Modeling Event-Based Behavior with State Machines (Part 1 – SysML Concepts)

Content Developer

OBJECT MANAGEMENT GROUP • CERTIFICATION PROGRAM •

SM

Section Objectives

- In this Section, you will learn:
  - How to model State Machine Diagrams in SysML

Overview

- This section will discuss:
  - State Machine Diagram Concepts
    - Why model State Machines?
    - State Machine Diagram Components
    - How State Machines are modeled
  - State Machine modeling for In-Class Project

Why Model State Machines?

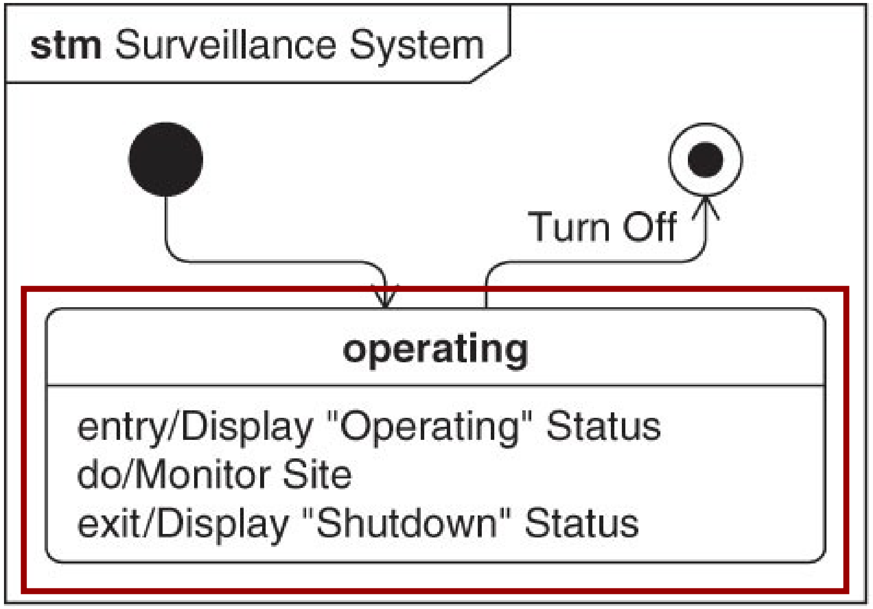
- Used to model how a block changes state
  - State – a significant condition in the life of a block
  - Example: Light: ON or OFF
- State Machines describe how a block transitions from one state to another
- Helps to define what happens when entering or exiting a state
- Graphical depiction of states, transitions, and events
- Clarification, Elaboration, Communication

State Machine Diagram Components

- State Machine diagrams can be comprised of the following:
  - States (Includes Initial and Final States)
    - Behaviors
  - Transitions
    - Triggers
    - Guards
    - Effects
  - Composite States
    - Single Region
    - Multiple Regions
  - Pseudo-states
    - History
    - Forks and Joins
    - Choice

States

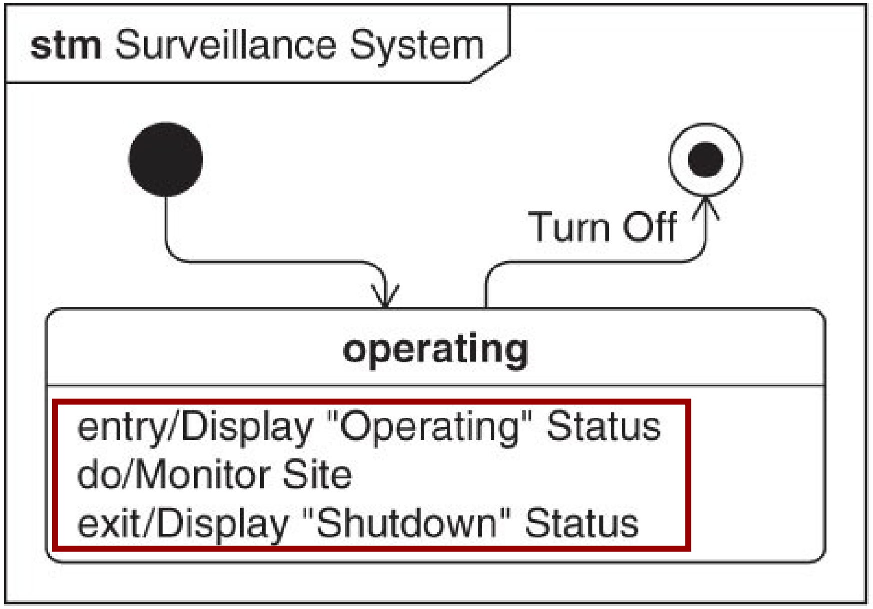
- States – represents a condition in the life of a block
  - Name the state with the name of the condition (e.g. Operating)
- Initial State – represented by a black solid dot
- Final State – represented by a bulls-eye


stm Surveillance System
Turn Off
operating
entry/Display "Operating" Status
do/Monitor Site
exit/Display "Shutdown" Status

