

# Object-Oriented Systems Engineering Method (OOSEM)

## Analyze System Requirements

# OOSEM Topics

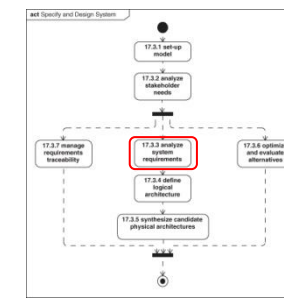
- OOSEM Overview
- Method
  - Setup Model
  - Analyze Stakeholder Needs
  - **Analyze System Requirements**
  - Define Logical Architecture
  - Synthesize Candidate Physical Architectures
  - Optimize and Evaluate Alternatives
  - Manage Requirements Traceability
  - Integrate and Verify System
- Summary

# Module Objectives

- After completion of this module, student should understand
  - The primary modeling artifacts from Analyze System Requirements
  - How to specify the functions and i/o of a system based on the scenario analysis
  - How an ibd can be used to capture the system external interfaces
  - The features of a black box system specification
  - How a state machine is used to specify the behavior of the system
  - The need to identify the design constraints
  - How to analyze variation in requirements

# Motivation

- Specify the system requirements in a concise manner
  - Every extra requirement costs
  - Every missed requirement costs
  - Ambiguity costs
- Deficiencies in system requirements propagate to large numbers of derived element and component requirements



# Analyze System Requirements

- Perform scenario analysis for each enterprise use case
- Specify the system requirements in terms of its input and output responses and other black box features
  - Interfaces
  - Functions
  - States
  - Performance, physical and quality characteristics
  - Stores

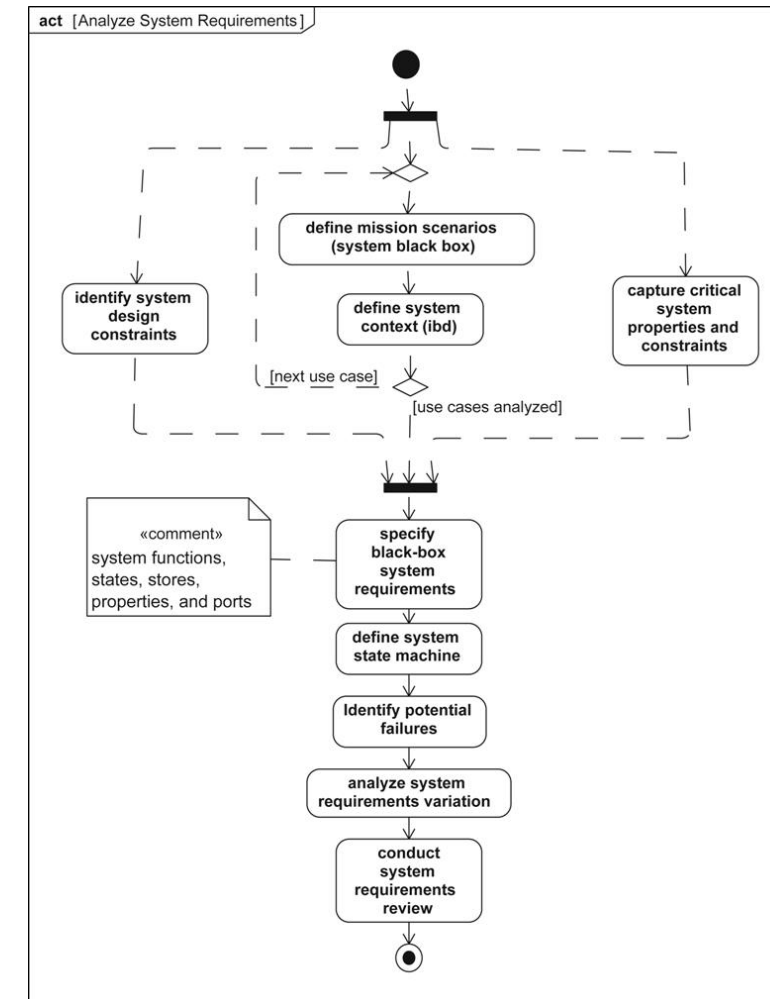


Figure 17.14

# Define Mission Scenarios

- Activity diagram specifies how system interacts with external systems and users for each use case scenario
- Partitions represent the system, external systems and users
- Actions in partition specify what the corresponding entity must or will do
- Nodes connected by object flows represent inputs and outputs (I/O)
- Constraints can be shown as pre- and post-conditions on actions
- Off nominal paths can also be specified using decision nodes

