Modeling Requirements and their Relationships (Part 1 – SysML Concepts)

Section Objectives

- In this Section, you will learn:
  - How to model requirements in SysML
  - How to depict requirement relationships in SysML

Overview

- This section will discuss:
  - Requirement Concepts
    - Attributes of ‘good’ requirements
    - Why model requirements
    - How requirements are depicted using SysML
    - How relationships between requirements are depicted
    - How relationships between requirements and other model elements are depicted
  - Requirements modeling for In-Class Project

Individual Requirement Attributes*

- Unambiguous – every requirement has only one interpretation
- Understandable – the interpretation of each requirement is clear
- Correct – the requirement states something required of the system, as judged by the stakeholders
- Concise – no unnecessary information is included in the requirement
- Traced – each stakeholder's requirement is traced to some document or statement of the stakeholders
- Traceable – each derived requirement must be traceable to a higher level requirement via some unique name or number
- Design independent – each requirement does not specify a particular solution or a portion of a particular solution
- Verifiable – a finite, cost-effective process can be defined to check that the requirement has been attained

*Source: The Engineering Design of Systems, 2nd edition by Buede

Attributes of a Set of Requirements*

- Unique – requirements are not overlapping or redundant with other requirements
- Complete – (a) everything the system is required to do throughout the system's life cycle is included, (b) responses to all possible inputs throughout the system's life cycle are defined, (c) the document is defined clearly and self-contained, and (d) there are no "to be defined" or "to be reviewed" statements
- Consistent – (a) internal - no two subsets of requirements conflict, and (b) external – no subset of requirements conflicts with external documents from which the requirements are traced
- Comparable – the relative necessity of the requirements is included
- Attainable – Solutions exist within performance, cost, and schedule constraints
- Organized – grouped according to a hierarchical set of concepts

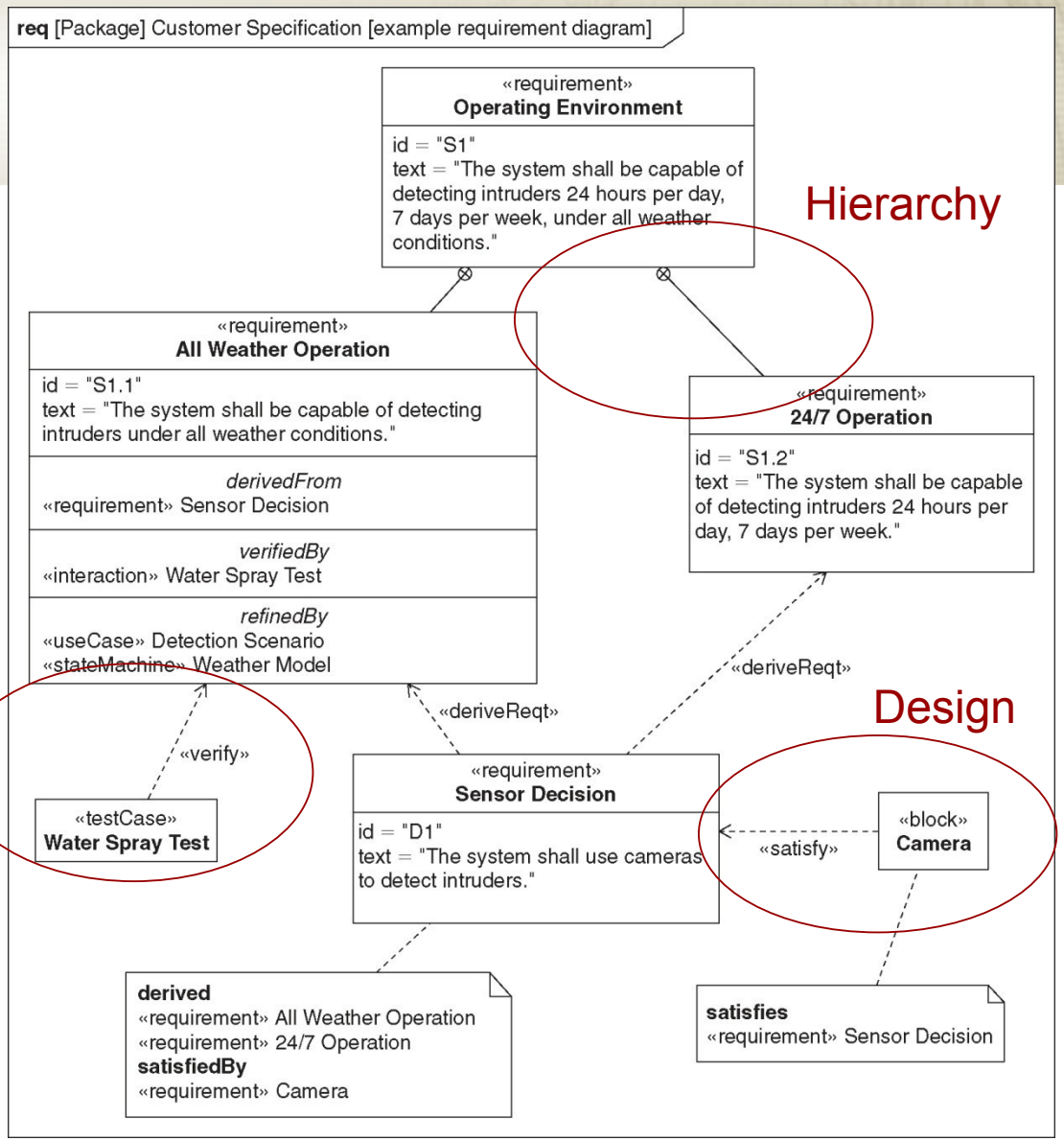
*Source: The Engineering Design of Systems, 2nd edition by Buede

