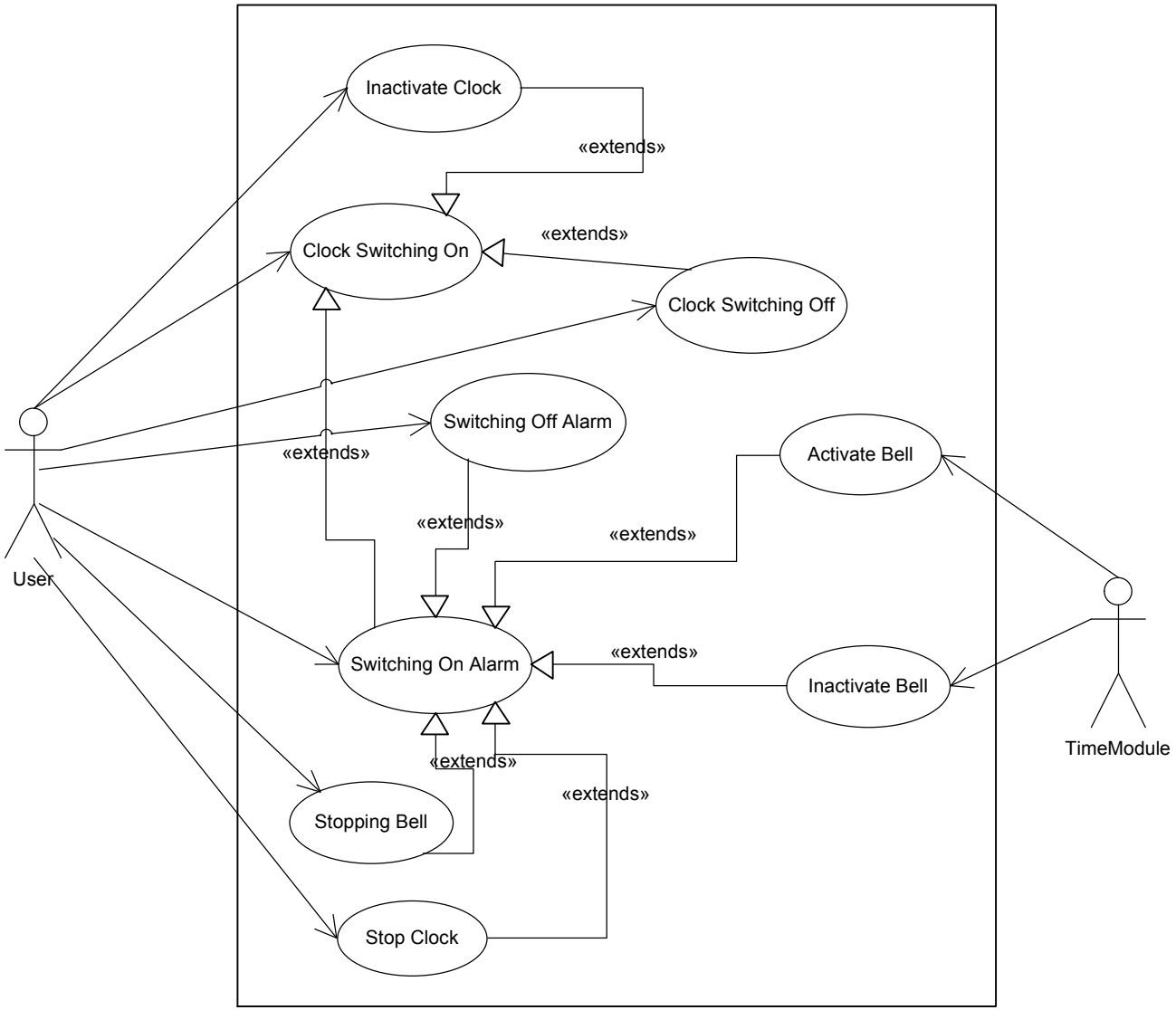
Class diagram



