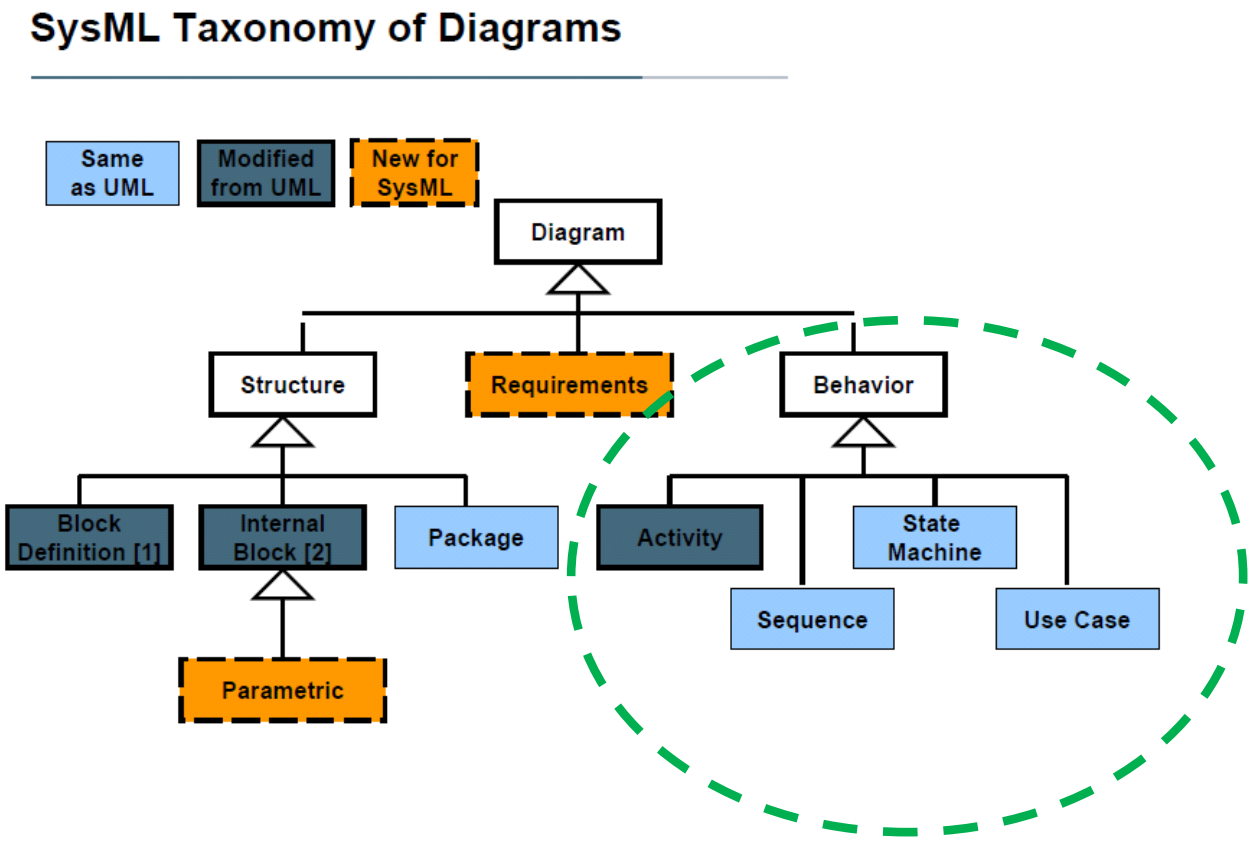
SysEng 6542
Model Based Systems Engineering

SysML - Modeling Behavior

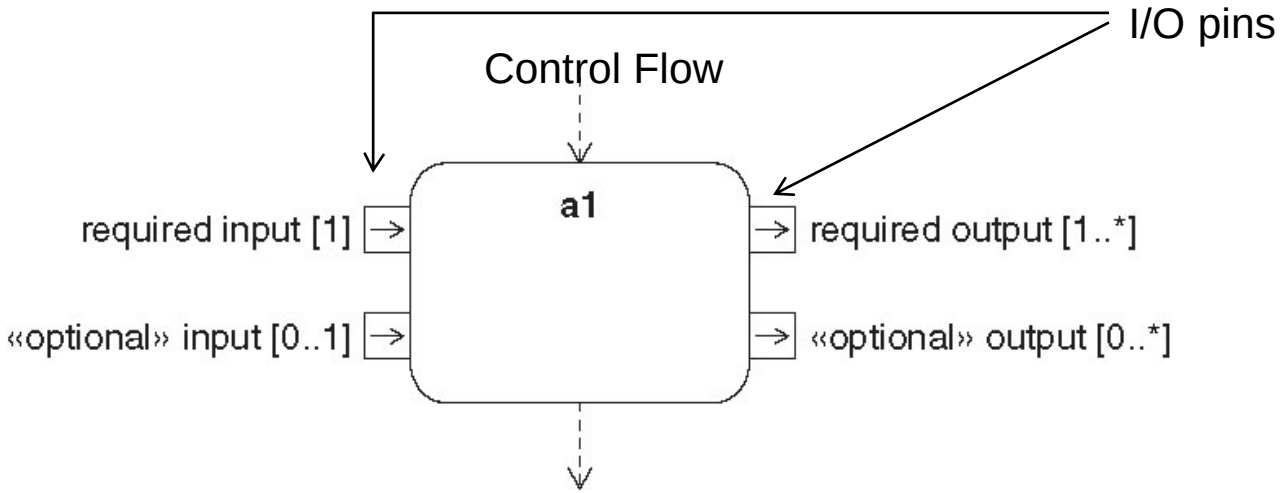
Dr Quoc Do

SysML Behavior Diagrams

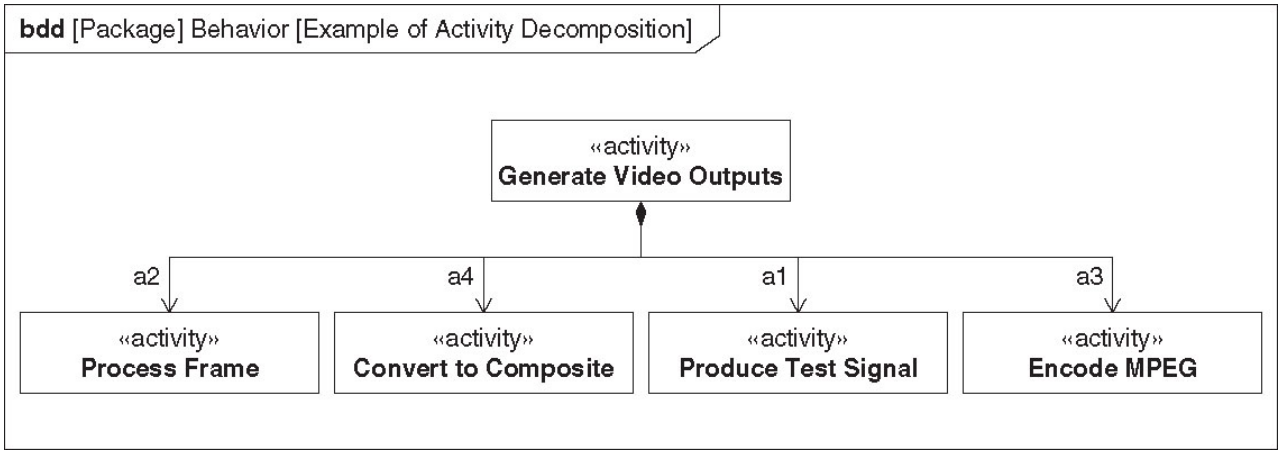


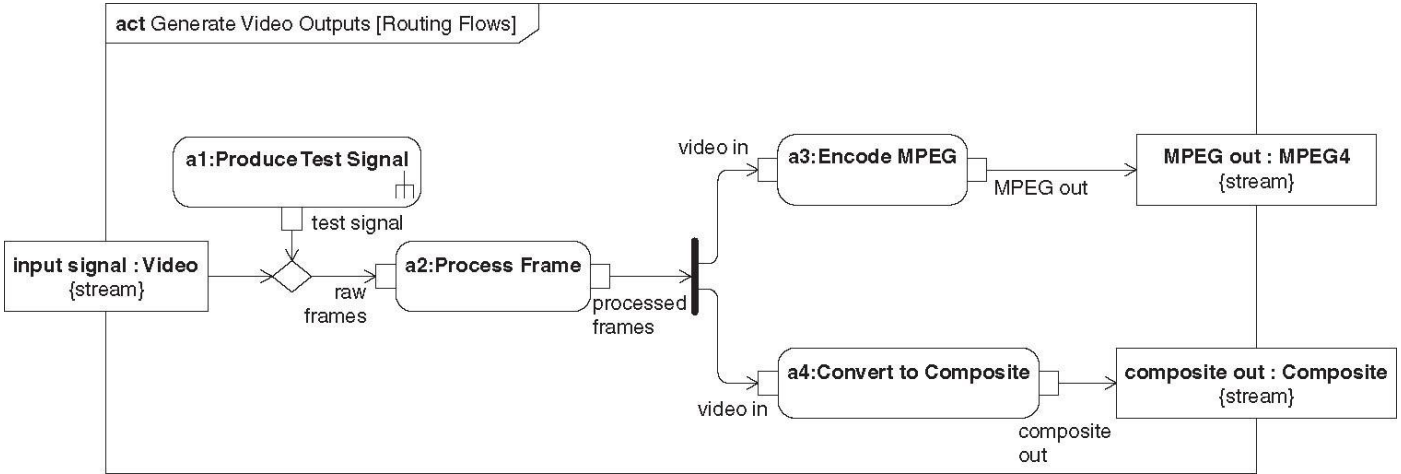
Activity Diagram

- Modified UML Behavior Diagram
- Defines the actions and flow from input to output for an aspect of the system
- Activity must be executing
- Required tokens must be available
  - Input, output, pins, and control
- Action places token on output ends and output control flows
- Action terminates if activity terminates



Activity Diagram





Activity Diagram – Routing Object Flows

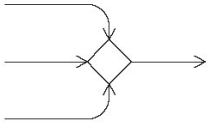
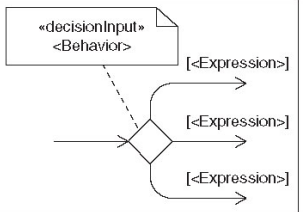
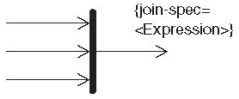
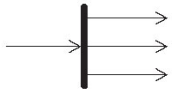
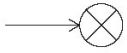
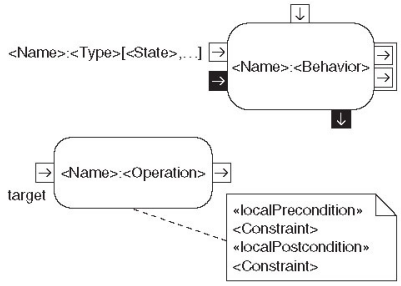
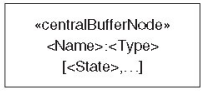
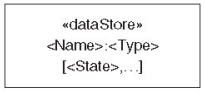
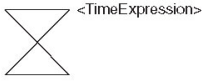
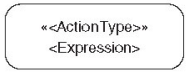
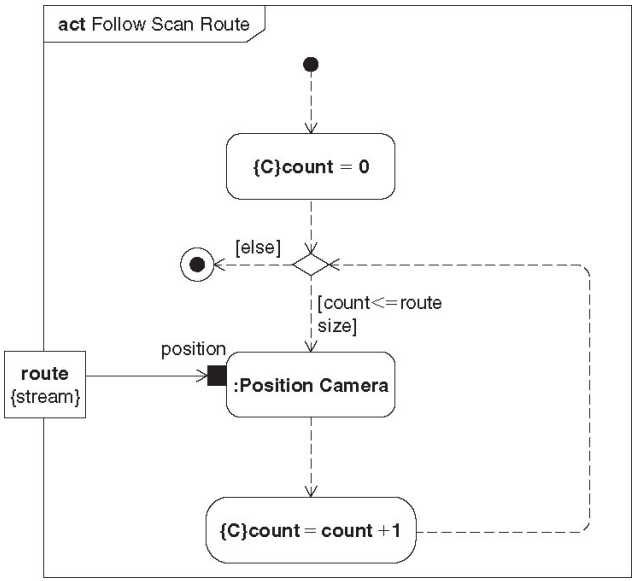
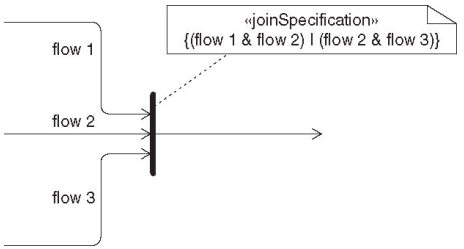
Diagram Element	Notation	Description	Section
Merge Node		A merge node has one output flow and multiple input flows—it routes each input token received on any input flow to its output flow. Unlike a join node, a merge node does not require tokens on all its input flows before offering them on its output flow. Rather it offers tokens on its output flow as soon as it receives them.	8.5.1, 8.6.1
Decision Node		A decision node has one input flow and multiple output flows—an input token can only traverse one output flow. The output flow is typically established by placing mutually exclusive guards on all outgoing flows and offering the token to the flow whose guard expression is satisfied. A decision node can have an accompanying decision input behavior, which is used to evaluate each incoming object token and whose result can be used in guard expressions.	8.5.1, 8.6.1
Join Node		A join node has one output flow and multiple input flows—it has the important characteristic of synchronizing the flow of tokens from many sources. Its default behavior can be overridden by providing a join specification, which can specify additional control logic.	8.5.1, 8.6.1
Fork Node		A fork node has one input flow and multiple output flows—it replicates every input token it receives onto each of its output flows. The tokens on each output flow may be handled independently and concurrently.	8.5.1, 8.6.1
Initial Node		When an activity starts executing a control token is placed on each initial node in the activity. The token can then trigger the execution of an action via an outgoing control flow.	8.6.1
Activity Final Node		When a control or object token reaches an activity final node during the execution of an activity, the execution terminates.	8.6.1
Flow Final Node		Control or object tokens received at a flow final node are consumed but have no effect on the execution of the enclosing activity. Typically they are used to terminate a particular sequence of actions without terminating an activity.	8.6.1