Why Model Requirements

- A requirements model can provide information that helps determine if the requirements meet many of the attributes listed
- SysML requirements modeling provides a 'link' between the text requirements and the rest of the model elements
- Provides 2 types of traceability for requirements
  - Source: Where it came from (e.g. containment, derive, copy, and trace relationships)
  - Target: How it's implemented in the system design (e.g. satisfy, verify, and refine relationships)
- SysML provides a graphical depiction of these relationships
- SysML also provides a means to capture rationale for a specific requirement or relationship

Requirements Diagram

- Requirements diagram graphically depicts:
  - Hierarchy between requirements
  - Relationships between requirements and model elements

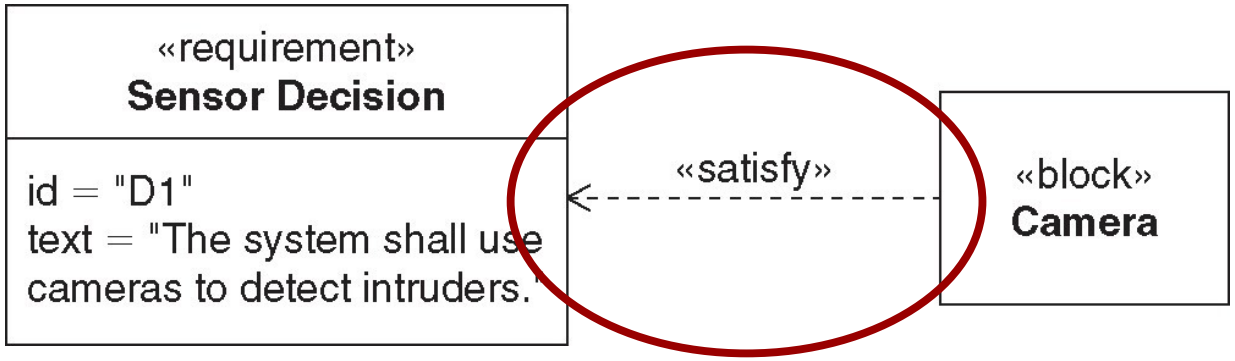


Representing Relationships

- Three ways to depict requirement relationships in SysML:
  - Direct
  - Compartment
  - Callout