© 2008 Elsevier, Inc.: A Practical Guide to SysML

Behaviors

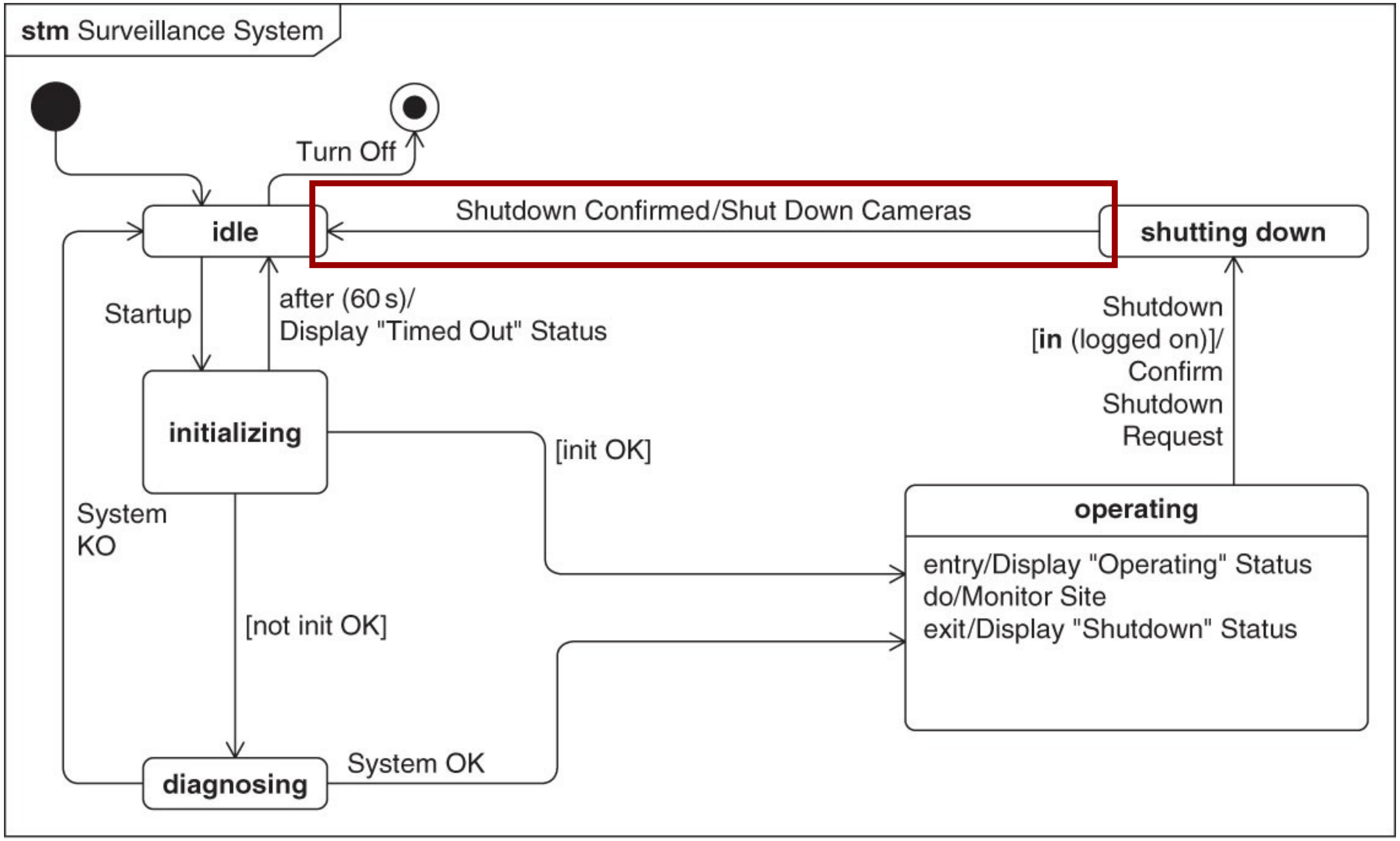
- Actions of a State
  - Types:
    - Entry – what happens when the state is entered
    - Exit – what happens when the state is exited
    - Do – what happens while in a state


stm Surveillance System
Turn Off
operating
entry/Display "Operating" Status
do/Monitor Site
exit/Display "Shutdown" Status

© 2008 Elsevier, Inc.: A Practical Guide to SysML

Transitions

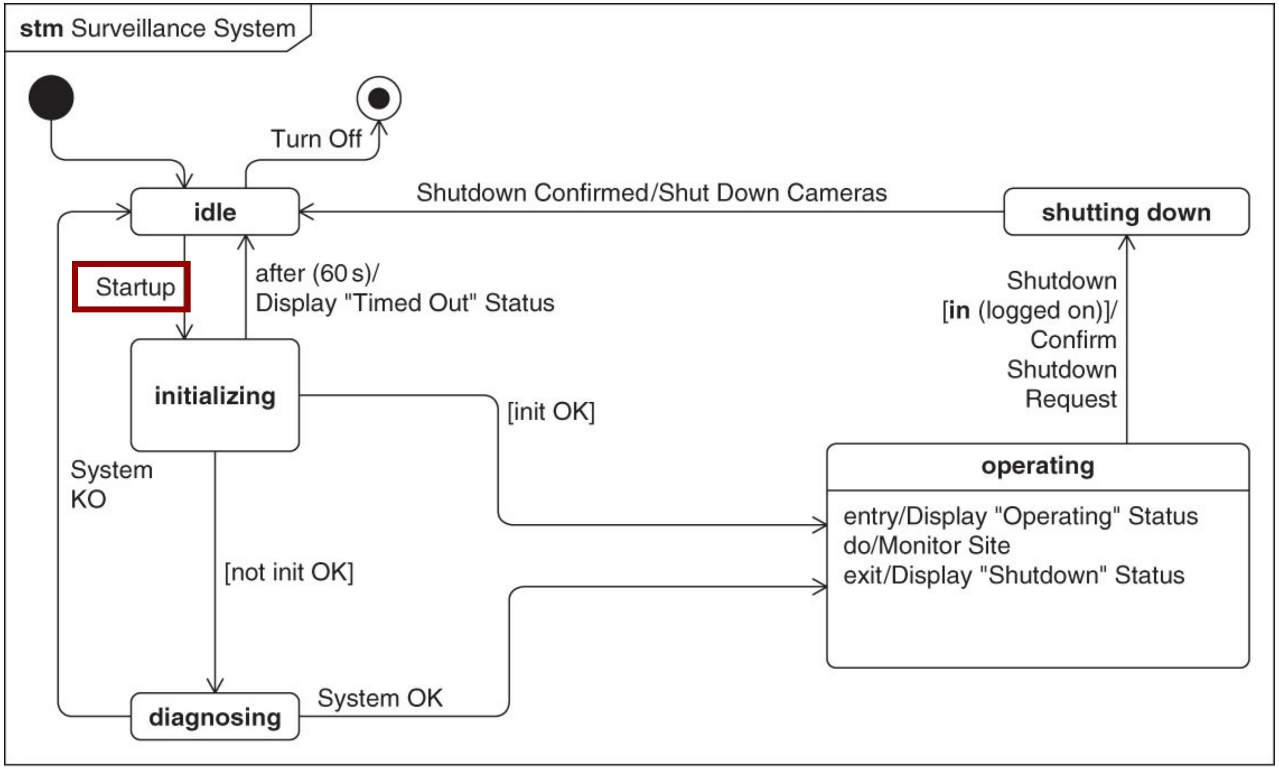
- Used to show the flow from one state to another (solid arrow)
- Can consist of triggers, guards, and effects