Diagram Element	Notation	Description	Section
Call Action Node		Call actions can invoke other behaviors either directly or through an operation, and are referred to as call behavior actions and call operation actions, respectively. A call action must own a set of pins that match in number and type of the parameters of the invoked behavior/operation. A called operation requires a target. Streaming pins may be marked as {stream} or filled (as shown). Where the parameters of the called entity are grouped into sets, the corresponding pins are as well. Pre- and postconditions can be specified that constrain the action such that it cannot begin to execute unless the precondition is satisfied, and must satisfy the postcondition to successfully complete execution.	8.1, 8.3, 8.4.2
Central Buffer Node		A central buffer node provides a store for object tokens outside of pins and parameter nodes. Tokens flow into a central buffer node and are stored there until they flow out again.	8.5.3
Datastore Node		A datastore node provides a copy of a stored token rather than the original. When an input token represents an object that is already in the store, it overwrites the previous token.	8.5.3
Control Operator Action Node		A control operator produces control values on an output parameter, and is able to accept a control value on an input parameter (treated as an object token). It is used to specify logic for enabling and disabling other actions.	8.6.2
Accept Event Action Node		An activity can accept events using an accept event action. The action has (sometimes hidden) output pins for received data.	8.7
Accept Time Event Node		A time event corresponds to an expiration of an (implicit) timer. In this case the action has a single (typically hidden) output pin that outputs a token containing the time of the accepted event occurrence.	8.7
Send Signal Action		An activity can send signals using a send signal action. It typically has pins corresponding to the signal data to be sent and the target for the signal.	8.7
Primitive Action Node		Primitive actions include: object access/ update/manipulation actions, which involve properties and variables, and value actions, which allow the specification of values. The <Expression> will depend on the nature of the action.	8.12.1

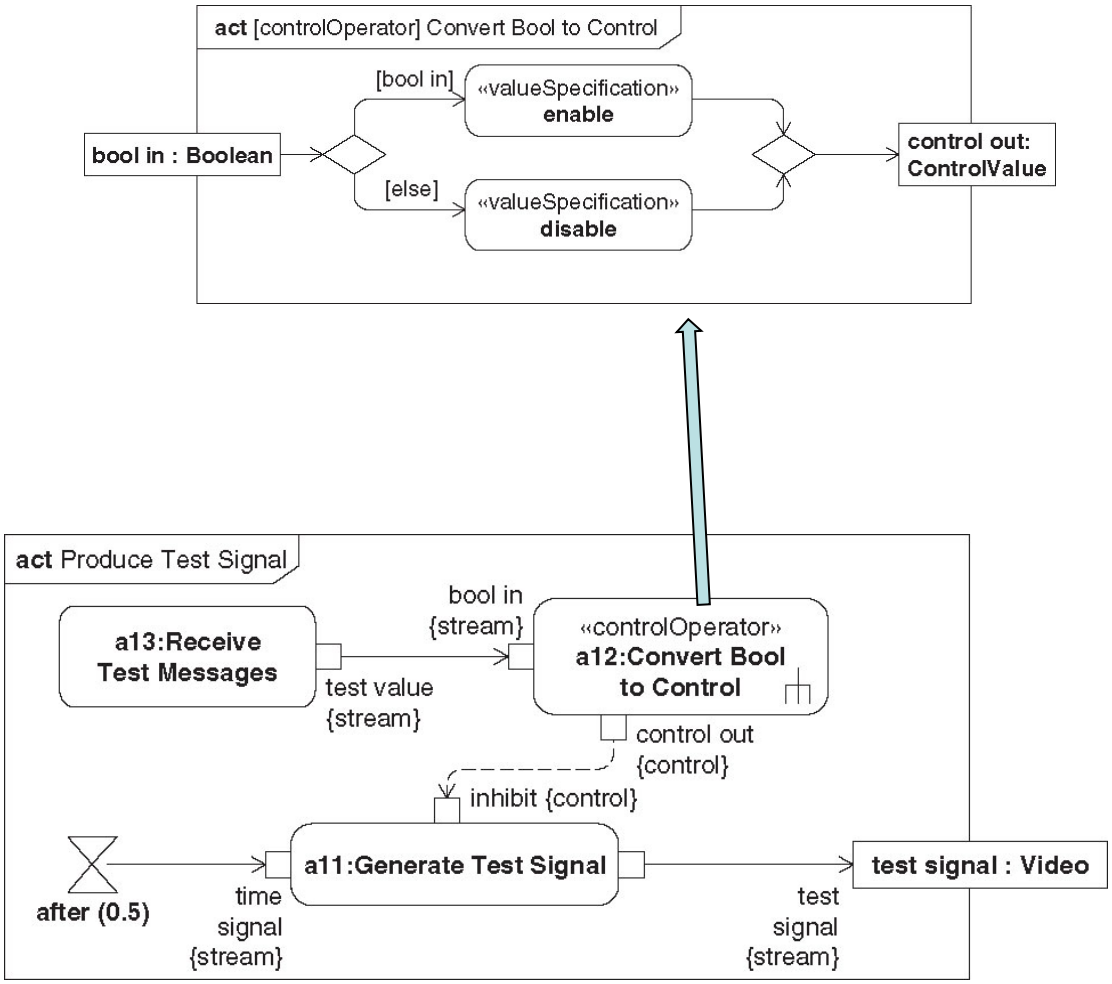
Control Logic with Nodes

- In addition to object flow nodes, control nodes can be:
  - Initial node (black dot)
  - Activity final node (bullseye)
  - Flow final node (45 degree crosshair)



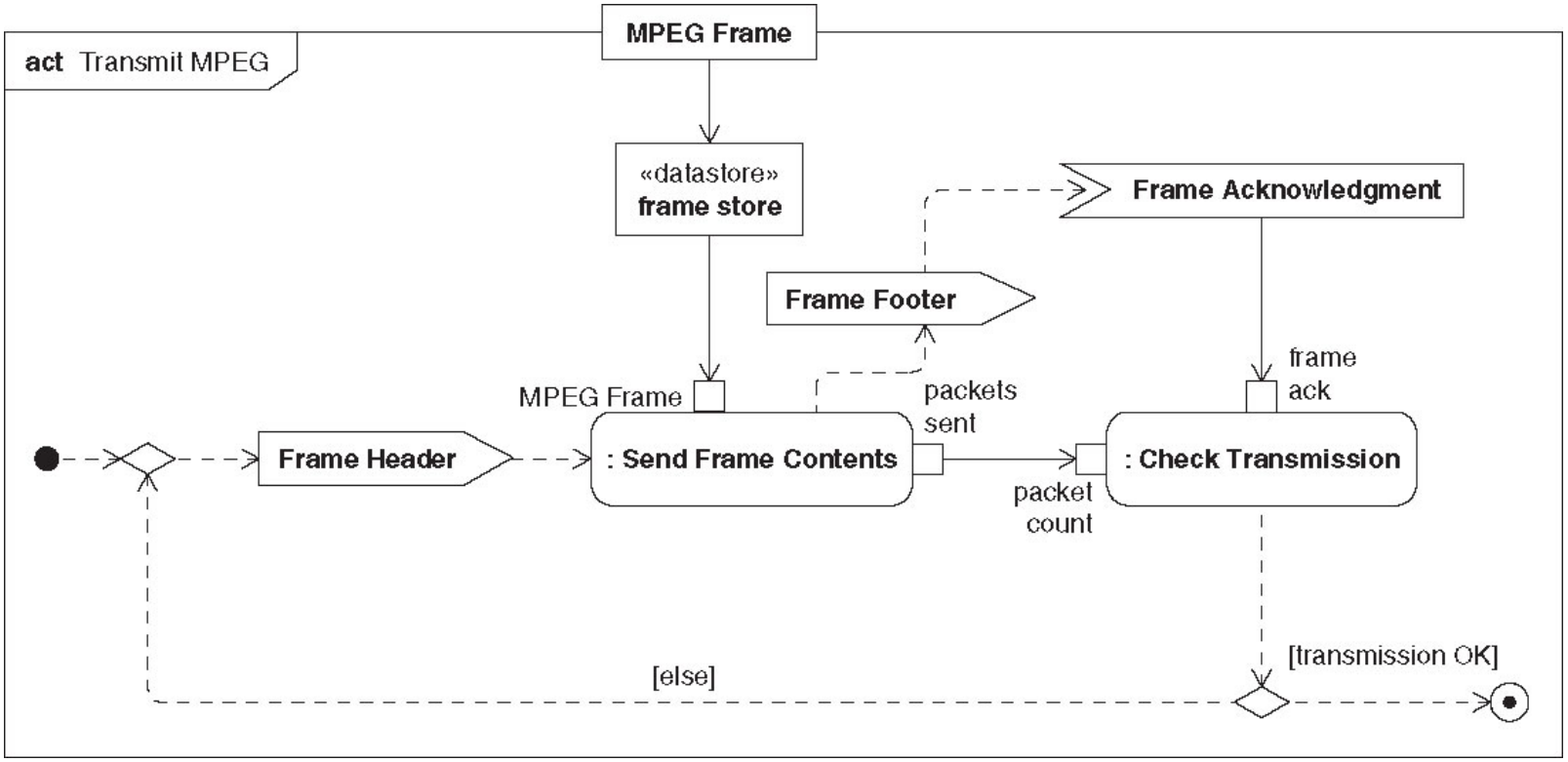


Activity Diagram



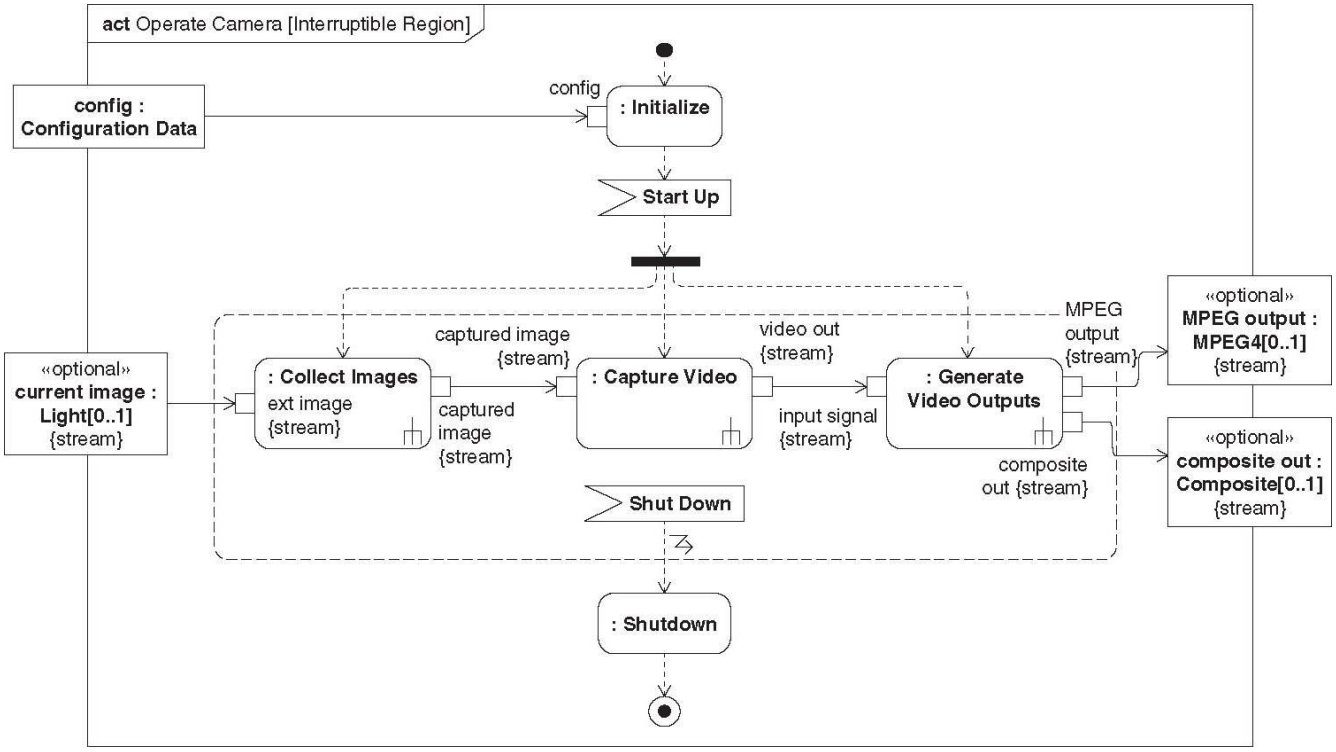
Activity Diagram - Examples

- Let's see shows how MPEG frames get transmitted over the surveillance camera network.