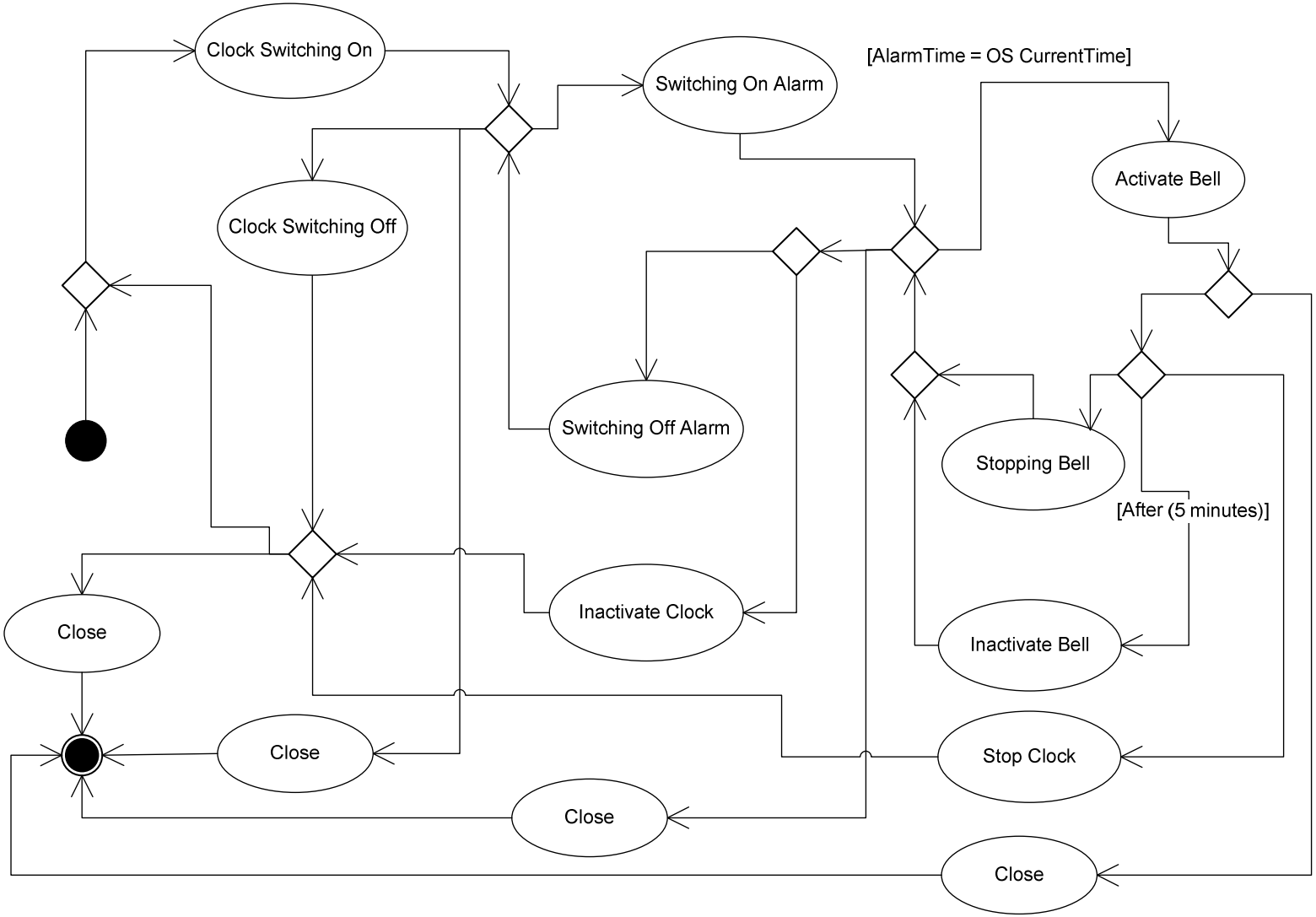
State transition diagram of AlarmSwithing