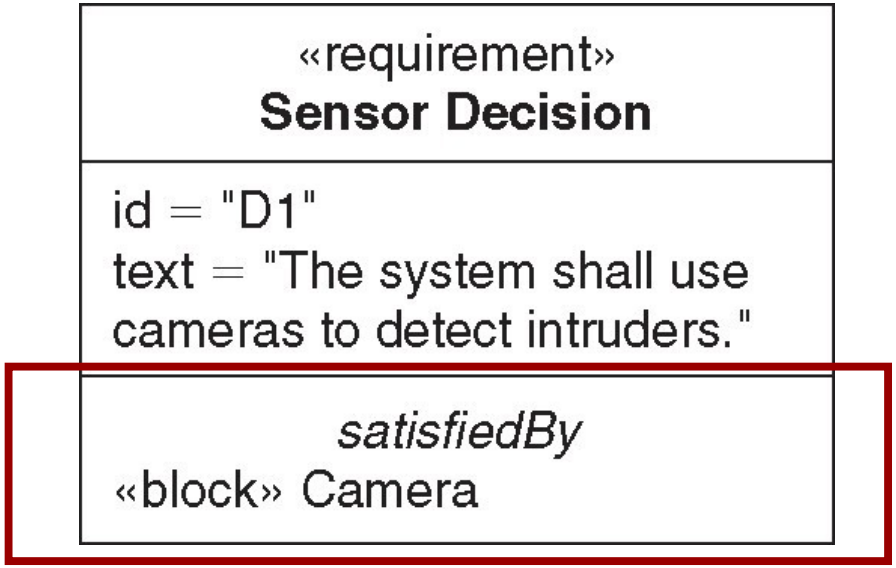
Direct Notation

- Used when the requirement and the related model element appear on the same diagram
- Establishes dependency of model element to requirement in model
- Read figure below as: "The camera satisfies the Sensor Decision requirement".



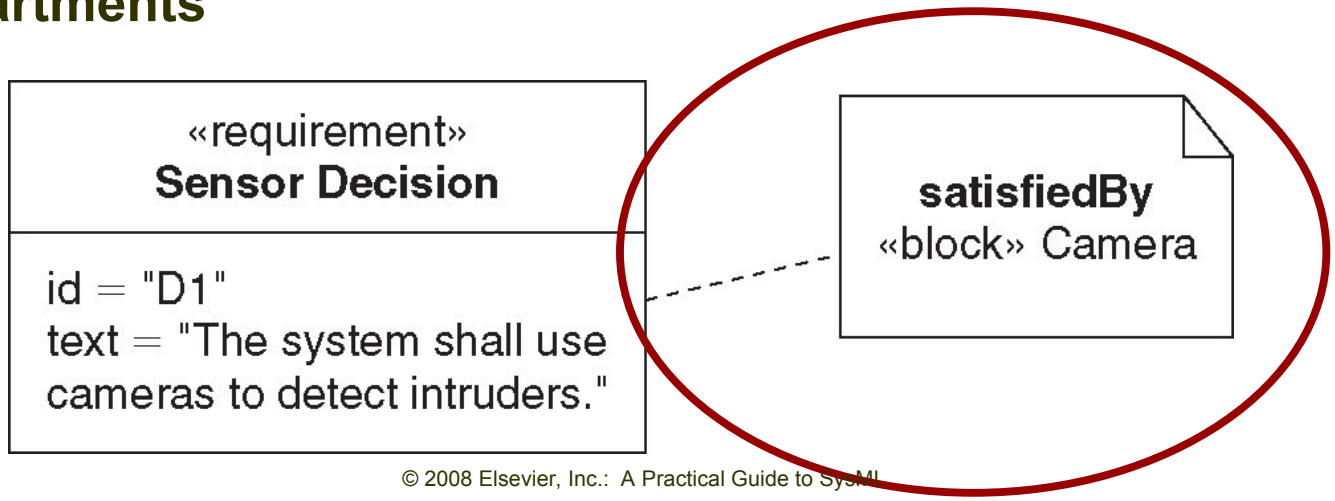
Compartment Notation

- Used when the requirement and model element do not appear on the same diagram.
- Used for model elements such as blocks or requirements that support compartments.



Callout Notation

- Used when the requirement and model element do not appear on the same diagram
  - Uses 'Note' box, rather than model element
- Can be used when the model element or tool does not support compartments