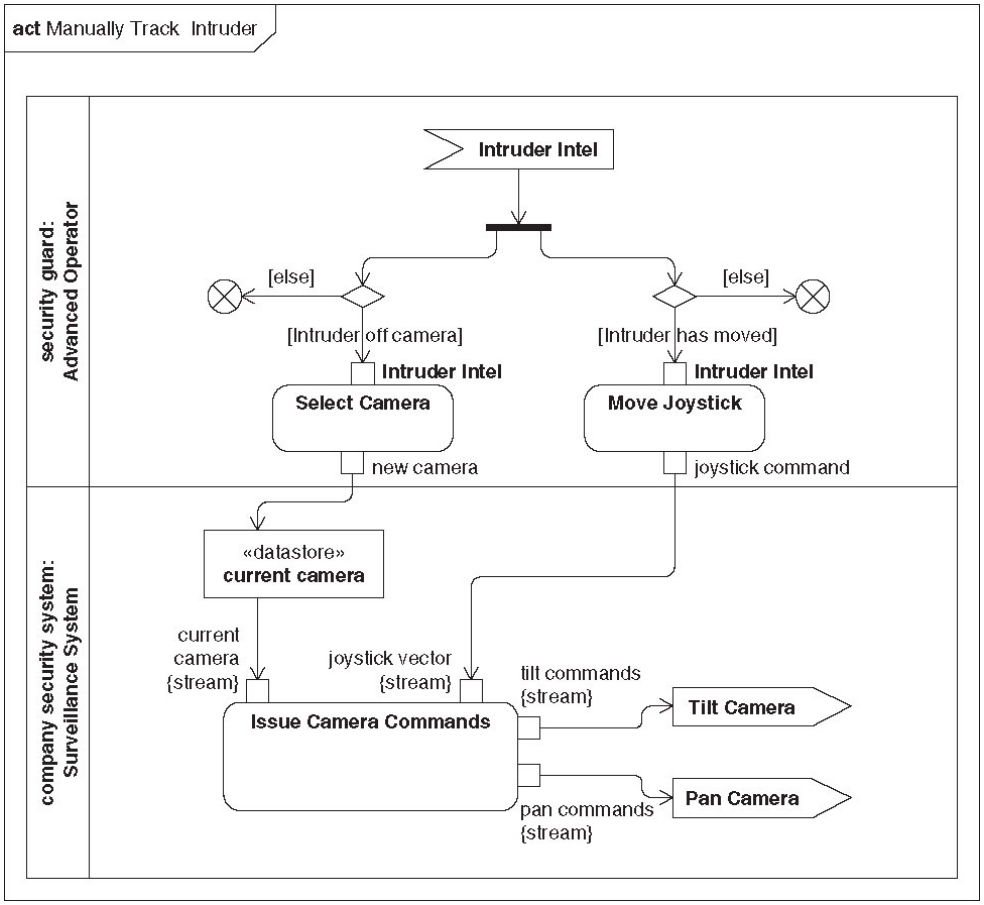
Activity Diagram Examples

- Let's see a more complete definition of the Camera behaviour



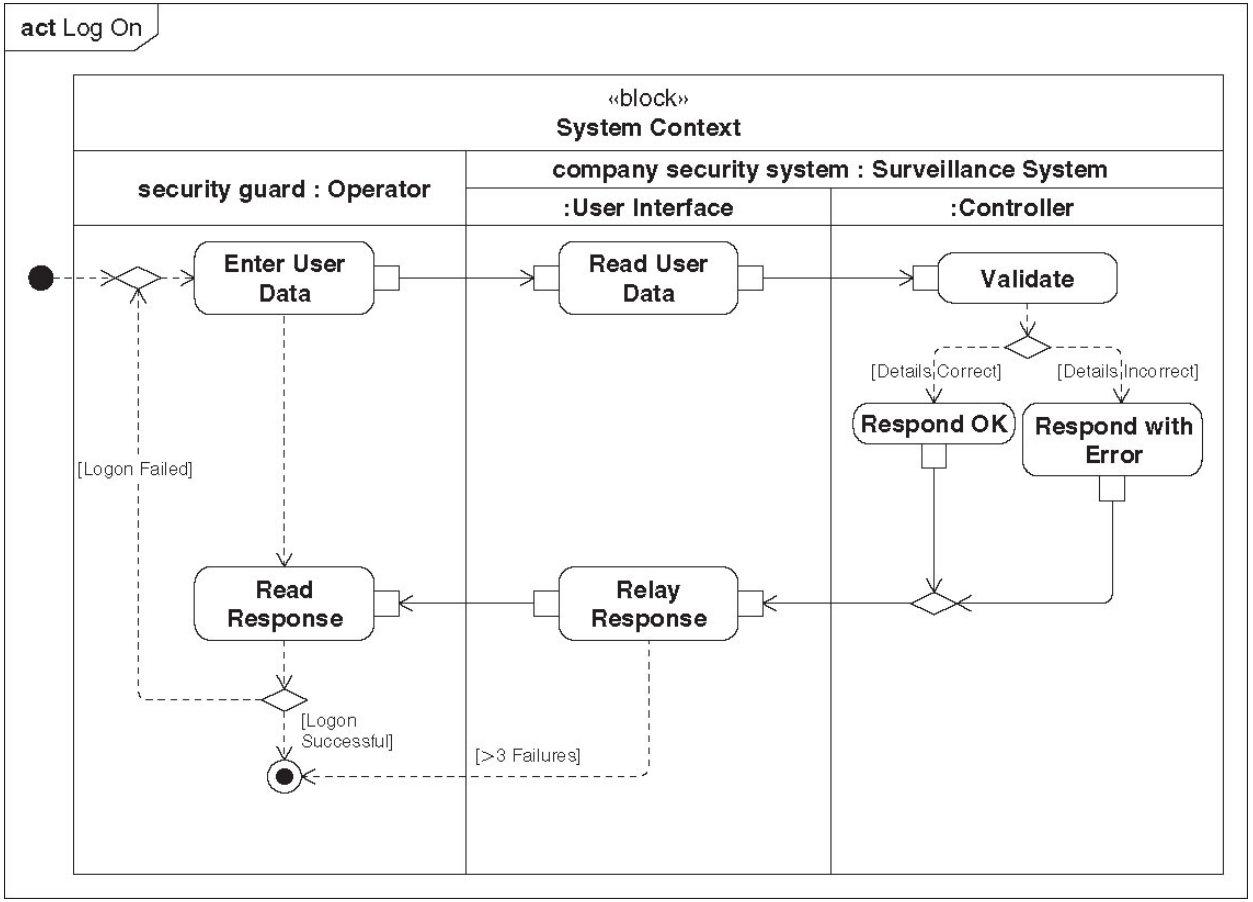
Relate Activities to Blocks

Activity Partitioning know as Swimlane

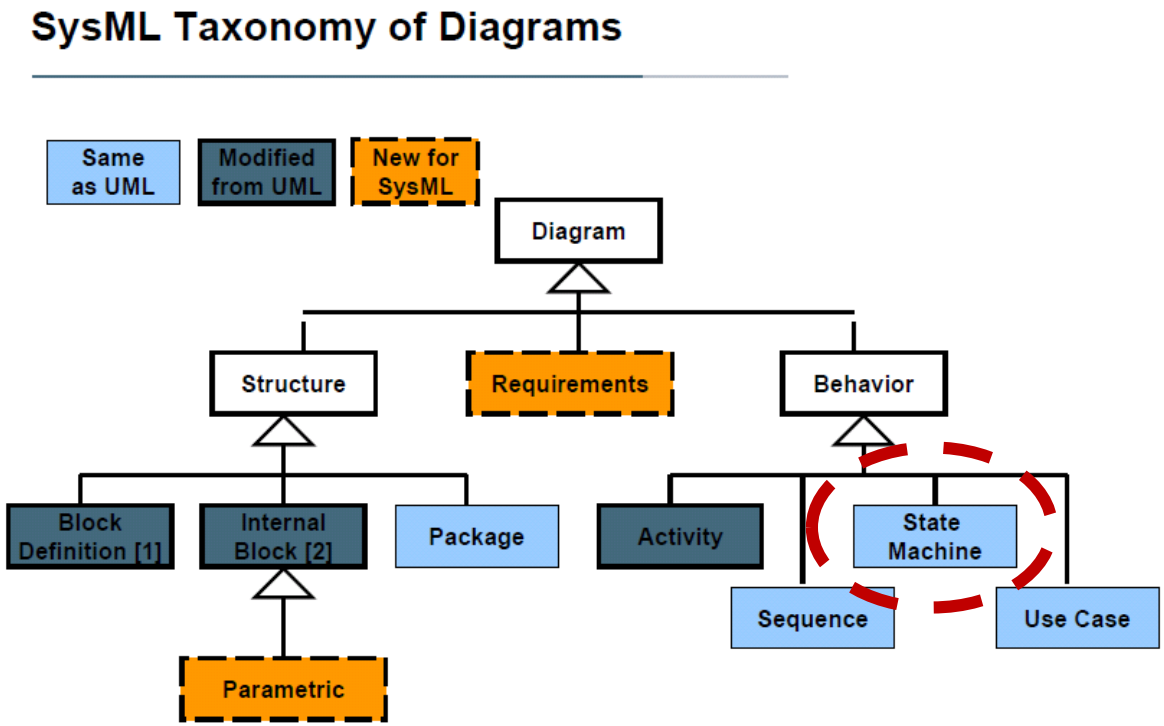


Relate Activities to Blocks

Nested Activity Partitioning



SysML – State Machine



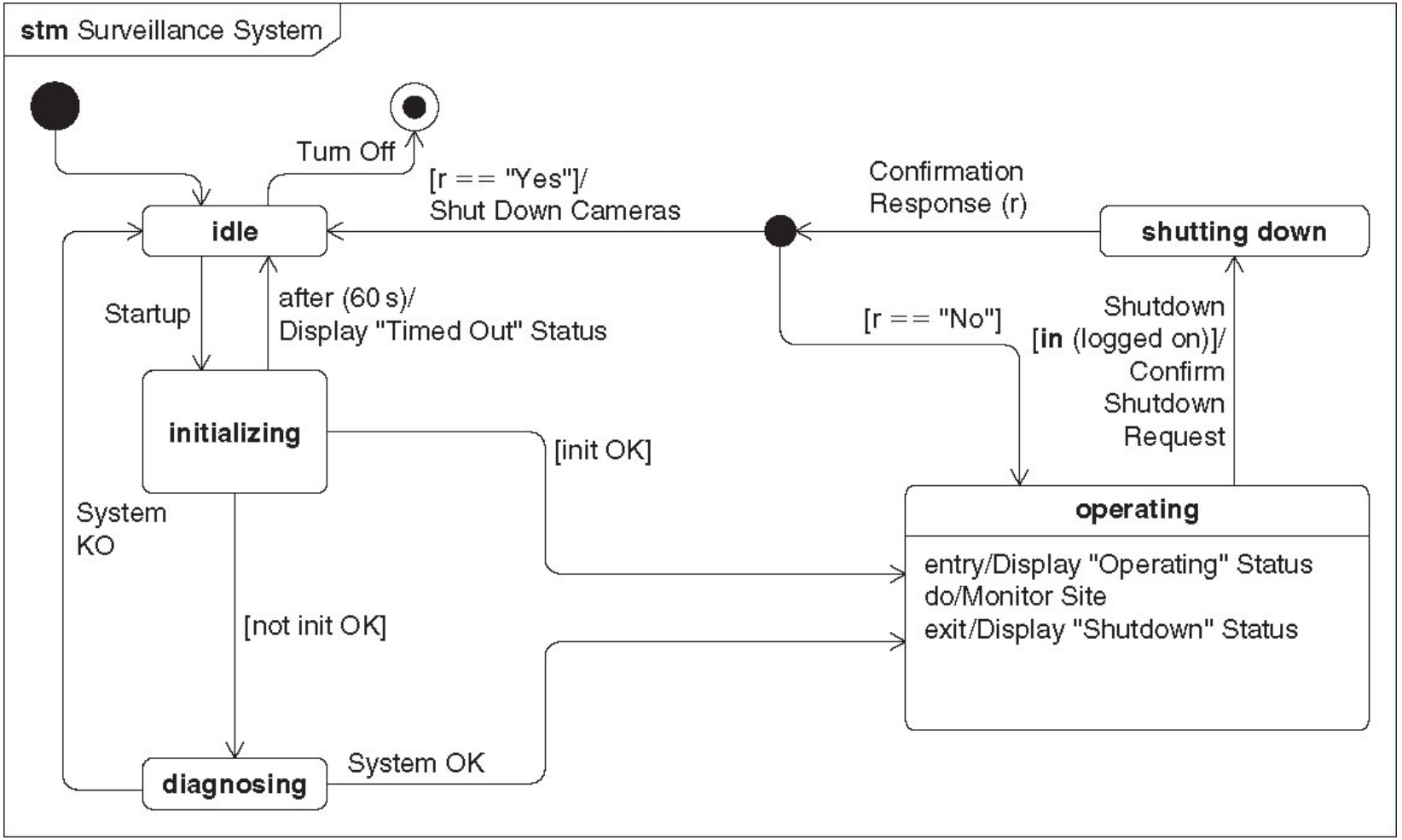
State Machine Diagram

- UML Behavior Diagram
- Makes it possible to include event driven behavior into system models
- State Machine – a device that stores the status of something at a given time and can process input to change this status and possibly take actions
- An abstract model of a machine with primitive memory