© 2008 Elsevier, Inc.: A Practical Guide to SysML

Triggers (Events)

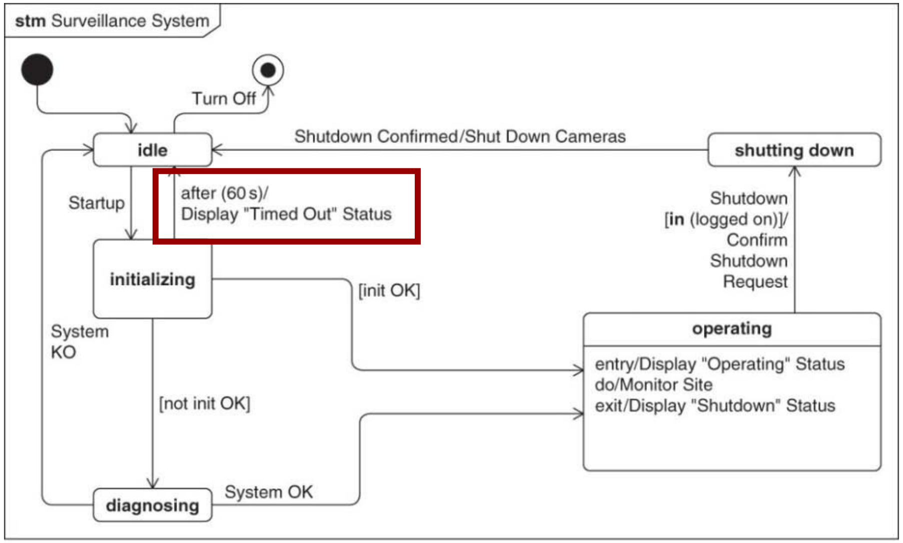
- The Event that causes the transition to fire
- Indicates what causes the state to change
- Depicted with text on a transition



© 2008 Elsevier, Inc.: A Practical Guide to SysML

Event Types

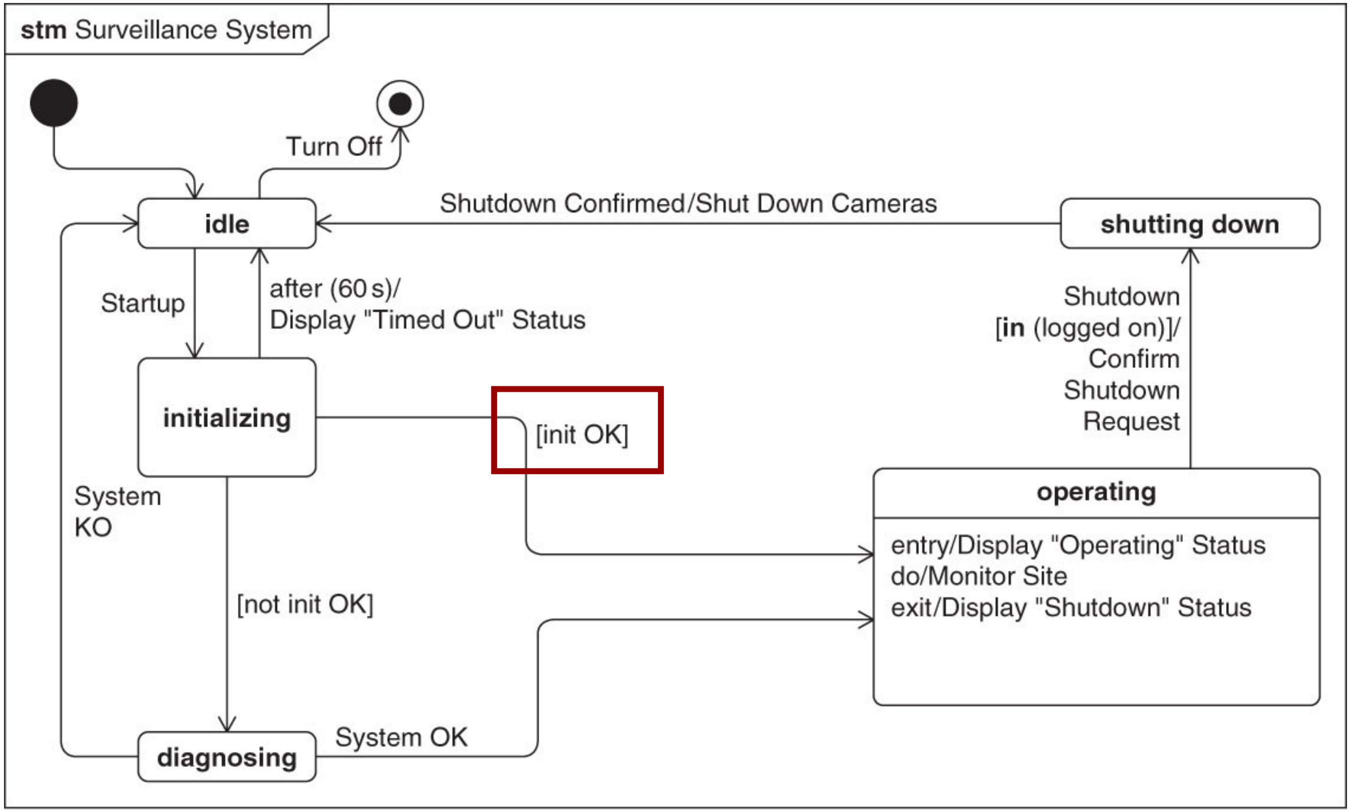
- **Signal Events – indicate a new asynchronous message has arrived**
- **Time Events – indicate that either a given time interval has passed since the current state was entered, or that a given instant in time has been reached (Keyword: ‘after’ or ‘at’)**
- **Change Events – indicate that some condition has been satisfied (Keyword: ‘when’)**
- **Call Events – indicate that an operation on the state machine’s owning block has been requested.**