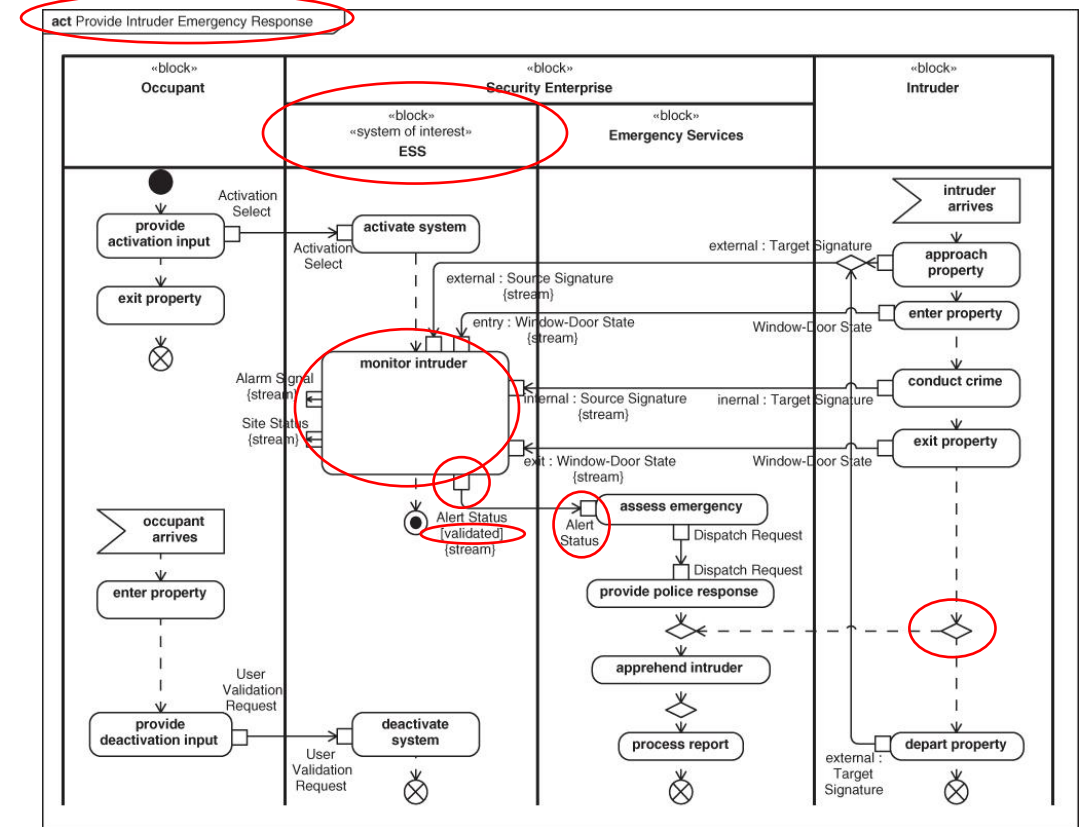
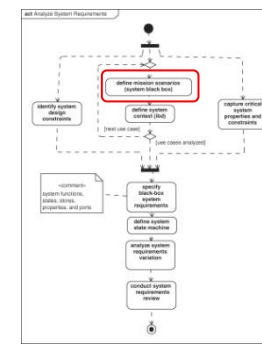


Figure 17.14

# Define System Inputs & Outputs



- Inputs and Outputs defined for each activity
- Identify commonality among I/O to simplify design