State Machine

- State – Current condition of the model
- Transition – possible movements between states
- Actions – state dependent occurrences

State Machine Diagram

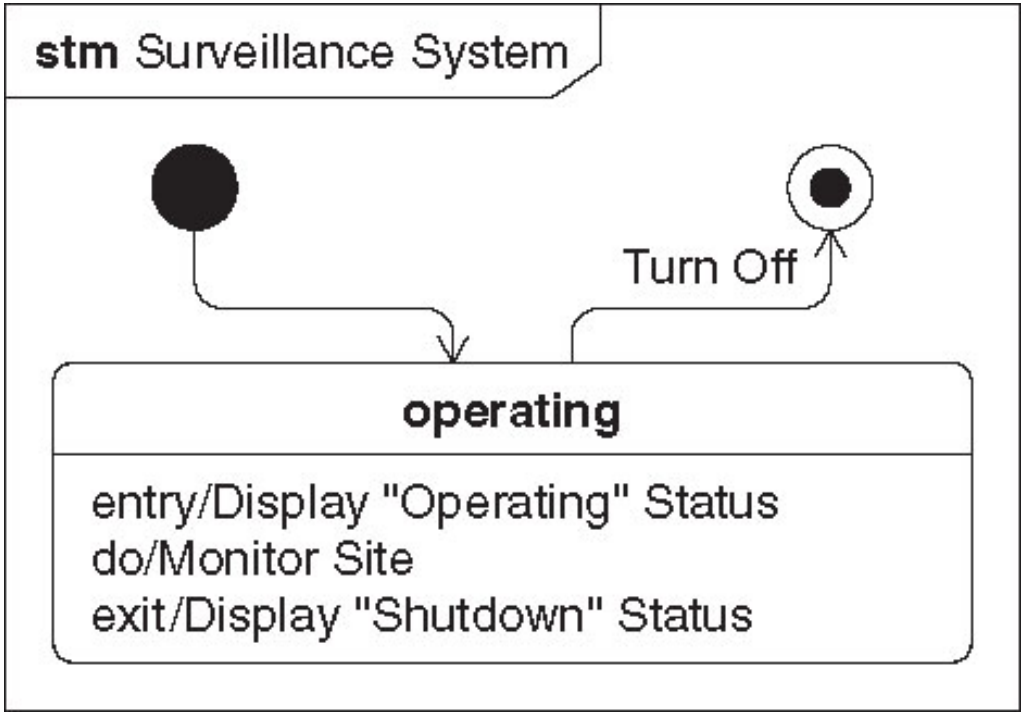


State Machine Diagram

- Can be referenced by interactions or called from an activity
- Normally owned by blocks
- States are viewed as regions, with only one region being active at a time

State Machine Diagram

- States can have entry and exit actions, as well as ‘do’ activities
- State machine diagram separated into regions, defined by states (pseudostates)
  - Initial pseudostate
  - Final state
  - Terminate pseudostate



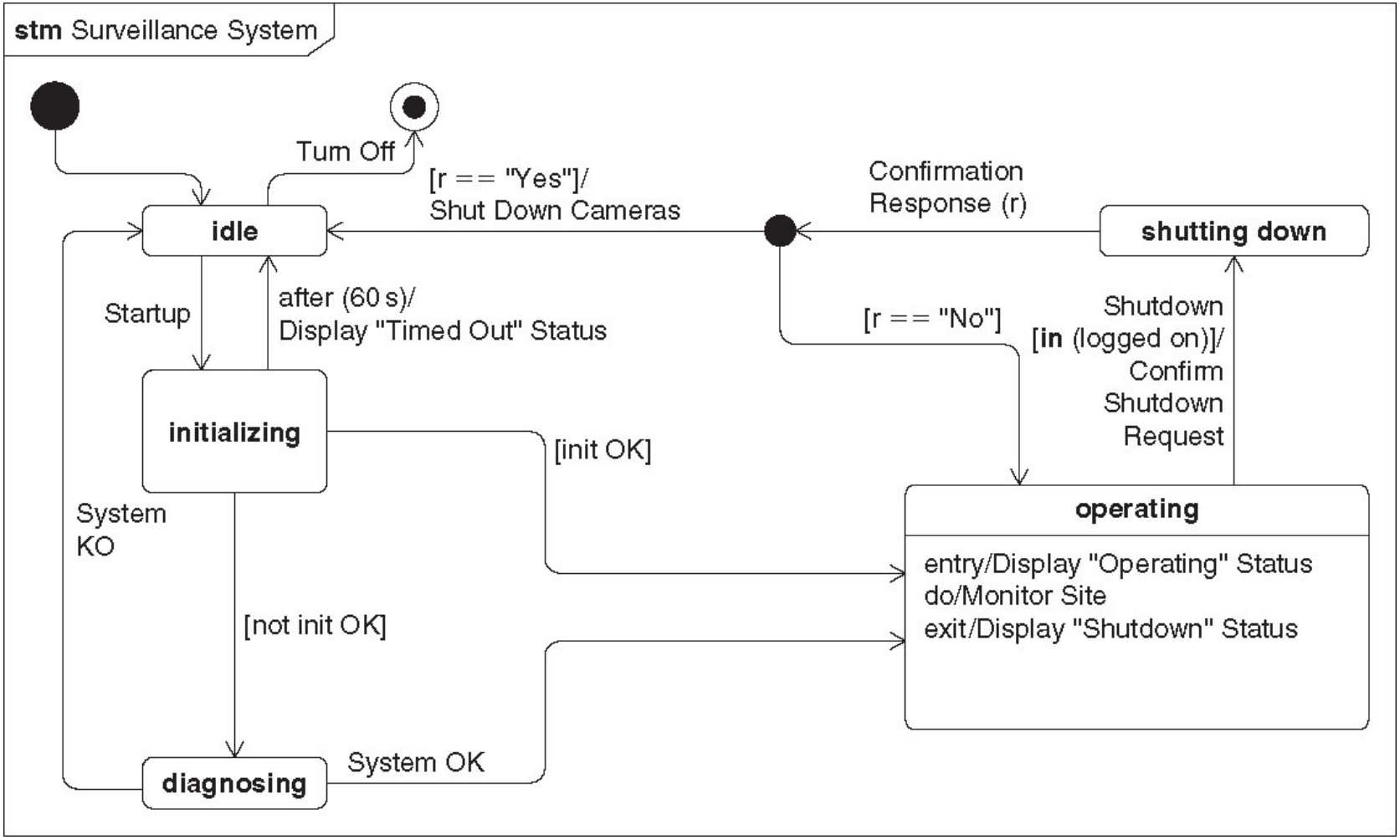
Transitions

- Transitions are defined by:
  - Triggers: an event causing a transition
  - Guard: evaluated to test if transition is valid
  - Effect: behavior executed when transition is executed
- Format
  - Trigger[Guard]/Effect
  - Can also be named

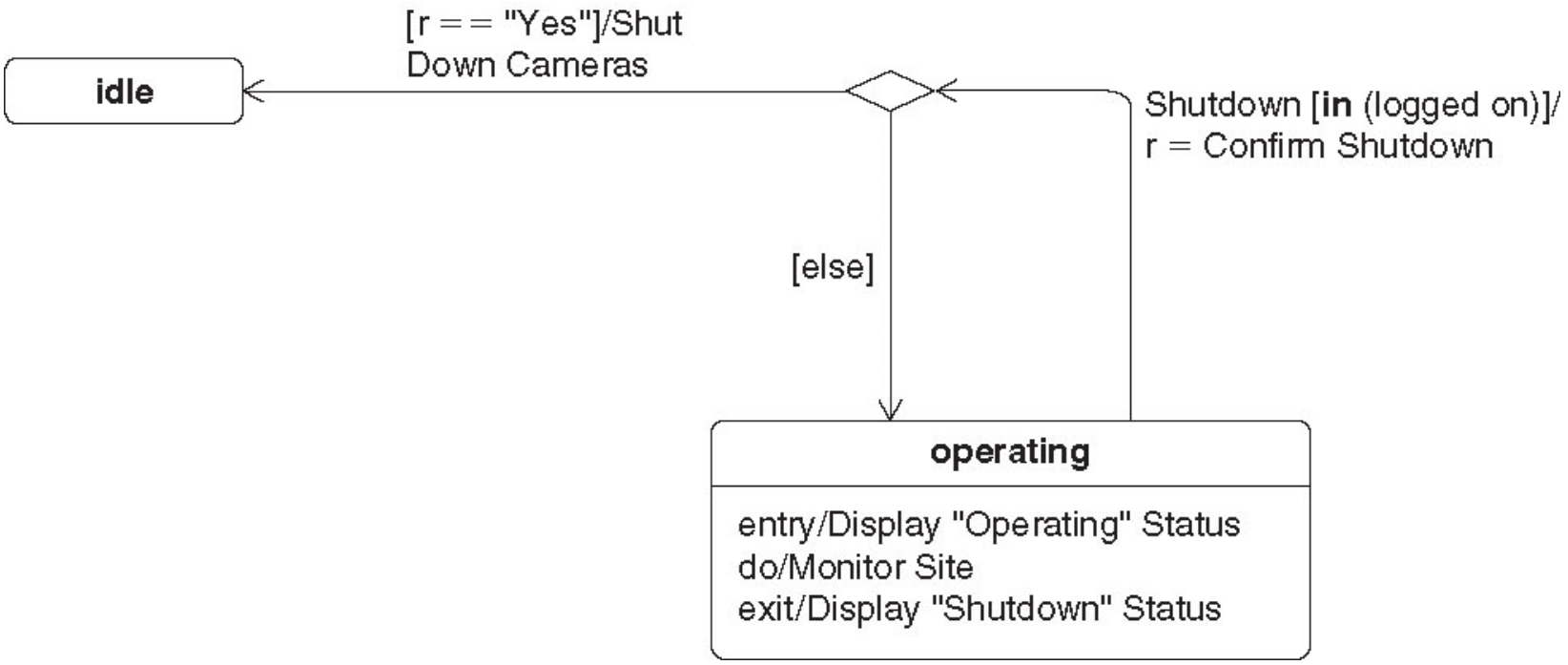
Transition Triggers

- Identify Stimuli that cause transition
  - Signal event: a new asynchronous message received
  - Time event: specified by a time interval
  - Change event: some condition has been satisfied
  - Call event: operation on owning block requested
  - Internal completion event