© 2008 Elsevier, Inc.: A Practical Guide to SysML

Guards

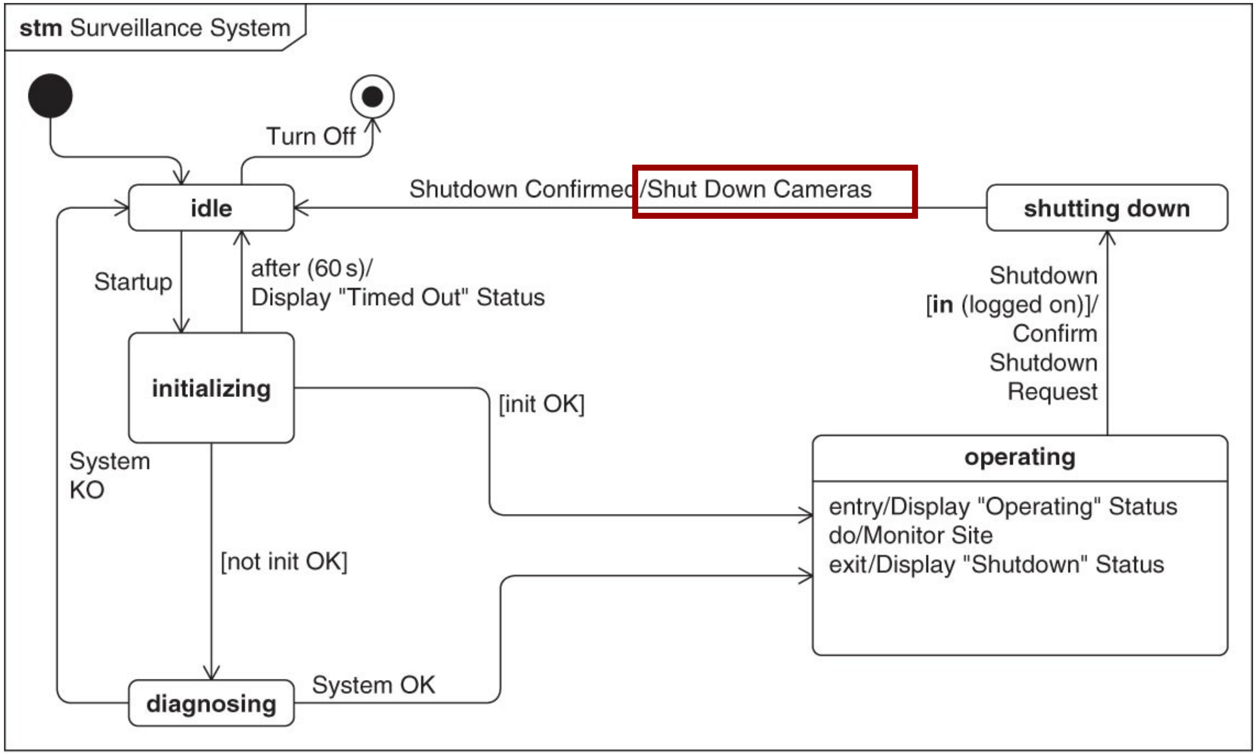
- An expression that must be true for transition to occur
- Contained in brackets [ ] on a transition