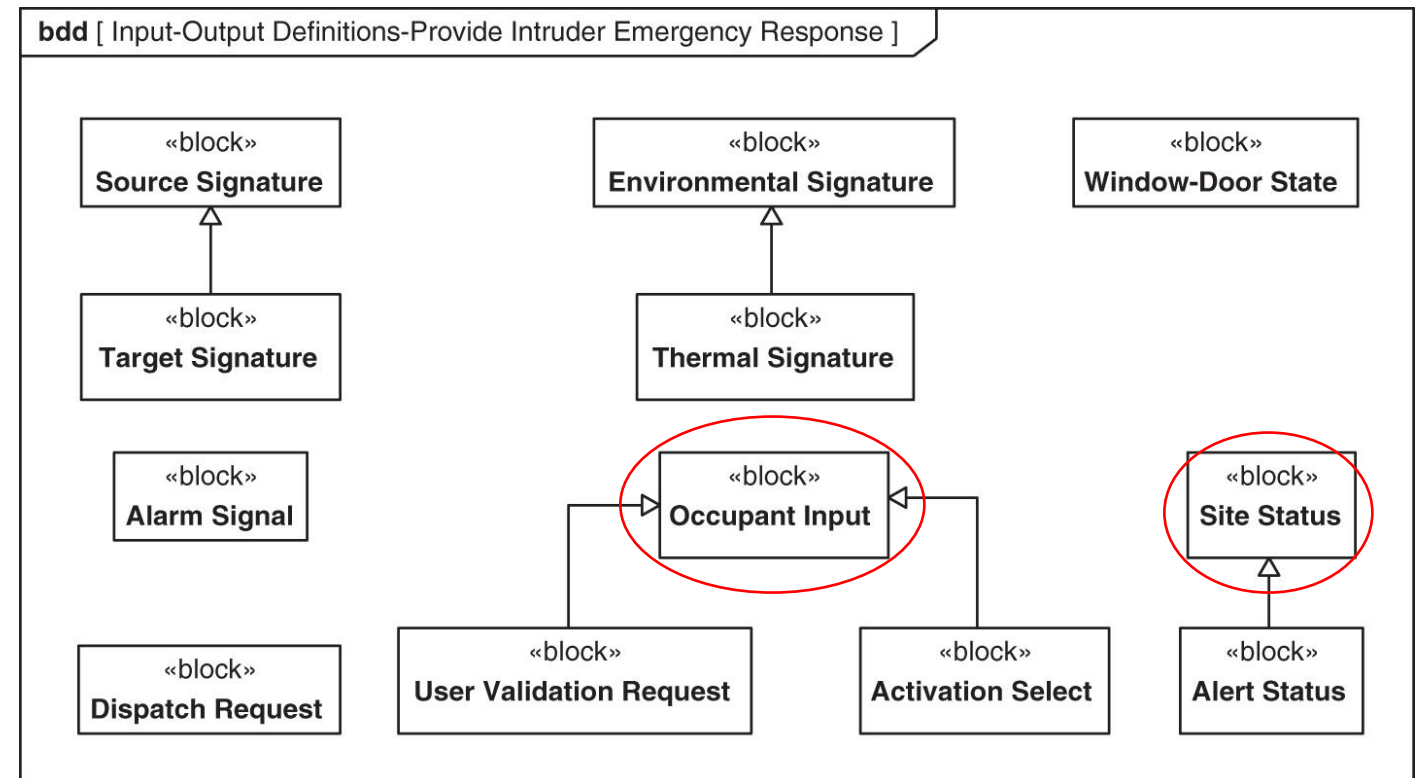


Figure 17.15

# Define System Context

- Specifies the required interfaces between the system and the external systems and/or users
  - Inputs/Outputs from activity diagrams are allocated to item flows
  - Ports specify the external interfaces