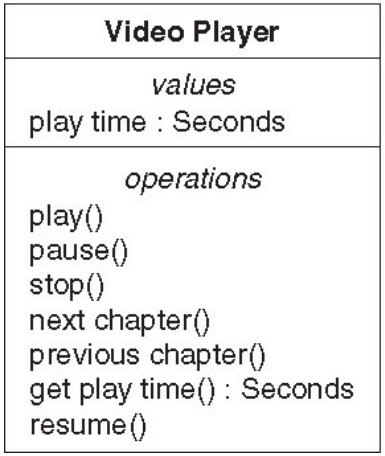
State Machine and Triggers

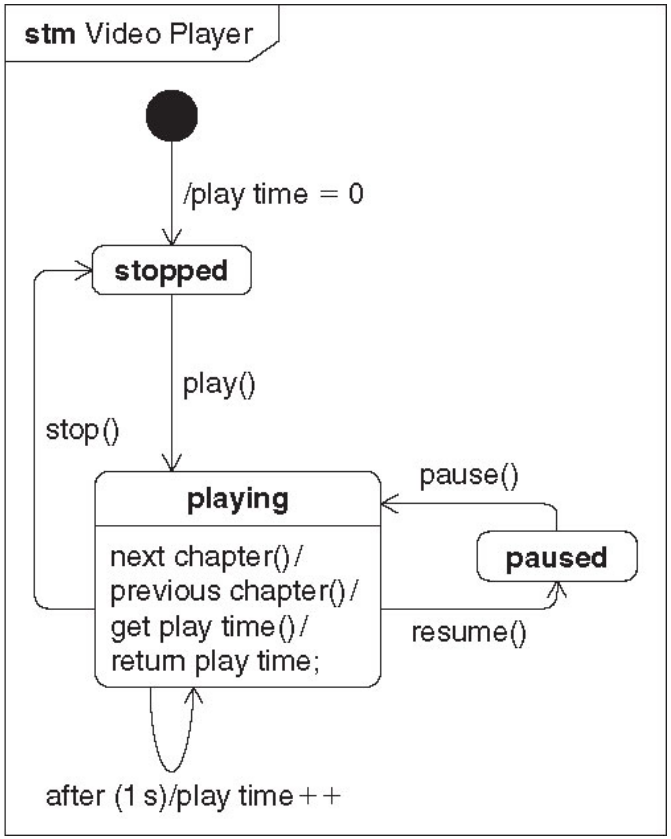


Choice Pseudosate

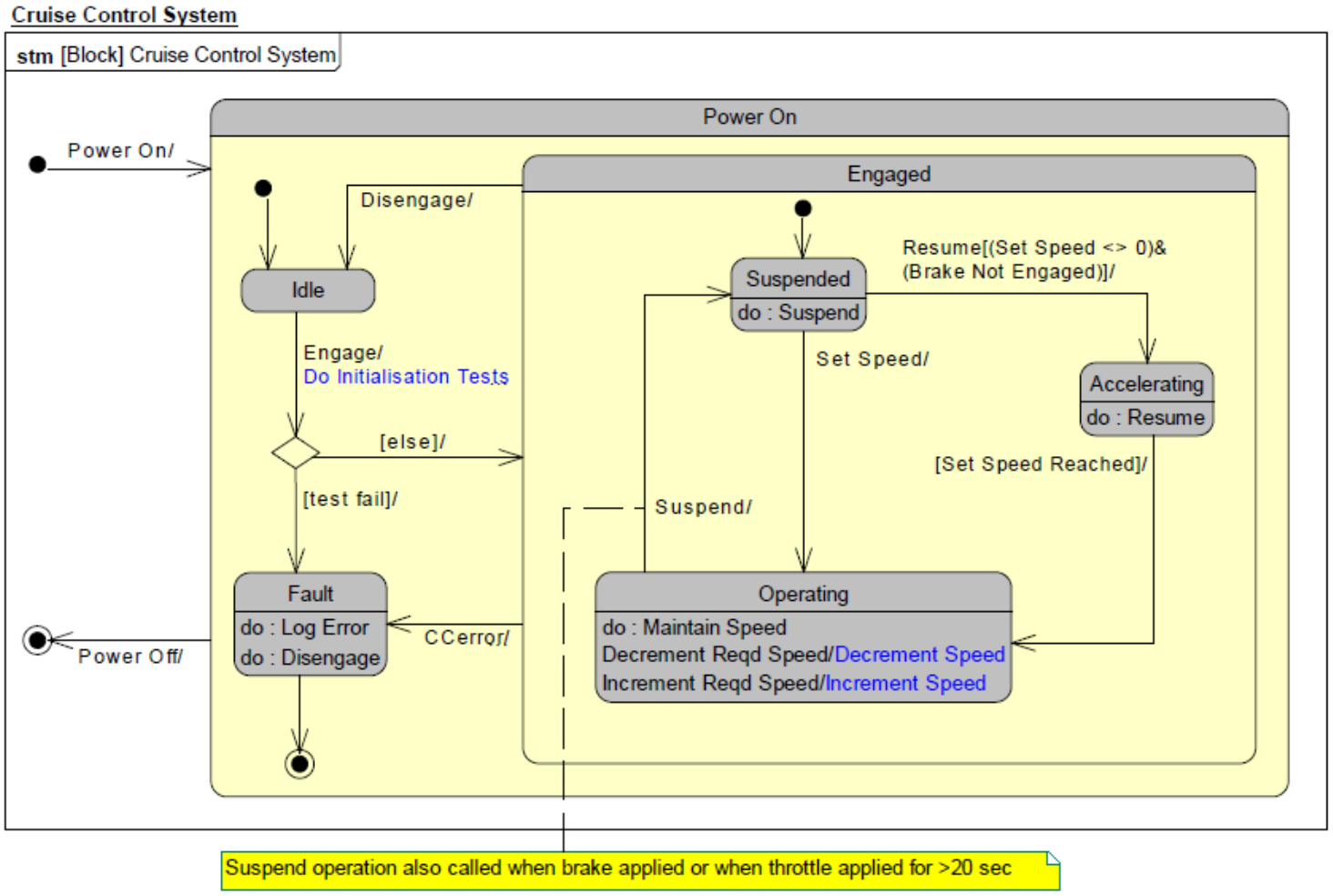


Example: Video Player

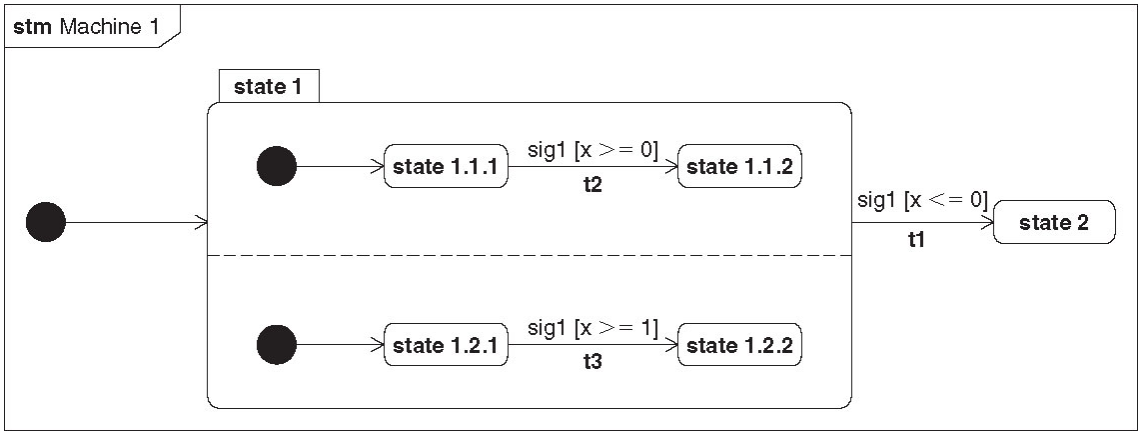


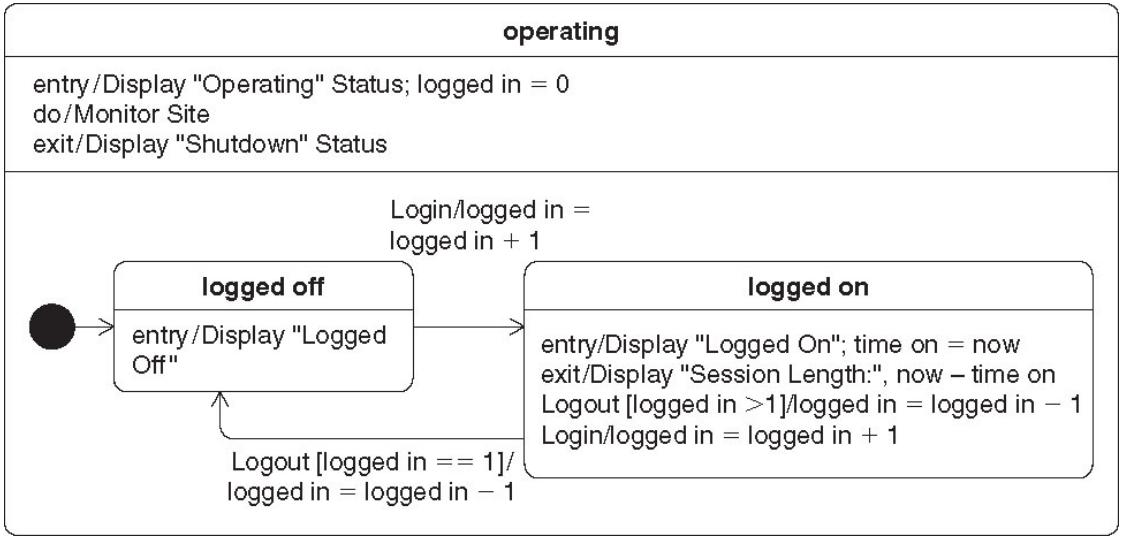


Example – Cruise Control System

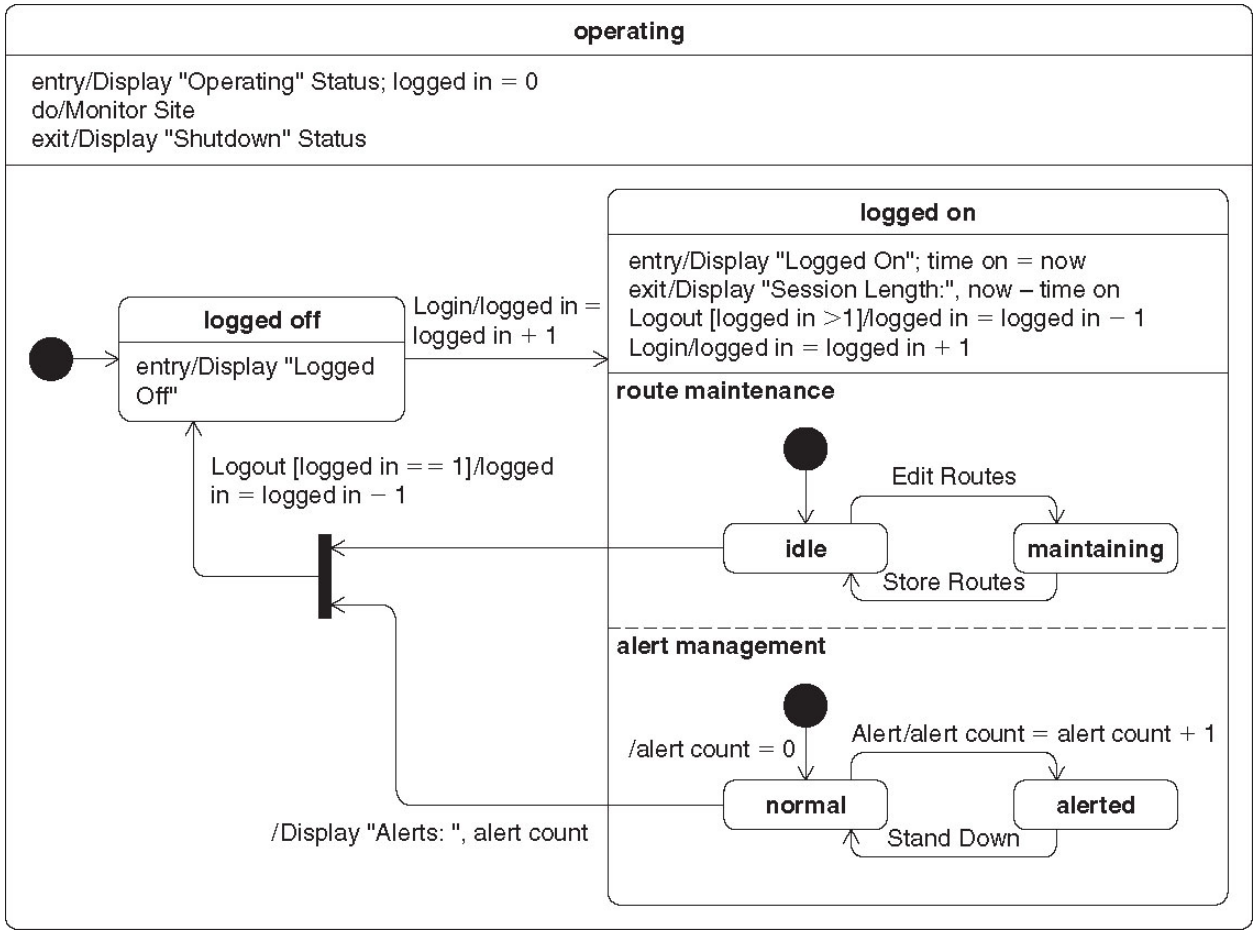


State Hierarchies

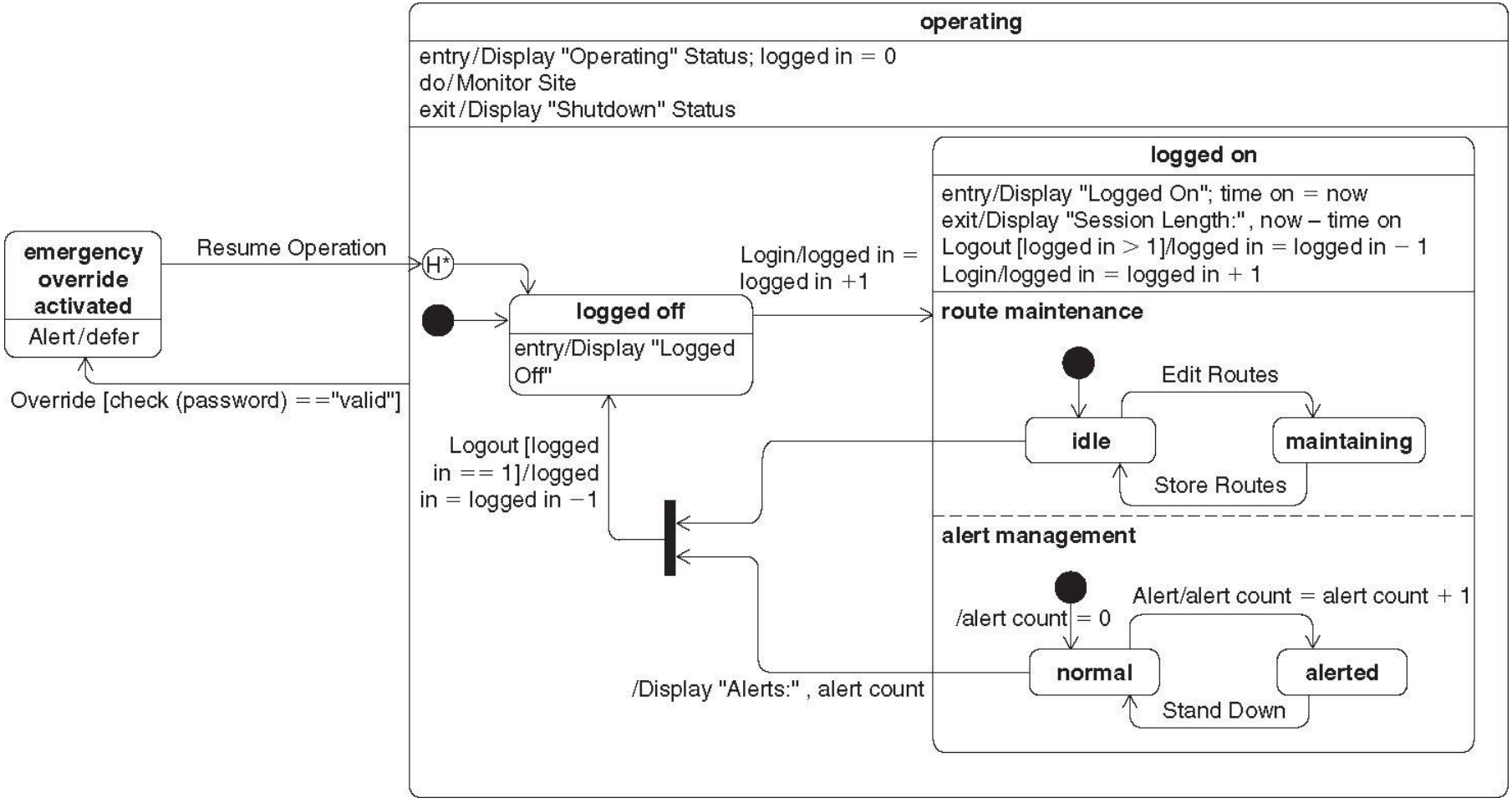




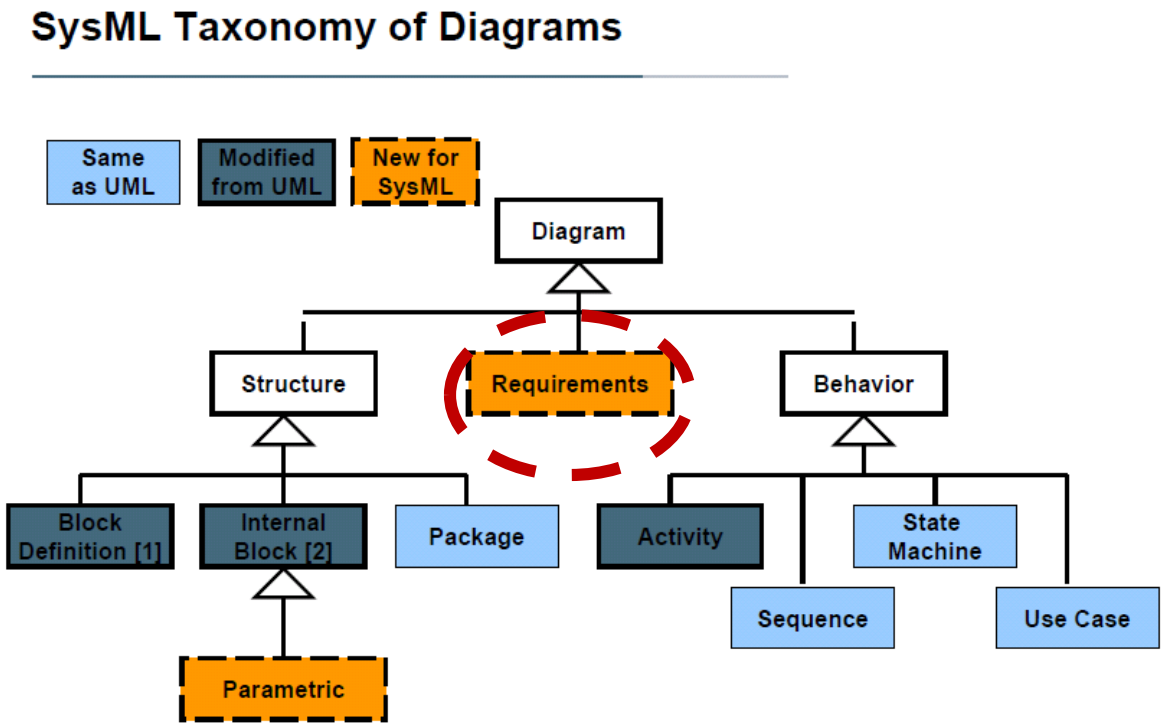
Composite State w/ Multiple Regions



History Pseudo-State



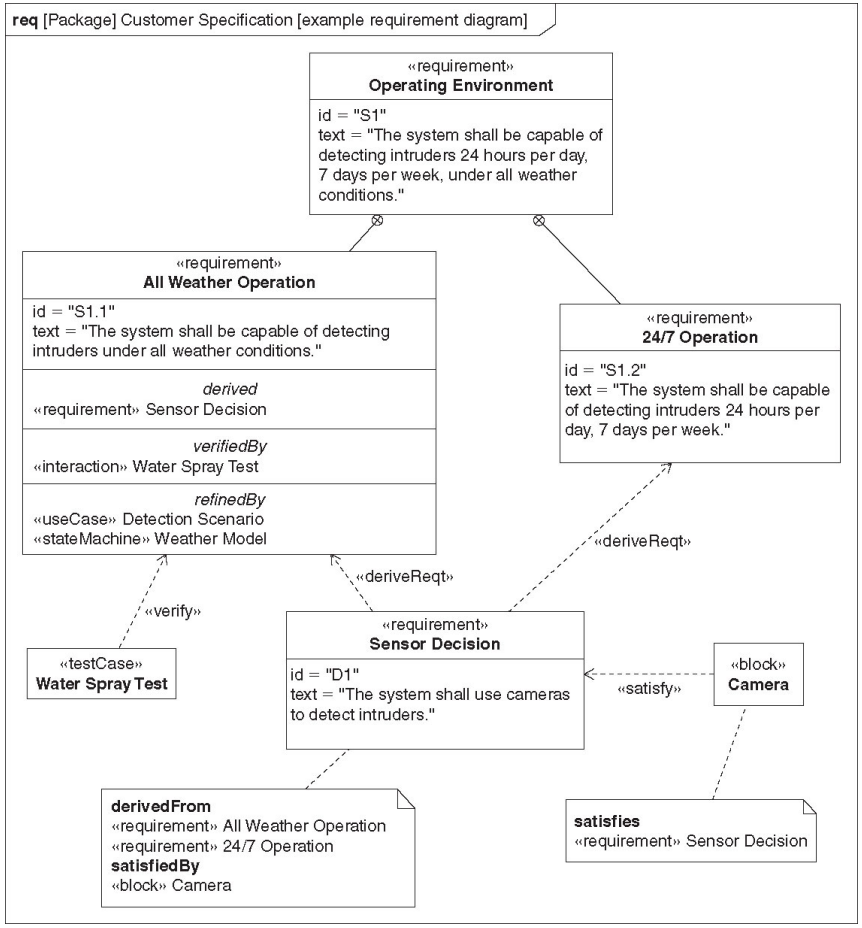
SysML – Requirements Diagram



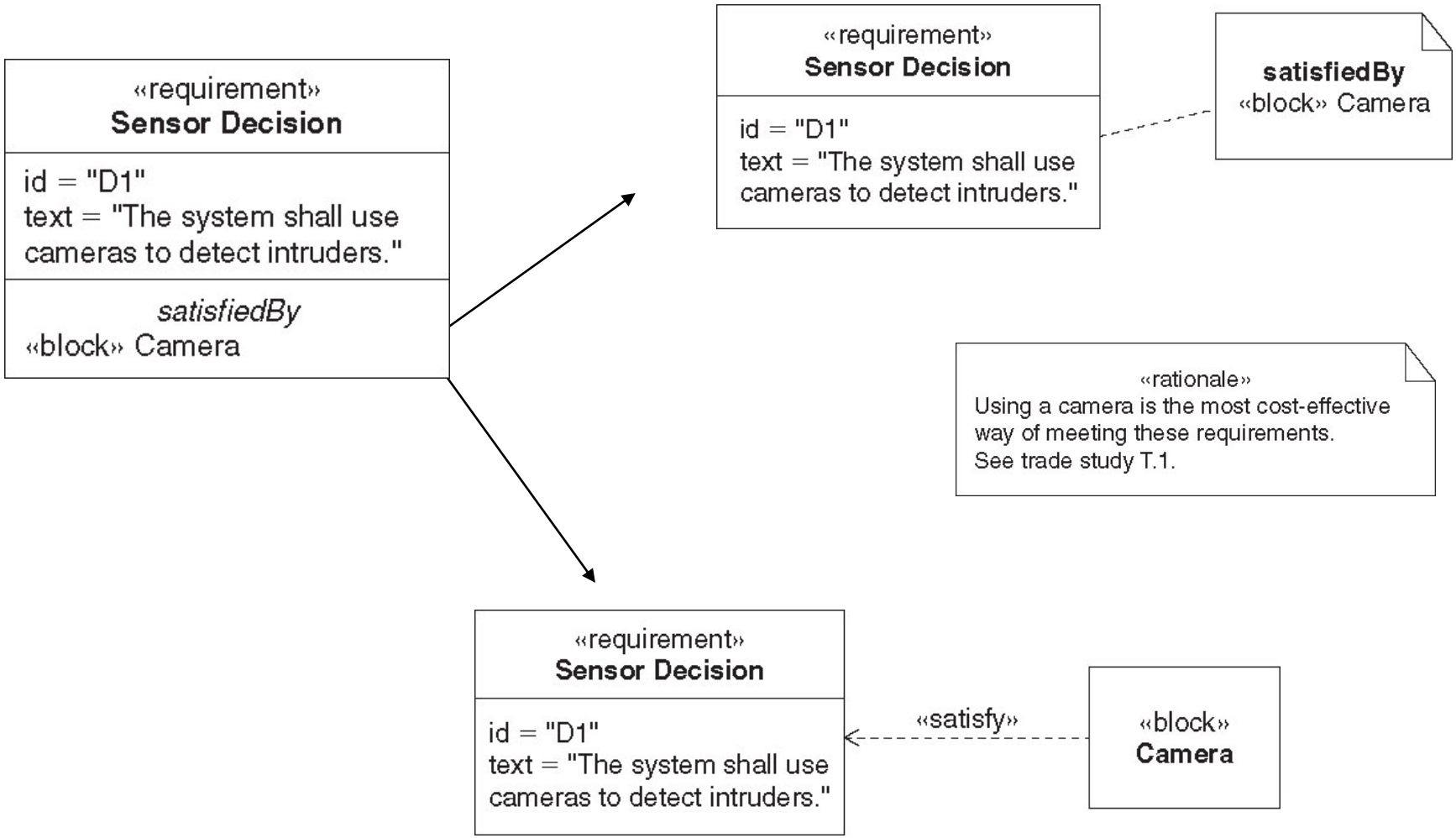
Requirements Diagram

- New Special Use Diagram
- Contains textual requirement information

Requirements Diagram



Requirements Diagram



Requirements Diagram

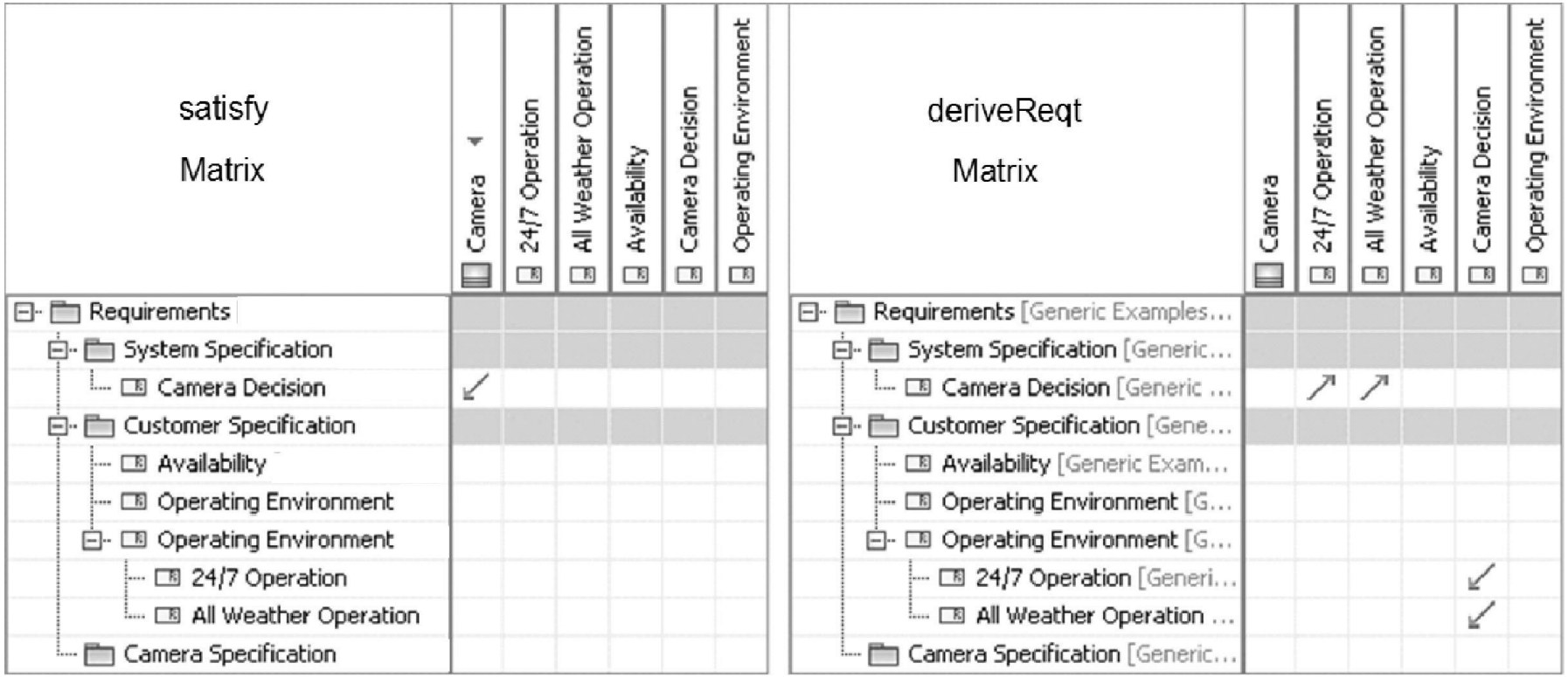