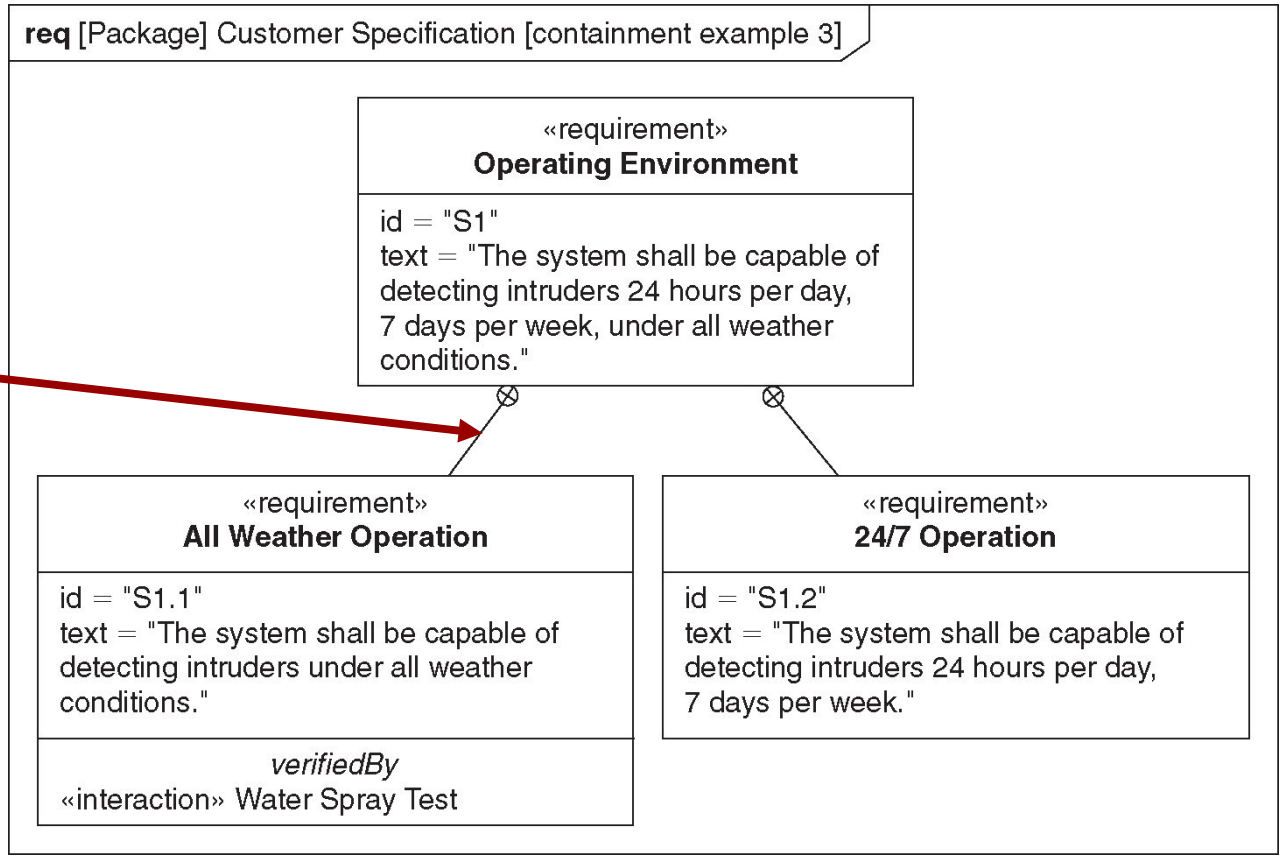


Requirement Relationships in SysML

- Seven types
  - Containment
  - Satisfy
  - Verify
  - Derive
  - Refine
  - Trace
  - Copy