Use-case diagram

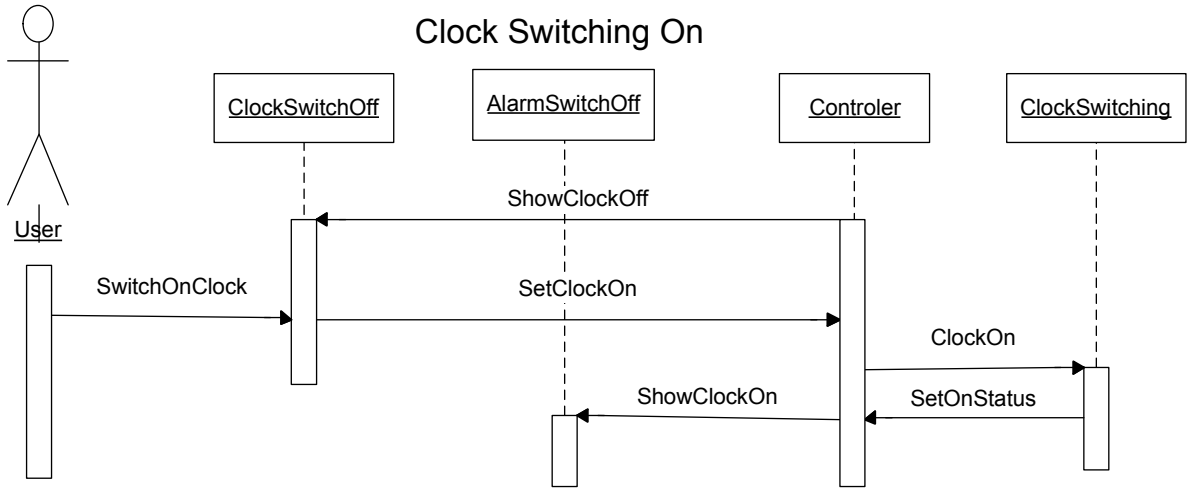


Activity diagram

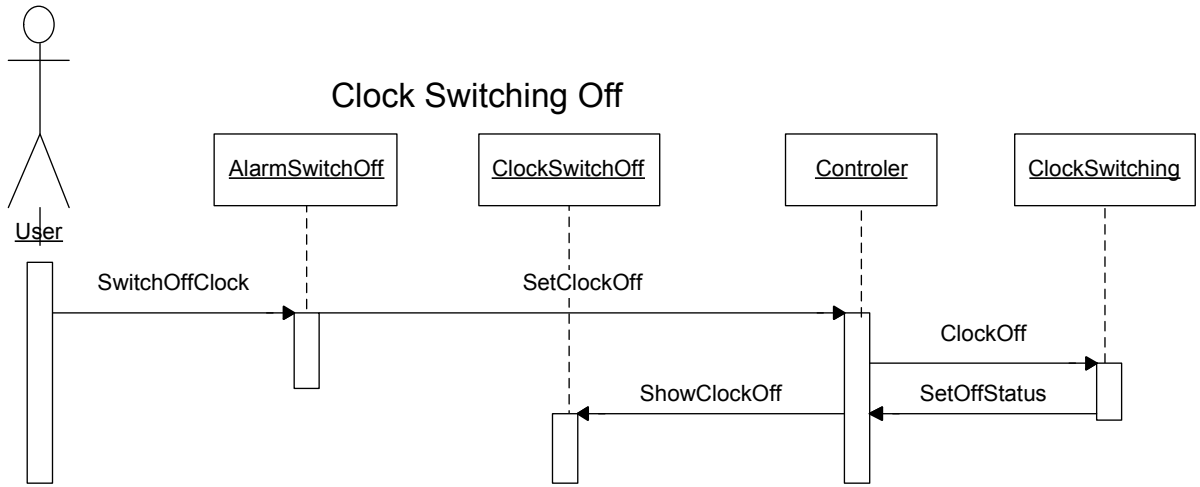


Sequence diagrams

SC1→SC2

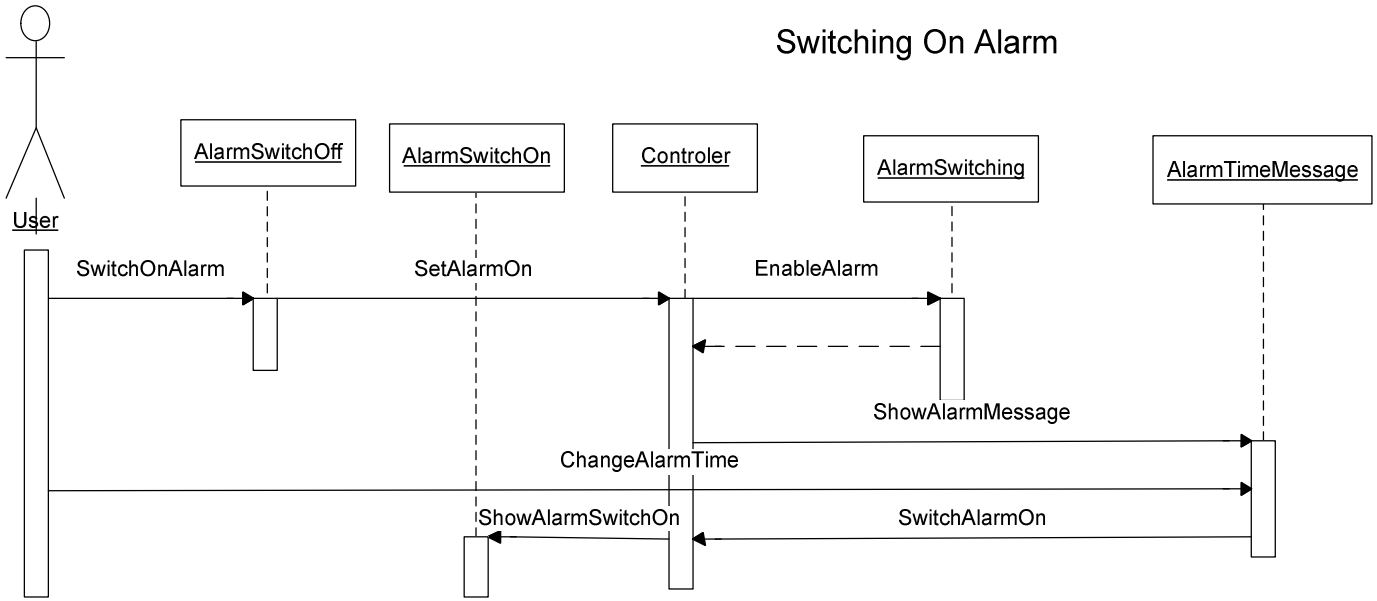


SC2→SC1

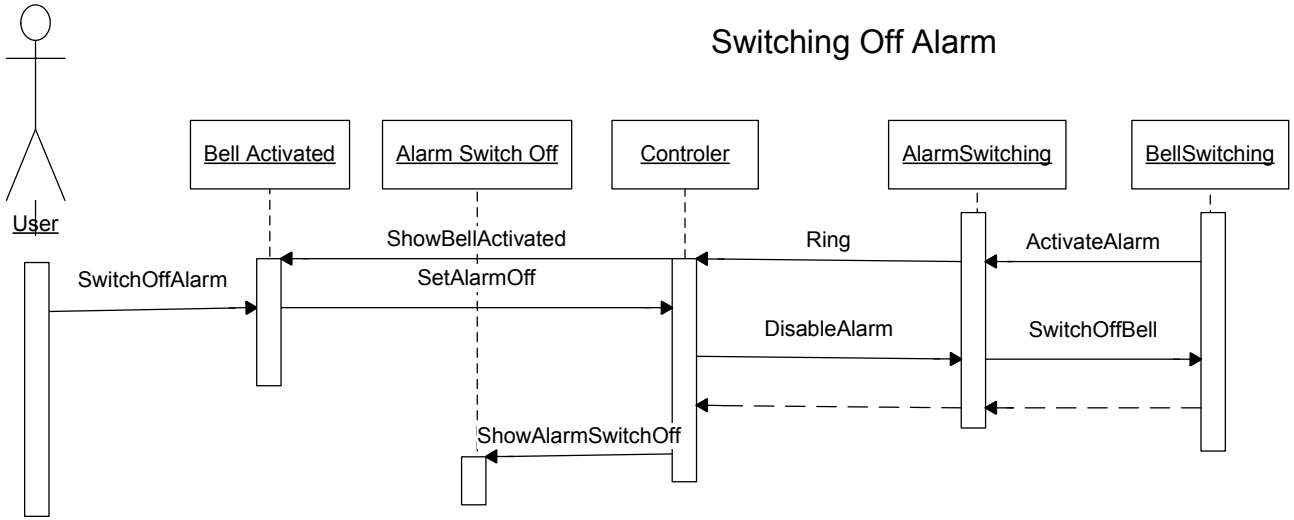


Sequence diagrams

SC2→SC3



SC4→SC2

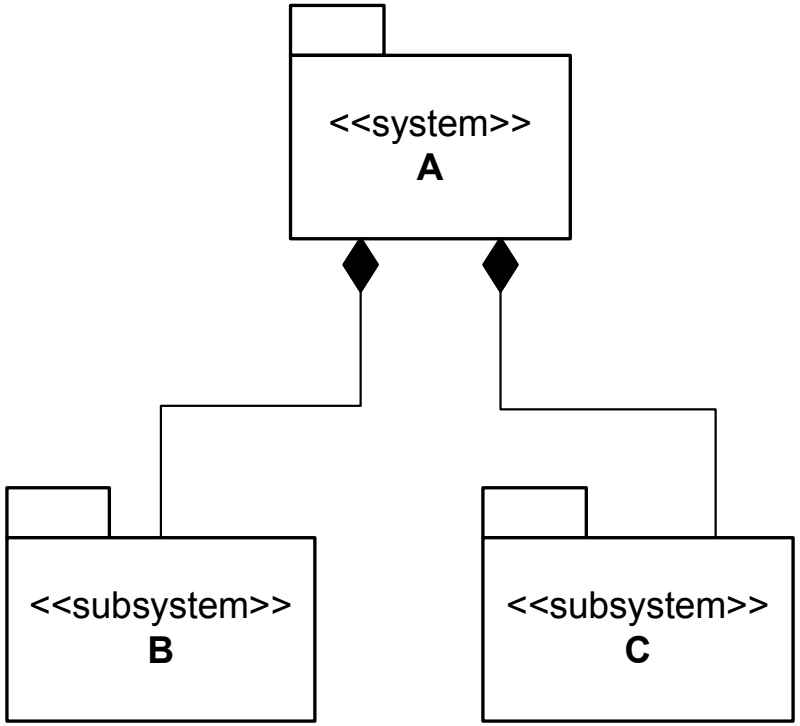


Object-Orientation

- ✓ The Unified Modeling Language™ (UML™) is the industry-standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems, http://www.uml.org/