table [Package] System Specification [Decomposition of top-level requirements]

id	name	text
S1	Operating Environment	The system shall be capable of detecting intruders 24 hours per day…
S1.1	Weather Operation	The system shall be capable of detecting intruders under all weather…
S1.2	24/7 Operation	The system shall detect intruders 24 hours per day, 7 days per week
S2	Availability	The system shall exhibit an operational availability (Ao) of 0.999…

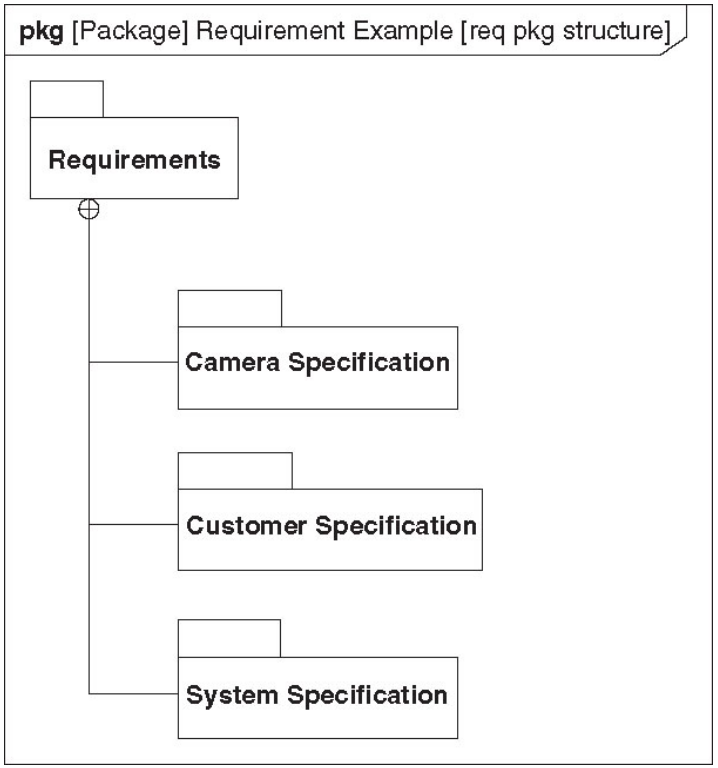
table [Requirement] Camera Decision [Requirements Tree]

id	name	relation	id	name	Rationale
D1	Sensor Decision	derivedFrom	S1.1	24/7 Operation	Using a camera is the most cost-effective way of meeting these requirements. See trade study T1.
		derivedFrom	S1.2	Weather Operation	Using a camera is the most cost-effective way of meeting these requirements. See trade study T1.