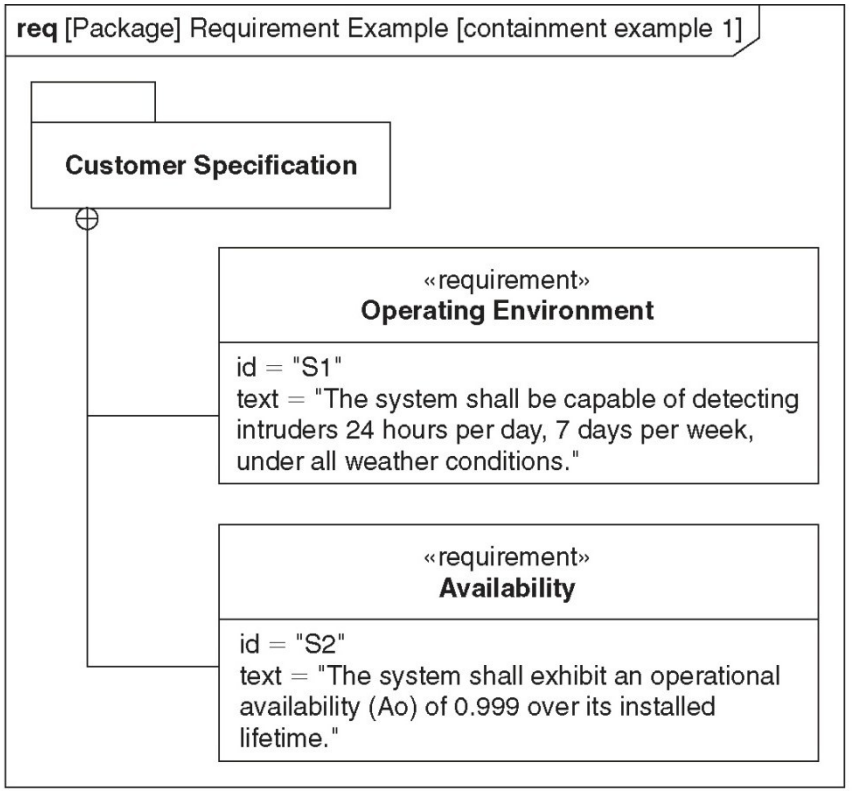
Containment Relationship

- Depicts Requirement Hierarchy
- Crosshair points to the parent requirement



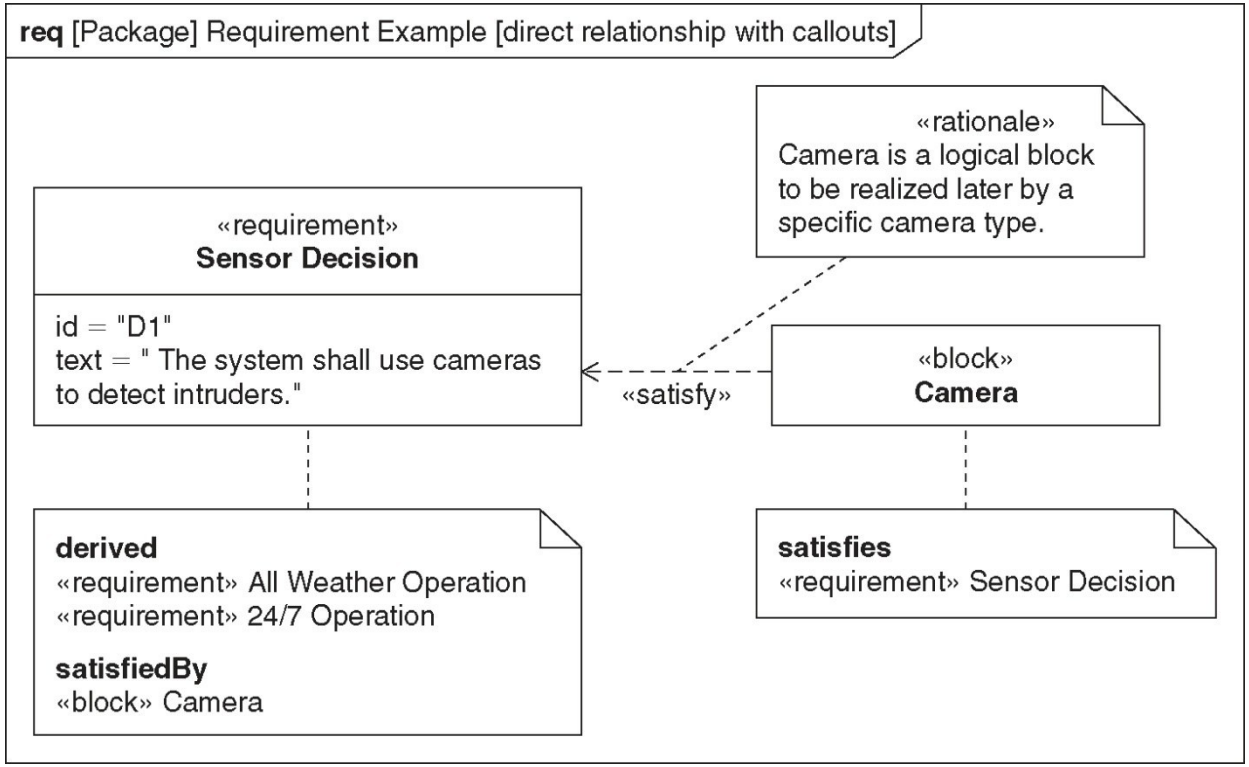
Containment Relationship (cont'd)

- The Containment Relationship can also be used to show which requirements are contained in a Package