© 2008 Elsevier, Inc.: A Practical Guide to SysML

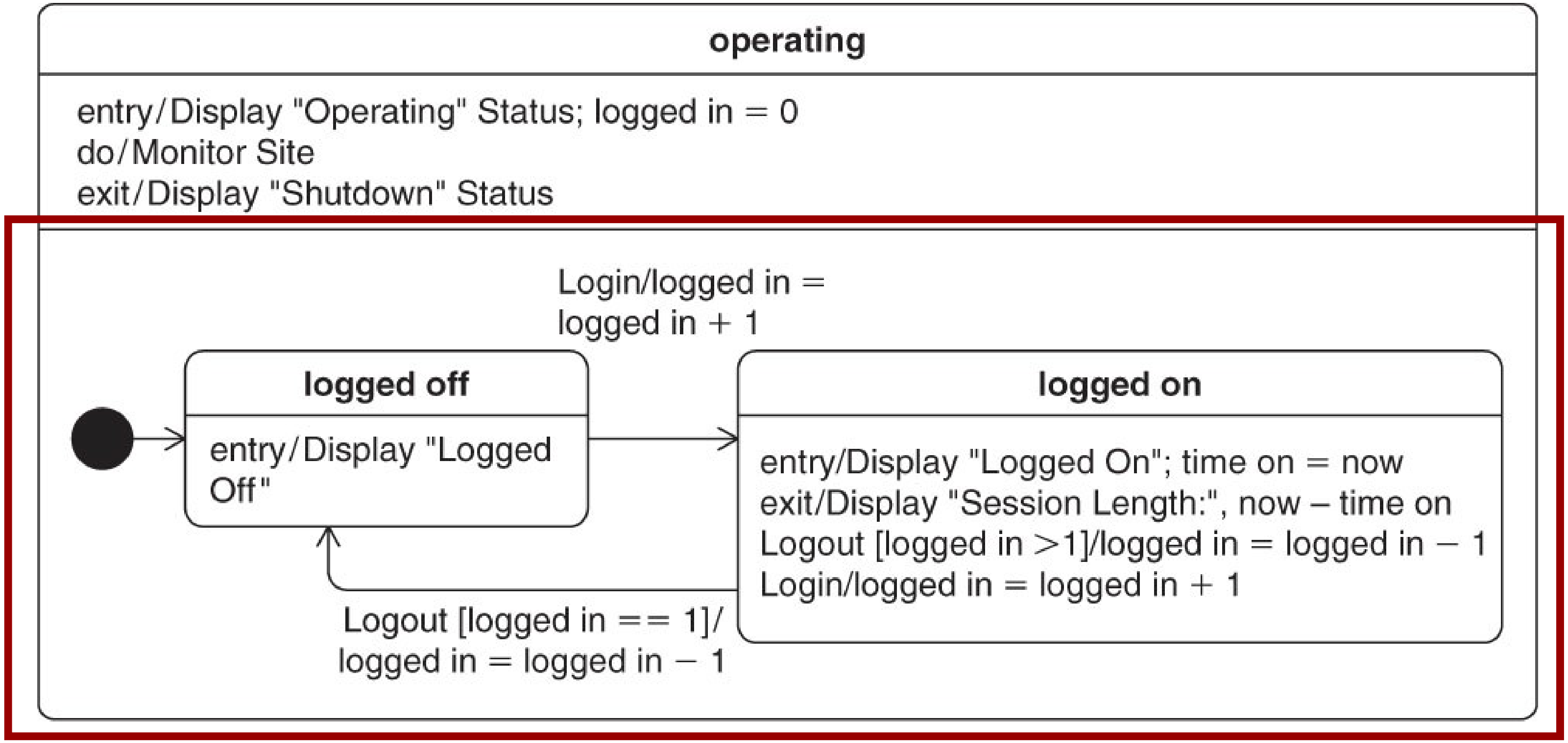
Effects (Behaviors)

- Describes the behavior that is executed when an event occurs, (i.e. the behavior that executes during transition from one state to another)
- Follows a '/' on a transition



Composite States – Single Region

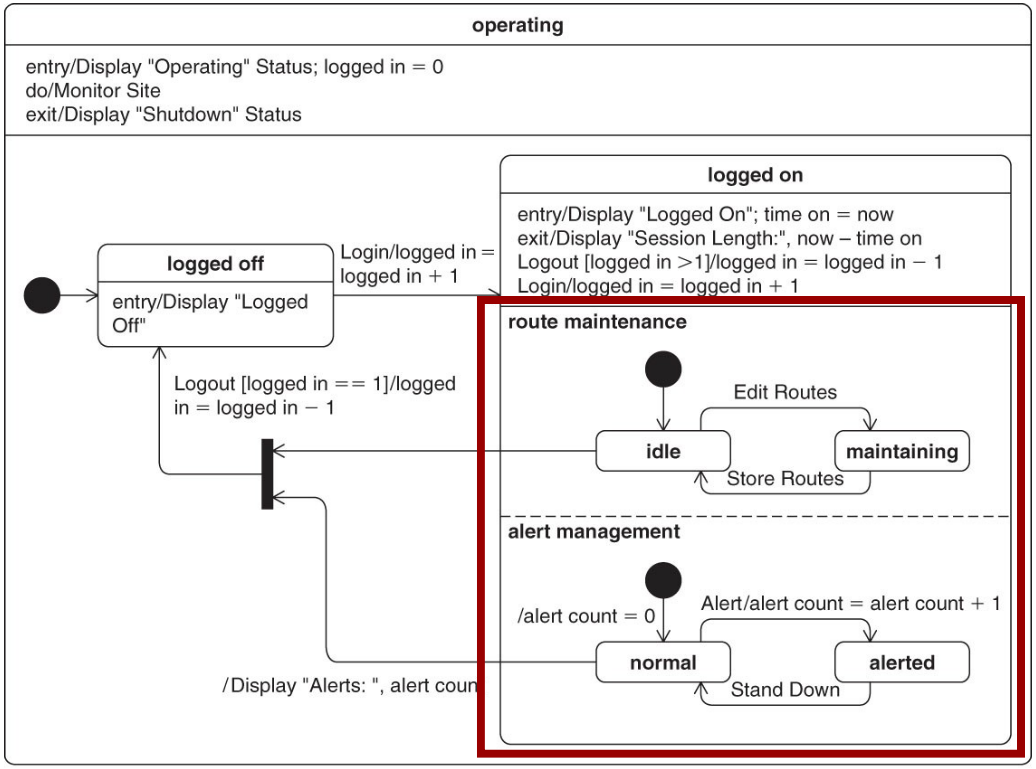
- Means of depicting the hierarchy of states
- Sub-states – states that are unique to another state of an entity
- Composite States are depicted by enclosing sub-states within a state


operating
entry/Display "Operating" Status; logged in = 0
do/Monitor Site
exit/Display "Shutdown" Status
Login/logged in = logged in + 1
logged off
entry/Display "Logged Off"
logged on
entry/Display "Logged On"; time on = now
exit/Display "Session Length:", now – time on
Logout [logged in >1]/logged in = logged in – 1
Login/logged in = logged in + 1
Logout [logged in == 1]/ logged in = logged in – 1