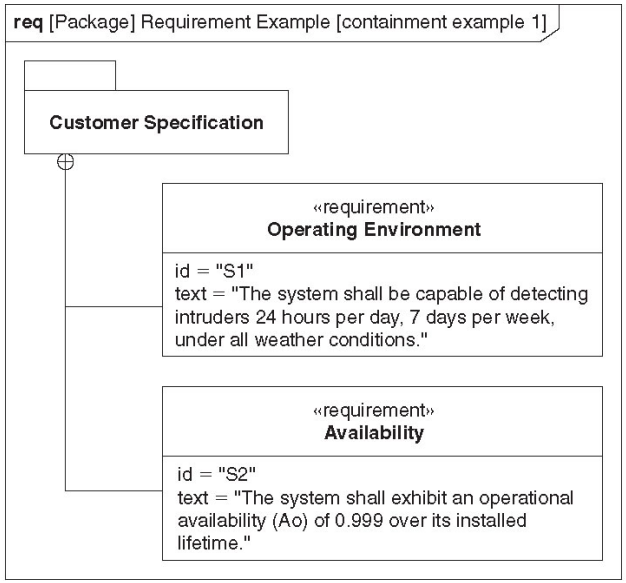
Requirements Diagram

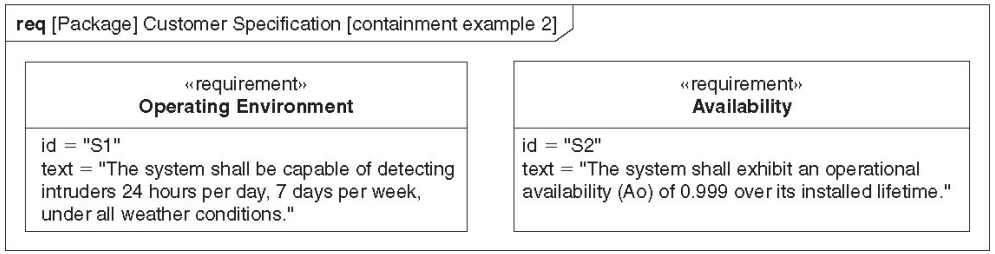


Requirements Diagram

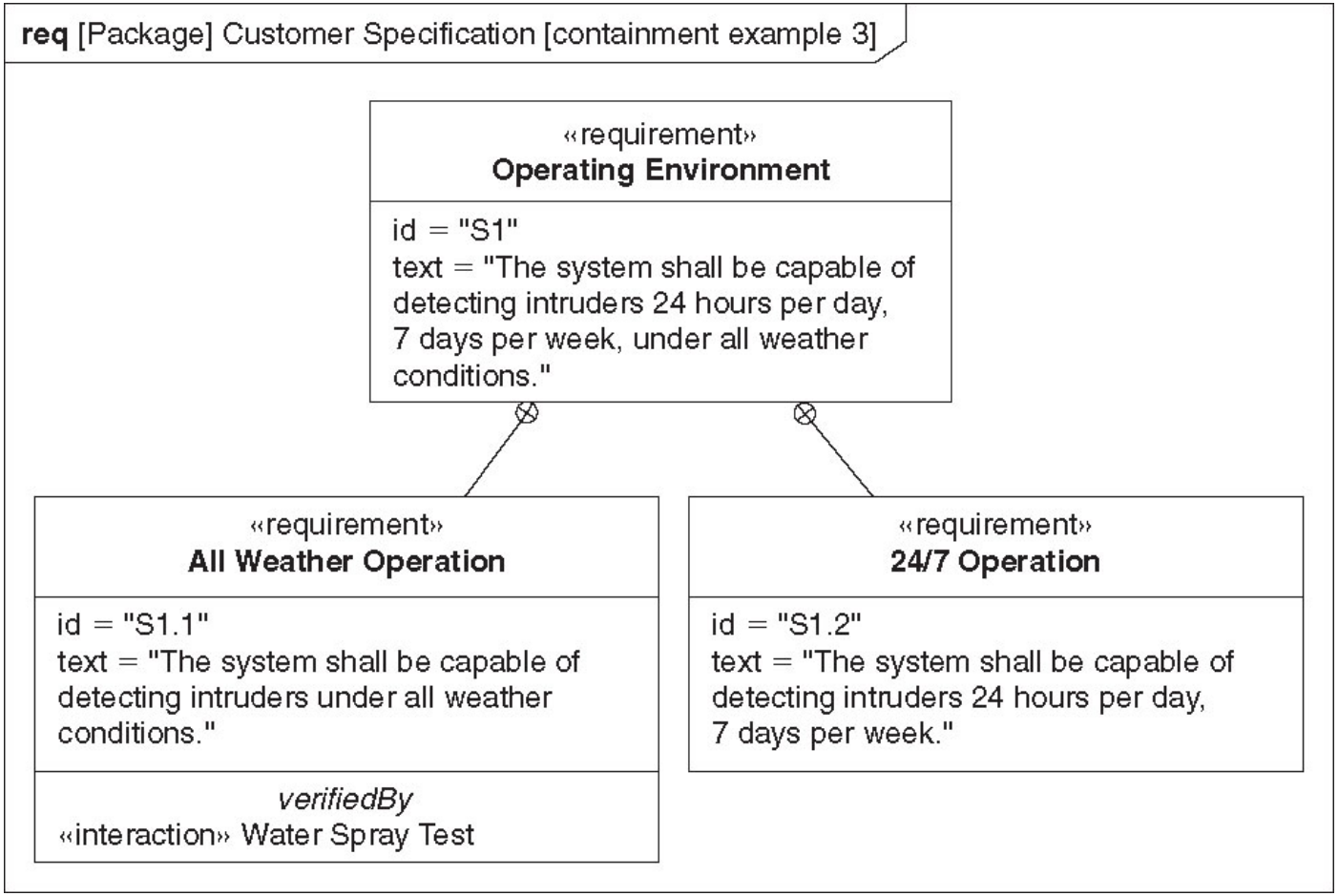


Requirements Diagram

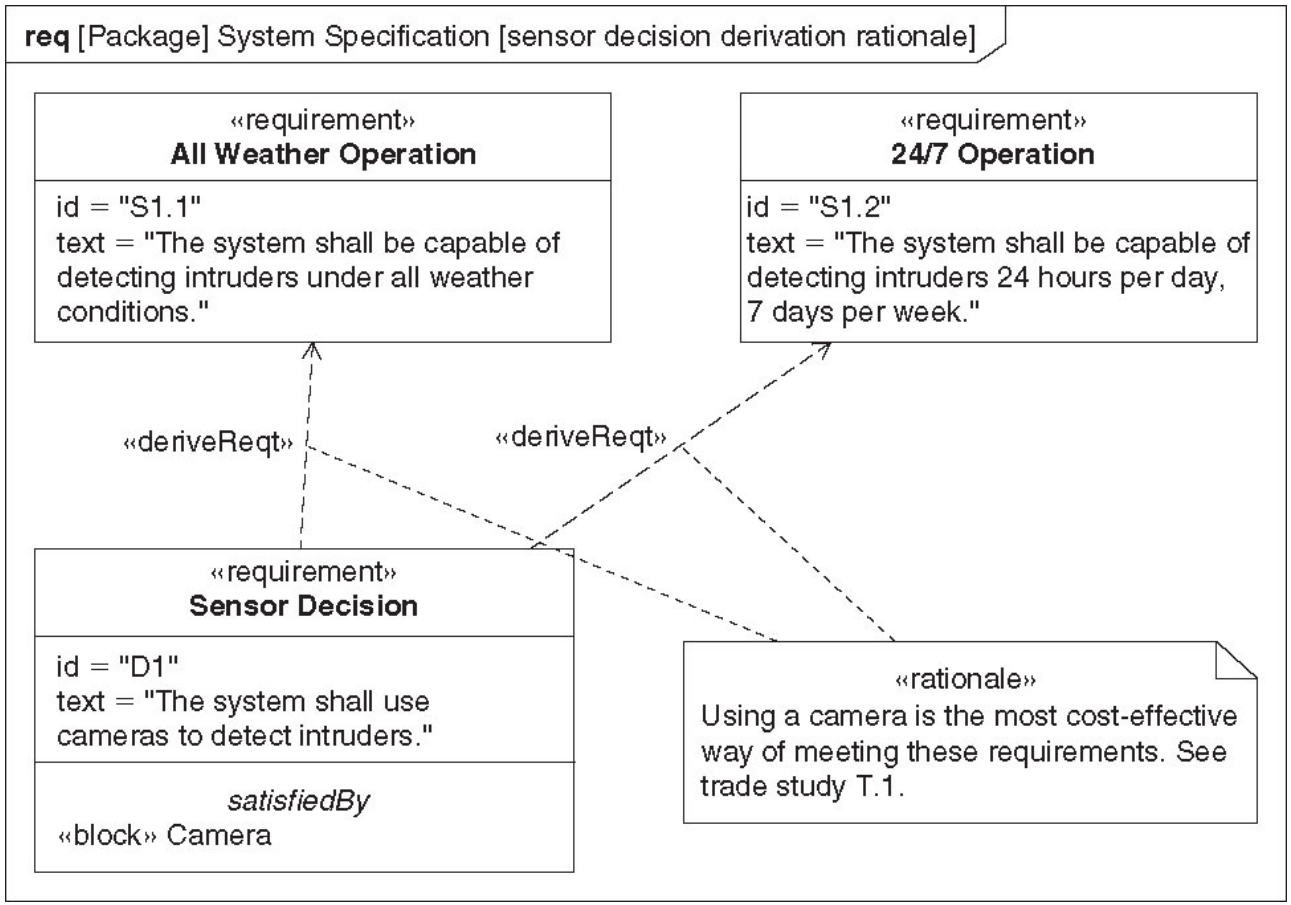




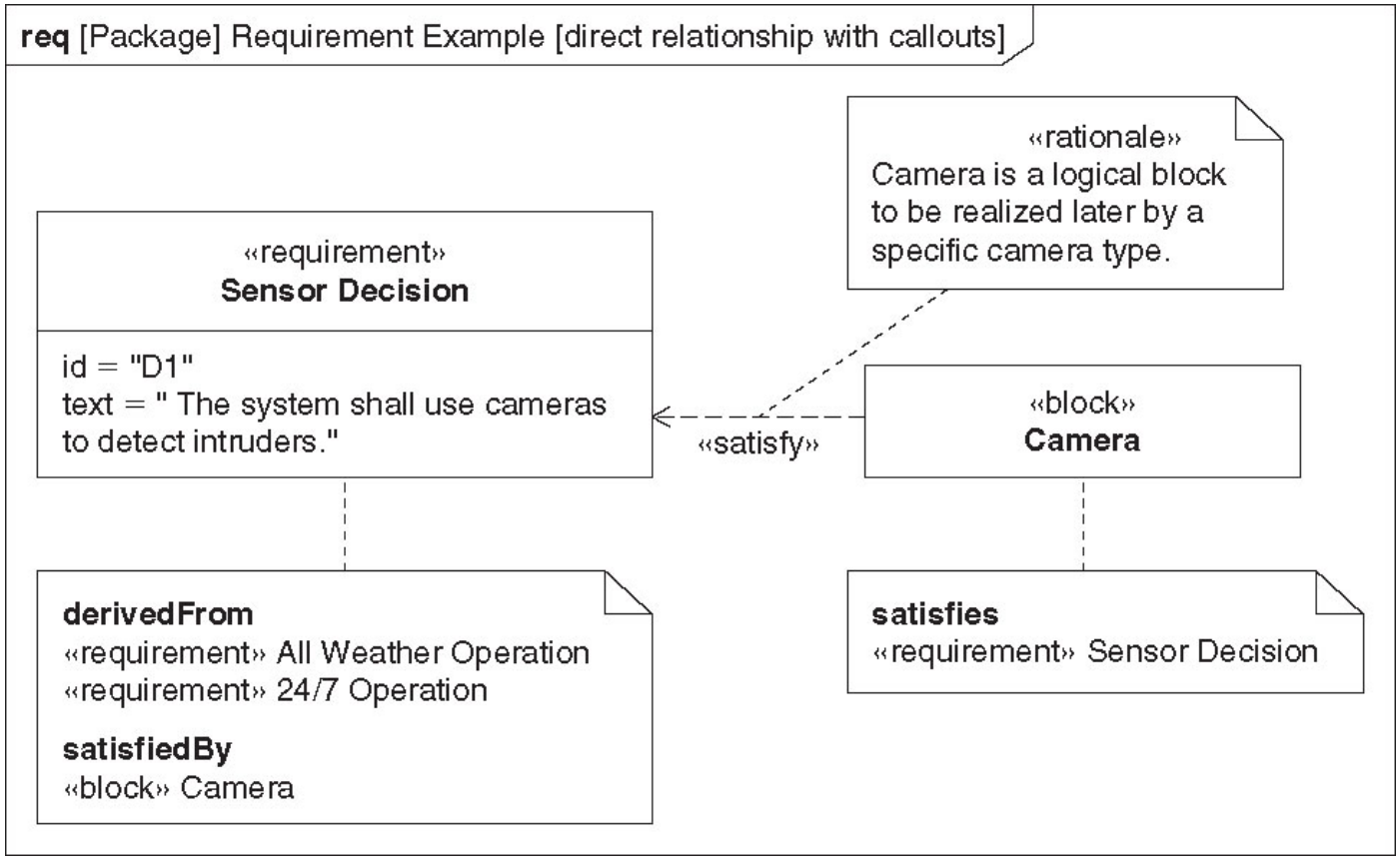
Requirements Diagram



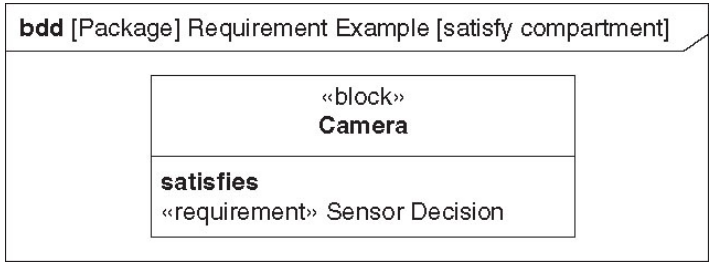