Satisfy Relationship

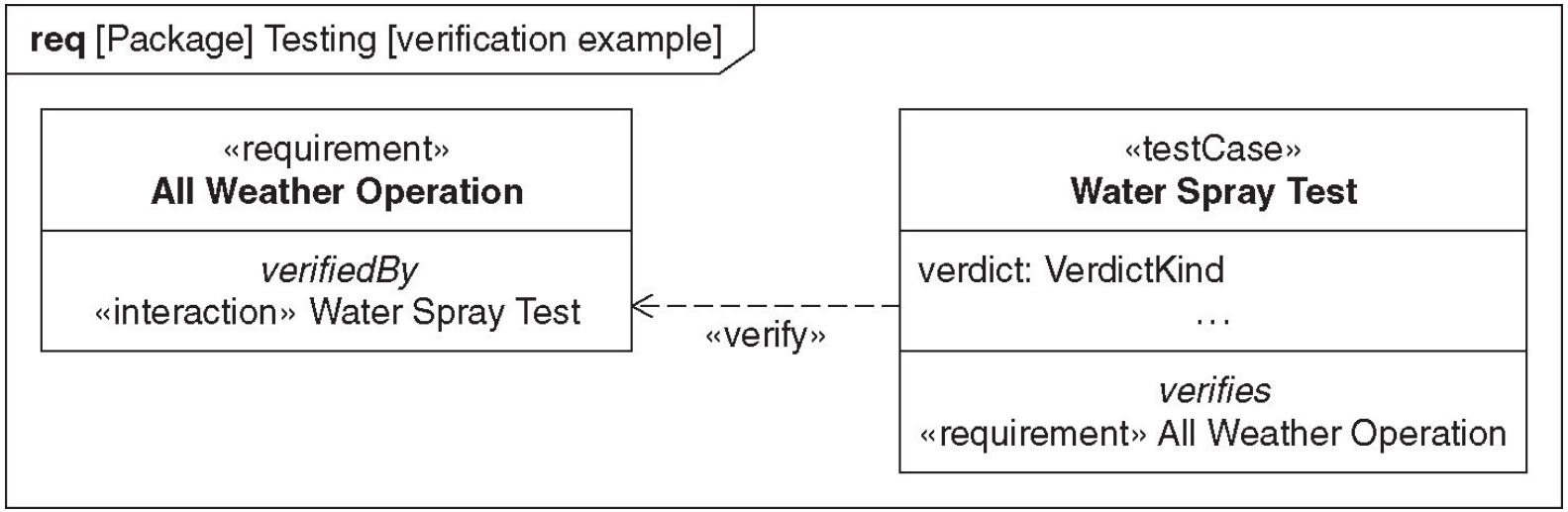
- Depicts when a model element satisfies a requirement.
- Arrowhead points to the requirement



Note: This diagram demonstrates both Direct and Callout Notation to represent a *satisfy* relationship. In practice, only one of these representations is needed.

Verify Relationship

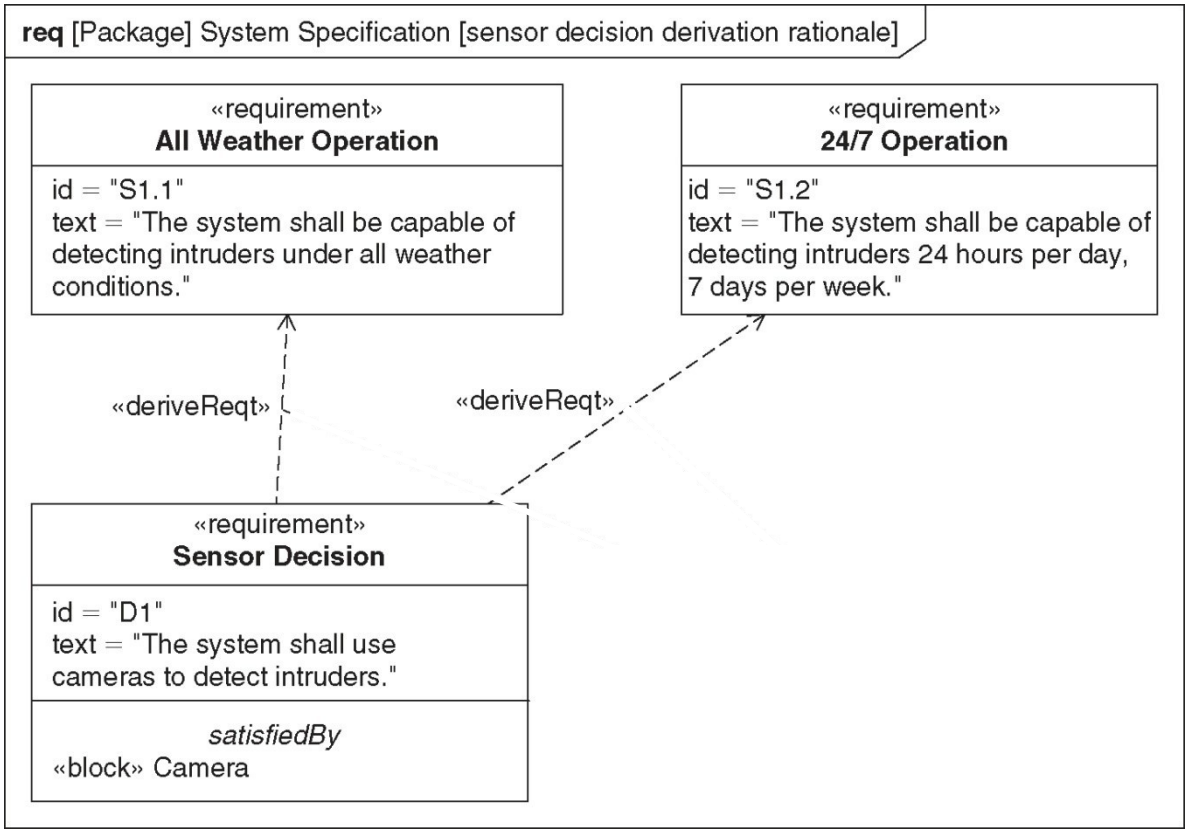
- Used to depict a test case that is used to verify a requirement
- Arrowhead points to the requirement that is verified