© 2008 Elsevier, Inc.: A Practical Guide to SysML

Composite States – Multiple (Orthogonal) Regions

- States can have more than one region
  - Used to show concurrent states
  - Each region defines a set of states
  - The states in a region are exclusive



operating
entry/Display "Operating" Status; logged in = 0
do/Monitor Site
exit/Display "Shutdown" Status
logged off
entry/Display "Logged Off"
Login/logged in = logged in + 1
logged on
entry/Display "Logged On"; time on = now
exit/Display "Session Length:", now – time on
Logout [logged in >1]/logged in = logged in – 1
Login/logged in = logged in + 1
route maintenance
Edit Routes
idle
maintaining
Store Routes
Logout [logged in == 1]/logged in = logged in – 1
alert management
/alert count = 0
Alert/alert count = alert count + 1
normal
alerted
Stand Down
/Display "Alerts: ", alert coun

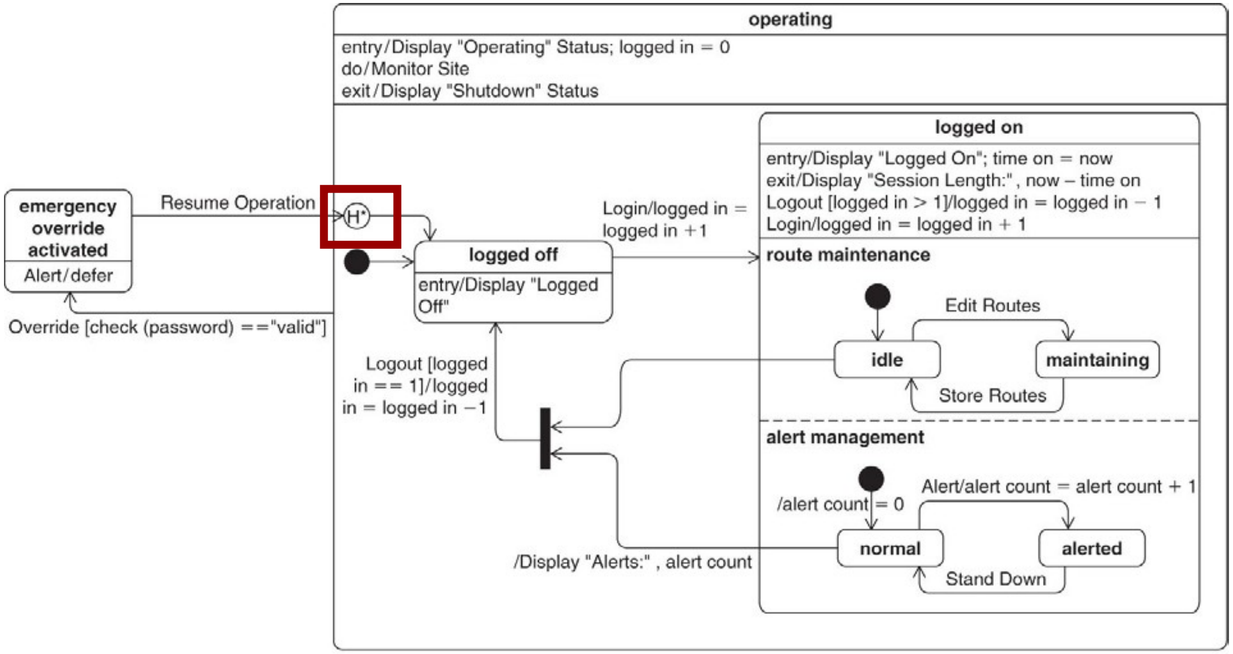
© 2008 Elsevier, Inc.: A Practical Guide to SysML

Pseudo-states

- Elements that exist to determine the next active state in a state machine
  - History
    - Shallow
    - Deep
  - Forks and Joins
  - Choice

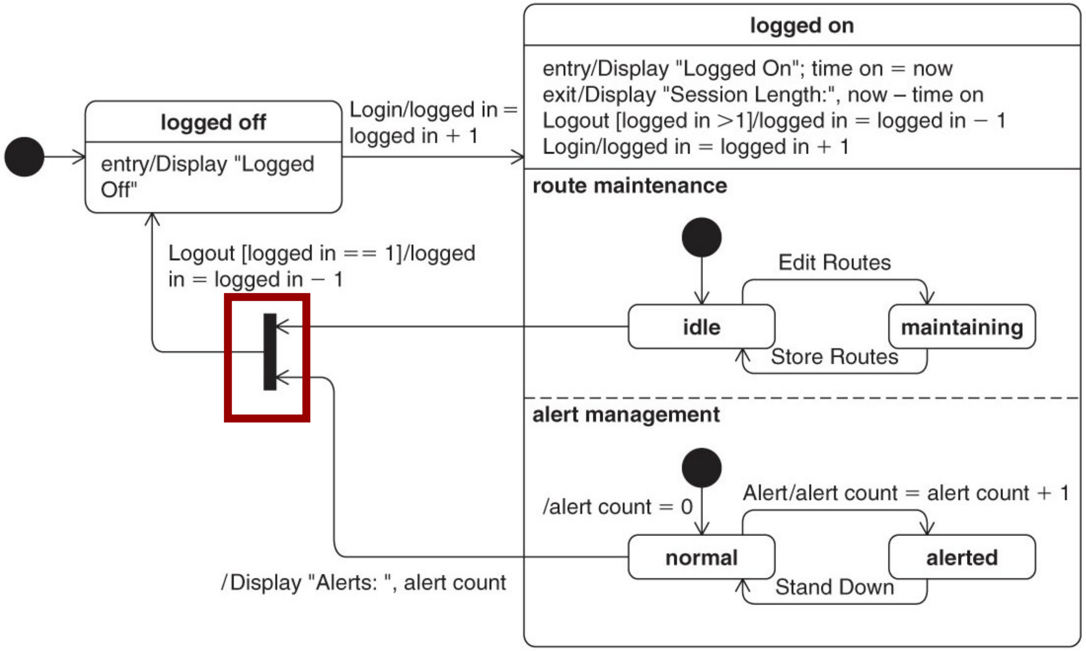
History (Shallow and Deep)