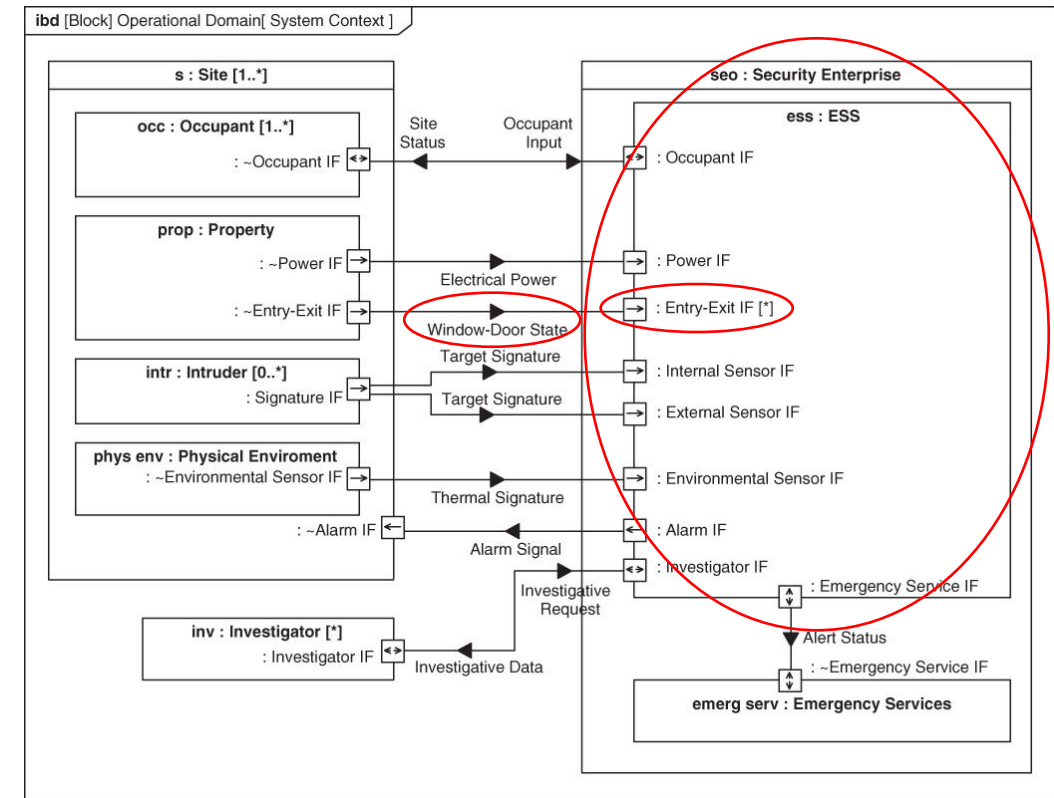
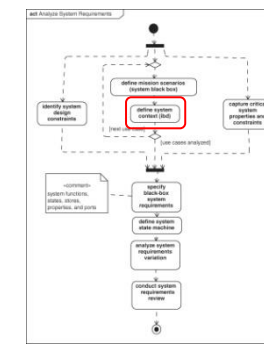
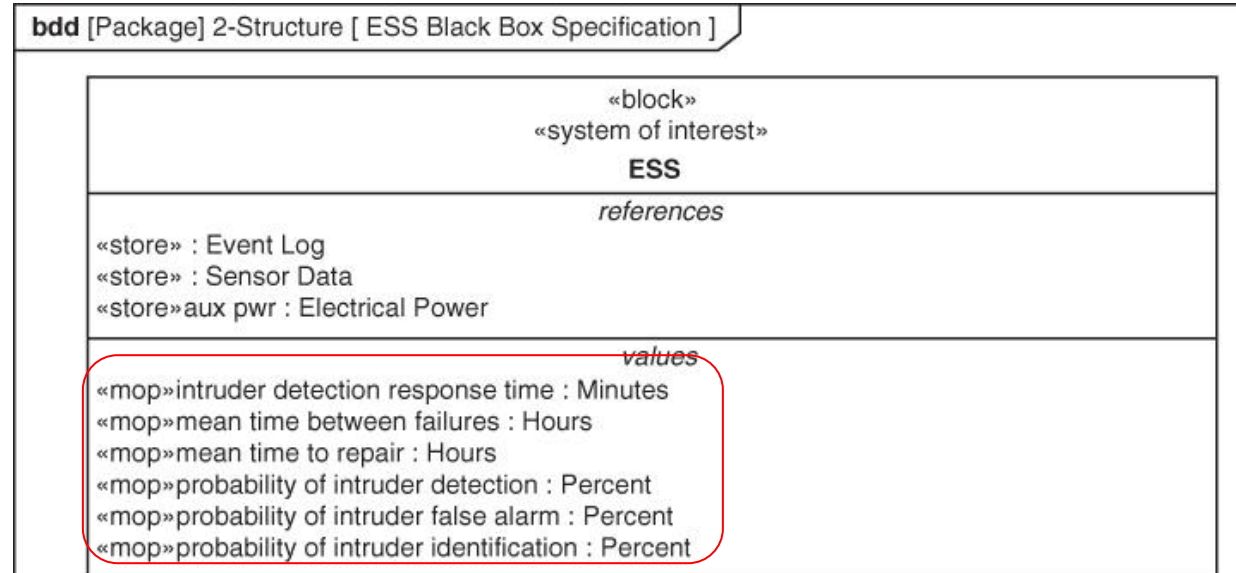