Note: This diagram demonstrates both Direct and Compartment Notation to represent a *verify* relationship. In practice, only one of these representations is needed.

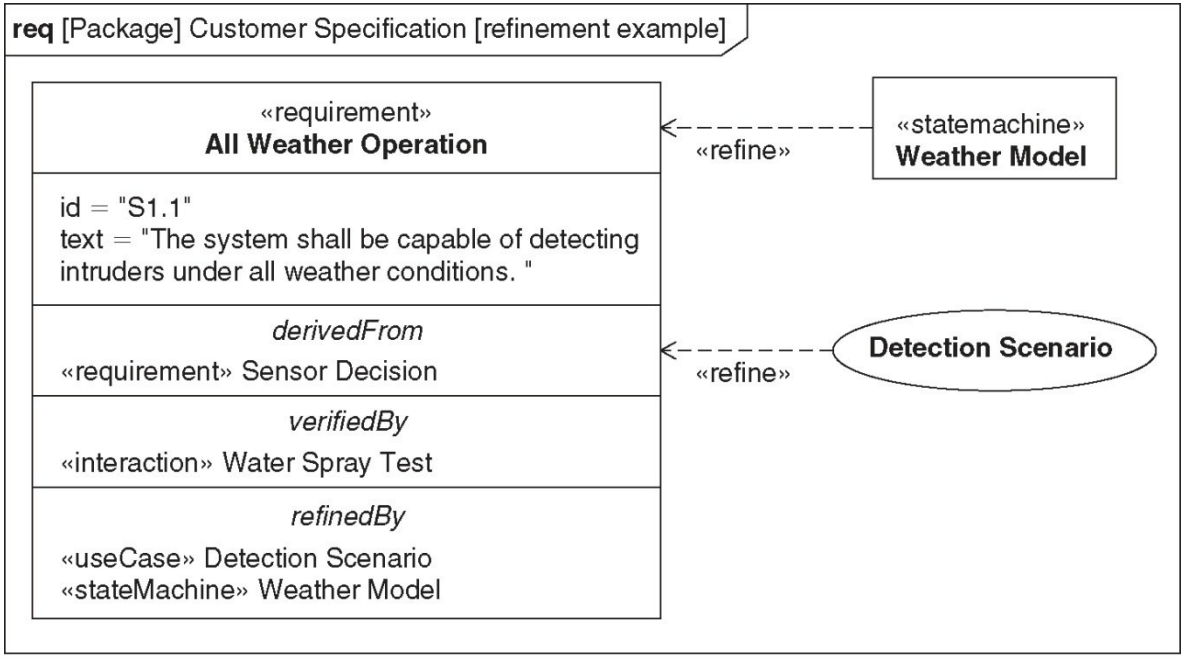
Derive Relationship

- Used when a requirement is derived from another requirement based on analysis
- Arrowhead points to the source requirement