- Denotes that the active sub-state will be remembered when the block transitions out of a composite state
- The block will then return to that sub-state when the block transitions back into that composite state
- Shallow History – remembers only the top-level state of the region
  - Depicted by the letter H surrounded by a circle
- Deep History – remembers the top-level state and the sub-states at all levels of nesting as well
  - Depicted by H* surrounded by a circle


operating
entry/Display "Operating" Status; logged in = 0
do/Monitor Site
exit/Display "Shutdown" Status
emergency override activated
Alert/defer
Resume Operation
H*
Override [check (password) =="valid"]
logged off
entry/Display "Logged Off"
Login/logged in = logged in +1
Logout [logged in == 1]/logged in = logged in −1
logged on
entry/Display "Logged On"; time on = now
exit/Display "Session Length:" , now – time on
Logout [logged in > 1]/logged in = logged in − 1
Login/logged in = logged in + 1
route maintenance
Edit Routes
idle
maintaining
Store Routes
alert management
/alert count = 0
Alert/alert count = alert count + 1
normal
alerted
Stand Down
/Display "Alerts:" , alert count

Forks and Joins

- Fork Node
  - Has one incoming transition, and multiple outgoing transitions
  - Outgoing transitions lead to different orthogonal states
- Join Node
  - Has multiple incoming transitions, and one outgoing transition
  - Incoming transitions come from different orthogonal states

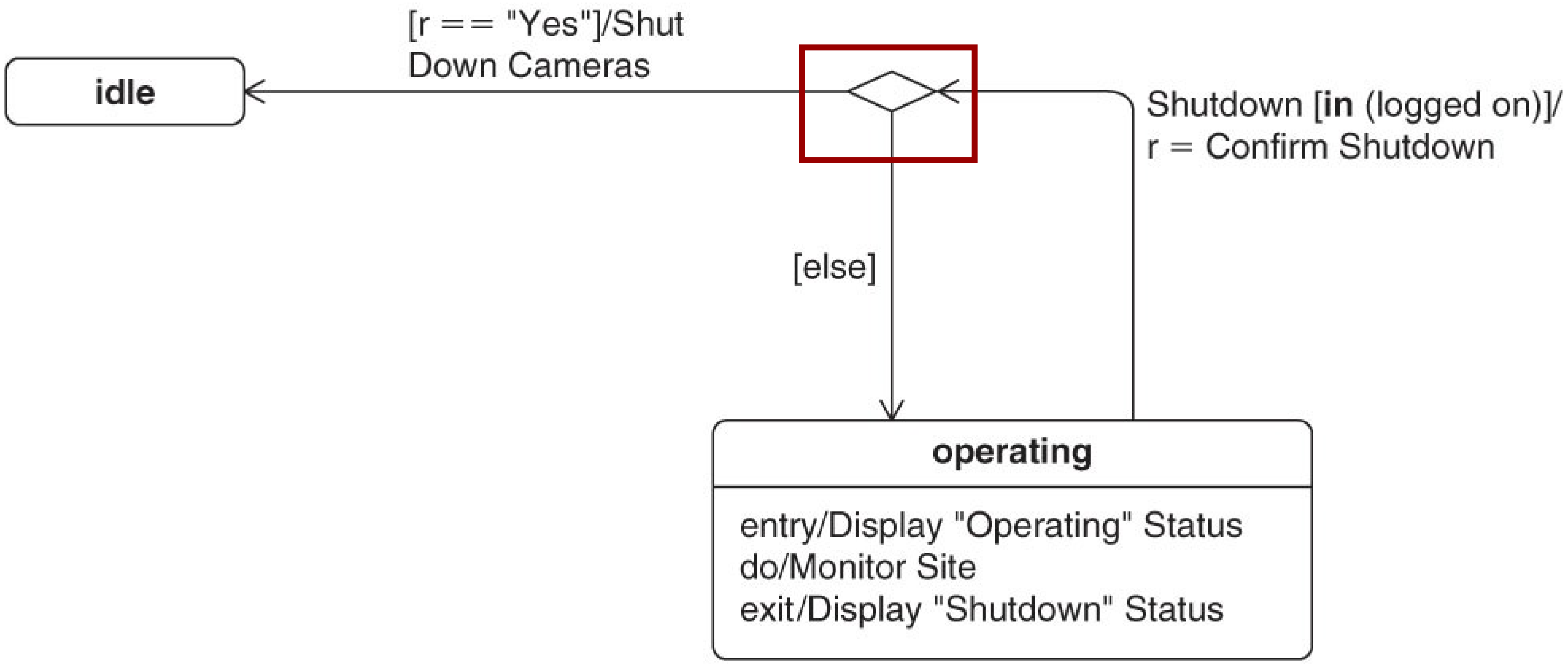


logged on
entry/Display "Logged On"; time on = now
exit/Display "Session Length:", now – time on
Logout [logged in >1]/logged in = logged in – 1
Login/logged in = logged in + 1
logged off
entry/Display "Logged Off"
Login/logged in = logged in + 1
Logout [logged in == 1]/logged in = logged in – 1
route maintenance
Edit Routes
idle
maintaining
Store Routes
alert management
/alert count = 0
Alert/alert count = alert count + 1
normal
alerted
Stand Down
/Display "Alerts: ", alert count