Figure 17.16



# Capture Critical System Properties

- Critical performance, physical, and other non functional requirements can be captured as value properties of the system block (e.g., intruder detection response time)
  - Derived from analysis models for each MOE using parametrics



**Figure 17.18**

# Specify Black-Box System Requirements

- Based on the scenario analysis, interface definitions, and engineering analyses
- System's externally observable behavior and physical characteristics
- Black-box specification features include:
  - Functions (i.e., operations, activities)
  - Interfaces (i.e. ports)
  - Performance, physical, and quality characteristics (i.e. value properties)
  - Required items that must be stored (i.e. reference properties)
  - States triggered by events

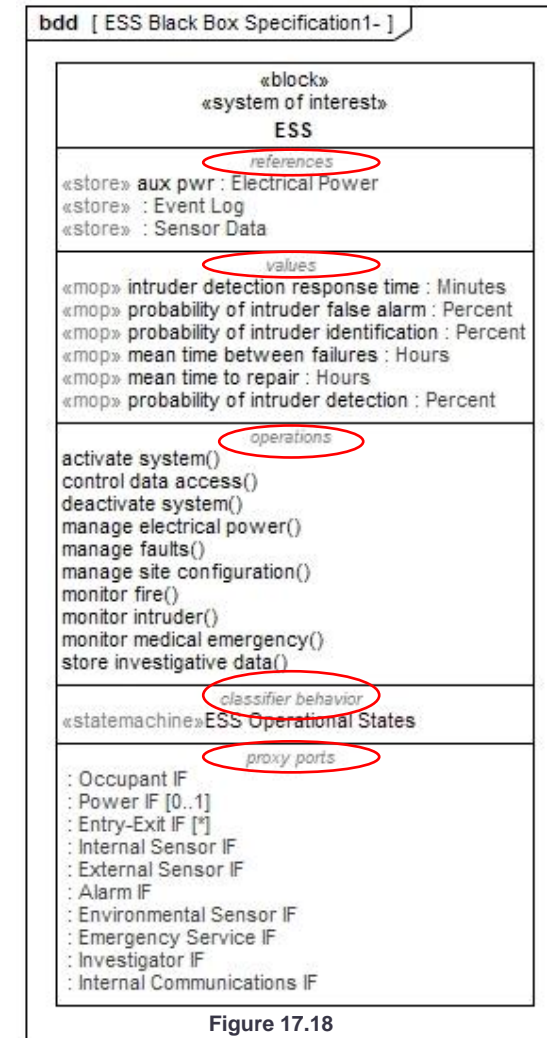
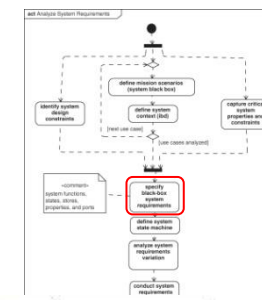


Figure 17.18

# Define System State Machine

- Defines how the system responds to events and how the system's behavior changes in different states
- Represents a composite behavior across multiple scenarios
- Can be used to specify requirements such as:
  - If an input event occurs while in the current state and guard conditions are satisfied, then system shall transition to the next state and execute the specified actions within performance constraints