- ✓ UML defines 12 types of diagrams:
  - Static diagrams (class, object, component, deployment),
  - Dynamic diagrams (use case, sequence, activity collaboration, state transition),
  - Diagrams that help to organize and manage system development process (packages, subsystems, models).

Model of the Architecture

- ✓ The models associated with a system or subsystem completely partition the elements, meaning that every element is owned by exactly one package
- ✓ View is a projection into a model
- ✓ A view typically may not cross system boundary



Modeling Views

- ✓ Model is a special kind of package.
- ✓ Five architecture views are organized into a set of nonoverlapping models

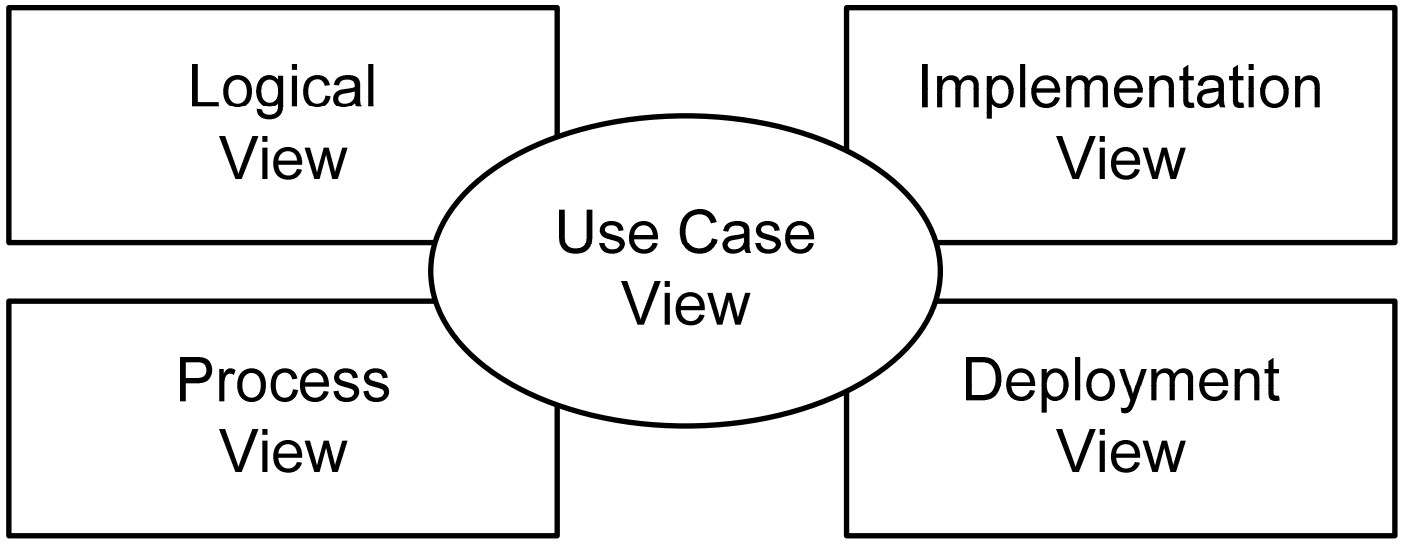


Diagram Types

- ✓ Use Case view
  - Use case diagram
- ✓ Logical view
  - Class diagram
- ✓ Process View
  - State diagram
  - Sequence diagram
  - Activity diagram
- ✓ Implementation view
  - Component (database, software) diagrams
- ✓ Deployment view
  - Hardware topology diagram with components