© 2008 Elsevier, Inc.: A Practical Guide to SysML

Choice

- Allows one or more input paths, and one or more output paths
- Transition flow is determined by Guards on output paths


[r == "Yes"]/Shut Down Cameras
idle
Shutdown [in (logged on)]/ r = Confirm Shutdown
[else]
operating
entry/Display "Operating" Status
do/Monitor Site
exit/Display "Shutdown" Status

© 2008 Elsevier, Inc.: A Practical Guide to SysML

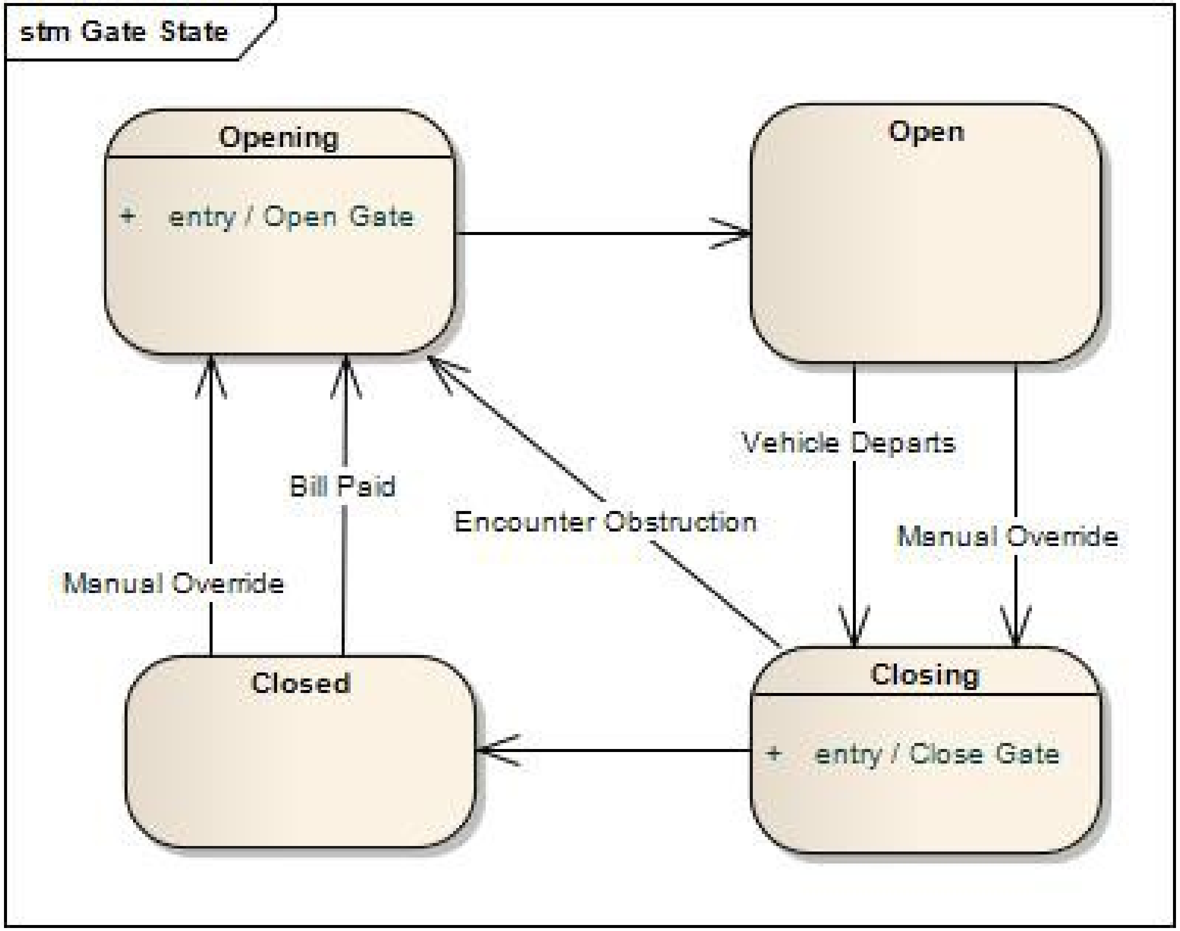
How State Machines are Modeled

- Identify which entities (blocks) need to be modeled
  - Identify entities with states
- Determine the Triggering Events related to the entity
  - What causes it to change state?
- List States that the Triggering Events cause
- Determine the Behaviors related to an entity
  - What happens when entering a state?
  - What happens while in a state?
  - What happens when exiting a state?
- Specify Composite States (if necessary)
- Create the State Machine diagram

State Machine Modeling for In-Class Project

- Build State Machine Model for Parking Garage Gate using EA
- Define
  - Triggers
  - States
  - Behaviors and Effects

State Machine for Parking Garage Gate



Summary

- State Machines Diagrams are used to depict how a Block changes State
- State Machines can be comprised of:
  - States
  - Transitions
  - Composite States
  - Pseudo-states
- States represent a condition in the life of a Block
- Behaviors are the actions associated with a State
- Transitions are used to show how a Block changes from one State to another
- Transitions can consist of Triggers, Guards, and Effects
- Composite States are used to depict the hierarchy of States
- Pseudo-states are used to determine the next active State in a State Machine diagram

Homework

- Read Chapter 10 (State Machines) in text
- Develop a State Machine Diagram for each of the following:
  - Alarm Clock Radio
  - Coke Machine
- Depict the associated:
  - States
  - Triggers
  - Behaviors
  - Effects
- Depict Regions and Sub-states (if applicable)
- Read Overview Section on Interactions (Section 9.1) in text