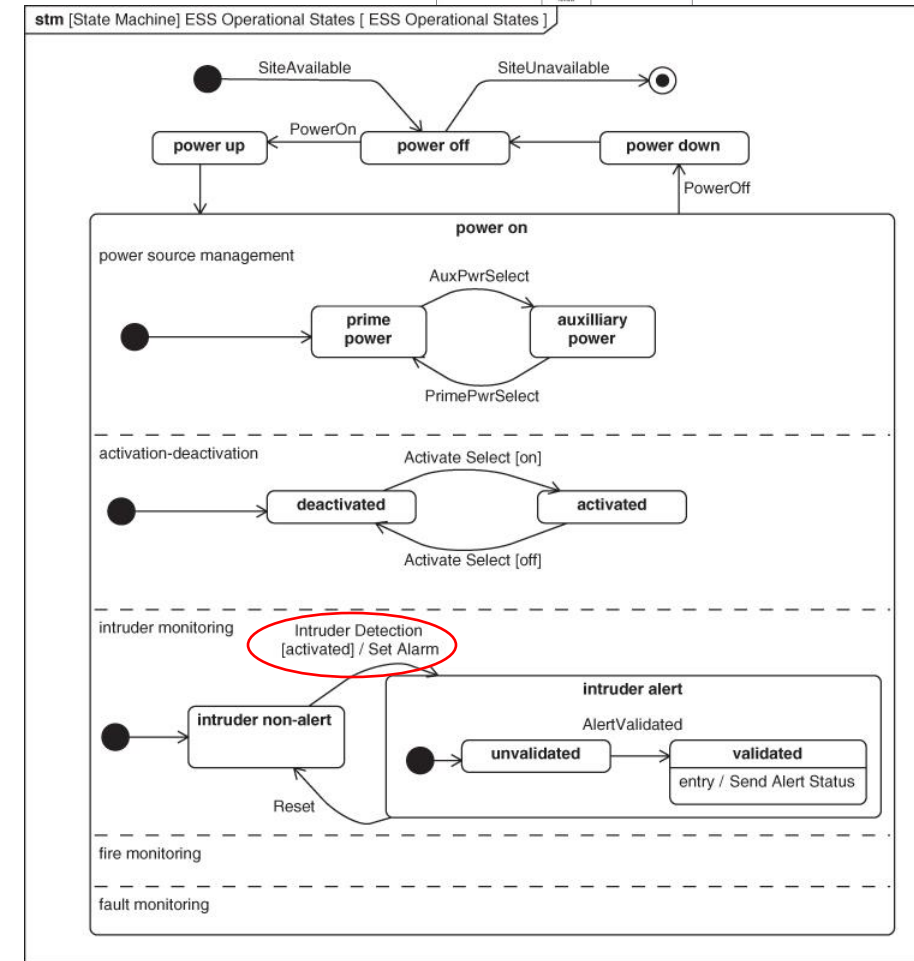
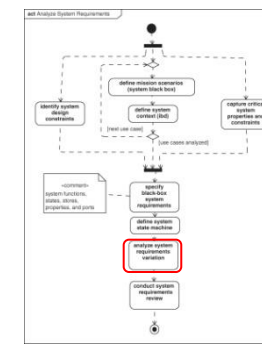


Figure 17.19



# Identify System Failure Modes

- A failure is a violation of a constraint
- A failure can be caused by another failure