Requirements Diagram

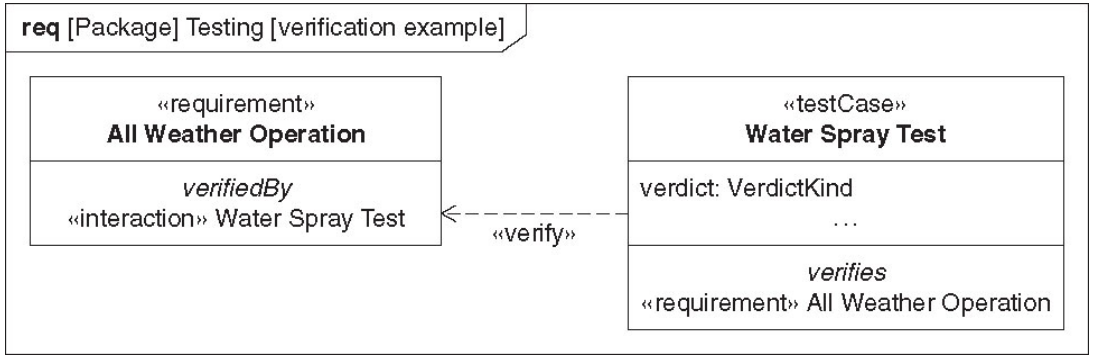


Requirements Diagram

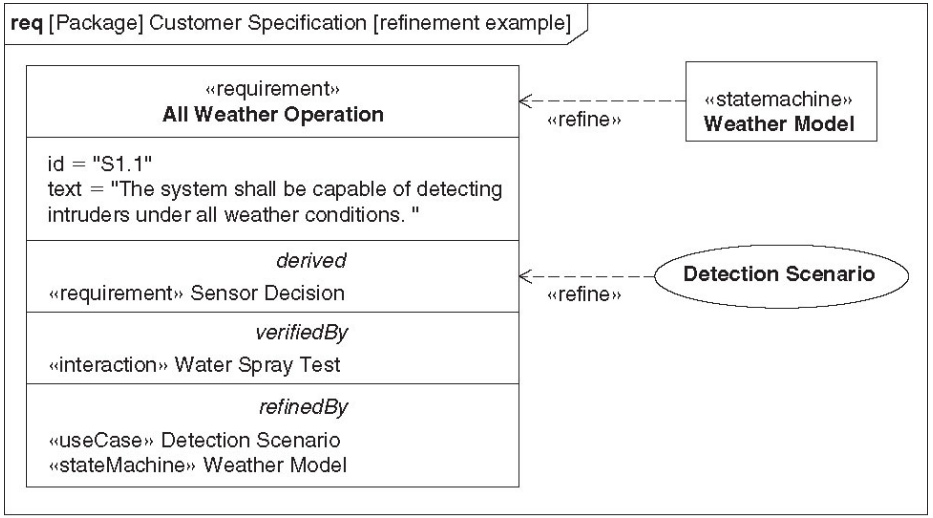


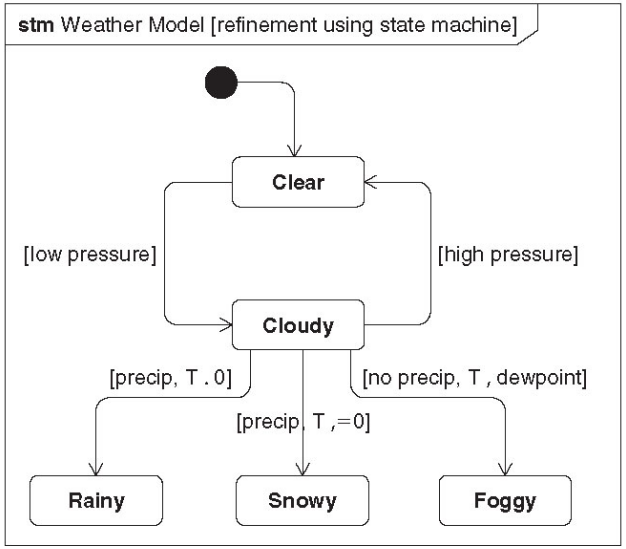
Requirements Diagram



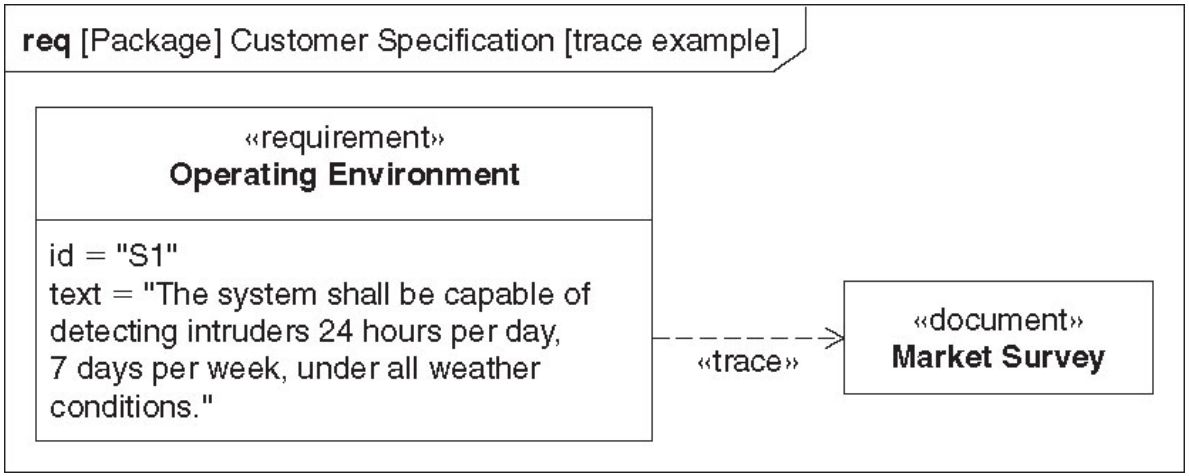


Requirements Diagram





Requirements Diagram



Program Completed

Missouri University of Science & Technology