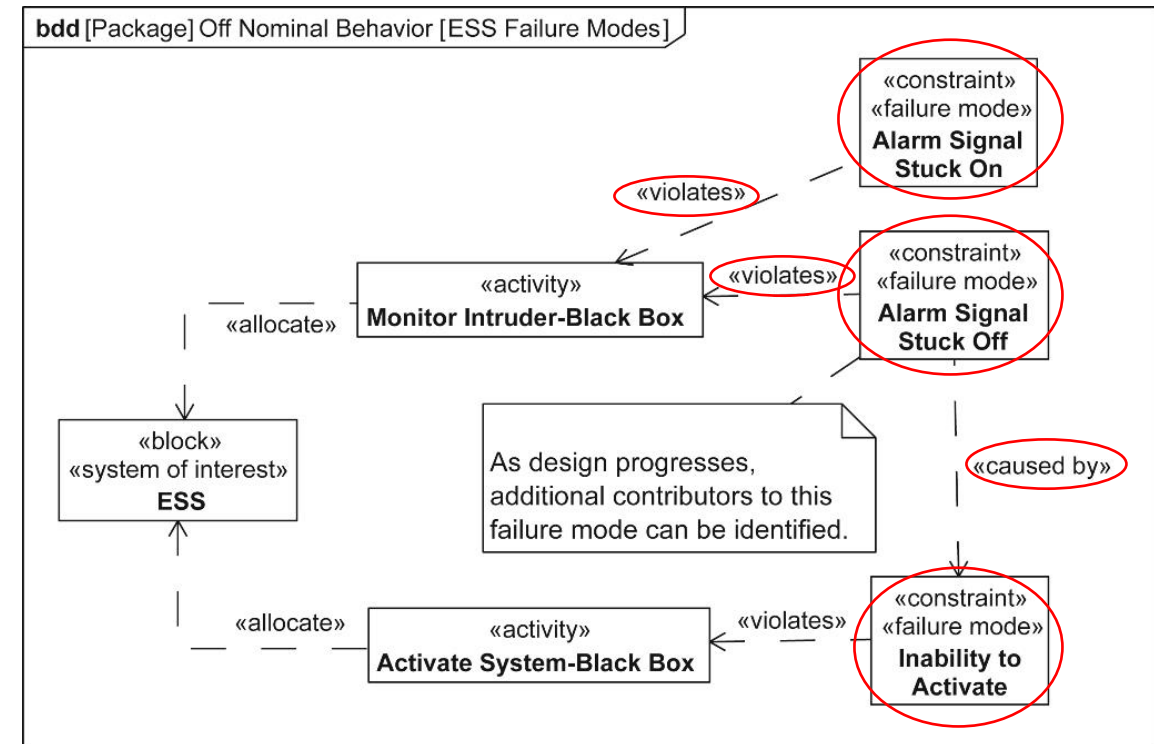
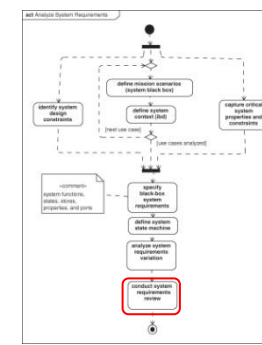


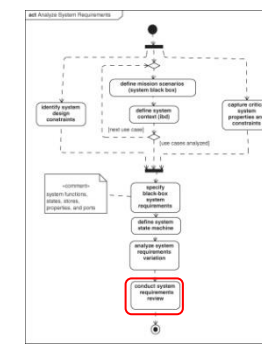
FIGURE 17.21

# Analyze System Requirements Variation



- Requirements Variation Analysis is an evaluation of the probability that a requirement will change due to:
  - Changing interfaces
  - System growth
  - Increased performance
  - New functionality
  - COTS or Technology upgrades
  - Changes in physical location
- Input to Risk Analysis & Mitigation and Variant modeling

# Analyze System Requirements Variation (cont.)



- Assess each system black box feature
  - Identify corresponding requirement
  - Define probability of change for each requirement variation (High, Medium, Low)
  - Evaluate criticality of change (e.g., impact on MOEs and stakeholder value)
- Provide input to risk analysis
  - Risk measure = probability of change \* criticality
- Mitigation
  - Design system or family of systems to accommodate potential requirements variation

Probability	Change
H	Interface with Senior Citizen patient monitoring equipment
M	Customer coverage beyond current region
H	Increased remote video monitoring capacity and storage requirements
H	Automated software distribution
M	Web based billing and payments
H	Requirement for reduced false alarm rate





# Summary

- System requirements are specified as a system ‘black-box’
  - Functions, interfaces, properties, stores, state machine
- Behavioral requirements derived from use case scenarios
- External interface requirements represented in system context ibd and defined on i/o bdd
- Performance, physical and other non-functional requirements captured as value properties
- State Machines specify a system’s overall behavior
- Failure modes identified for system black box
- Requirements variation analysis based on a risk methodology contributes to a more robust and flexible design
- Design constraints identified, validated, & imposed on architecture