Refine Relationship

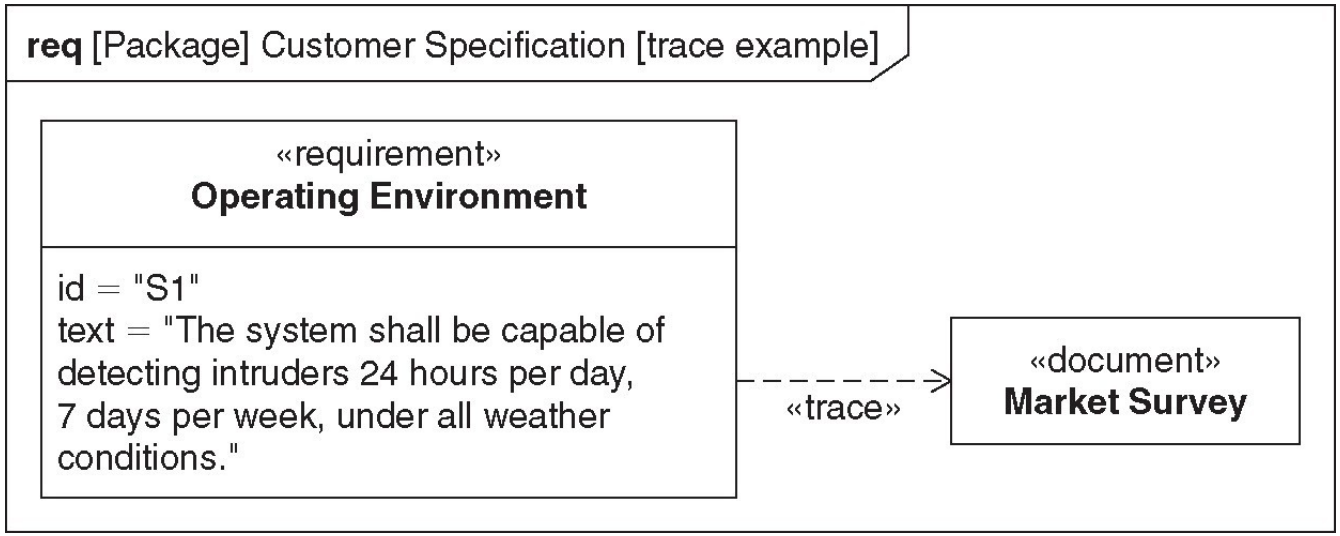
- Used to depict a model element that clarifies a requirement
  - Typically, a use case or behavioral diagram
- Arrowhead points to the requirement



Note: This diagram demonstrates both Direct and Compartment Notation to represent a *refine* relationship. In practice, only one of these representations is needed.

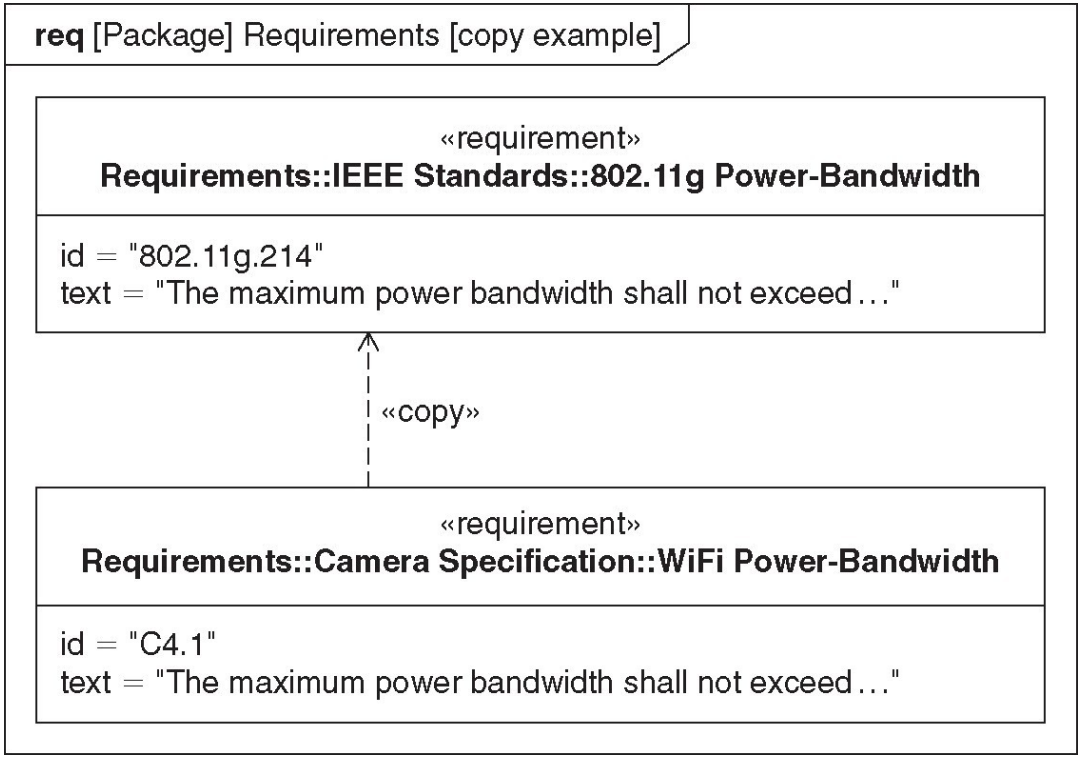
Trace Relationship

- Used to relate requirements to a model element that represents a source for the requirement
- Arrowhead points to the model element



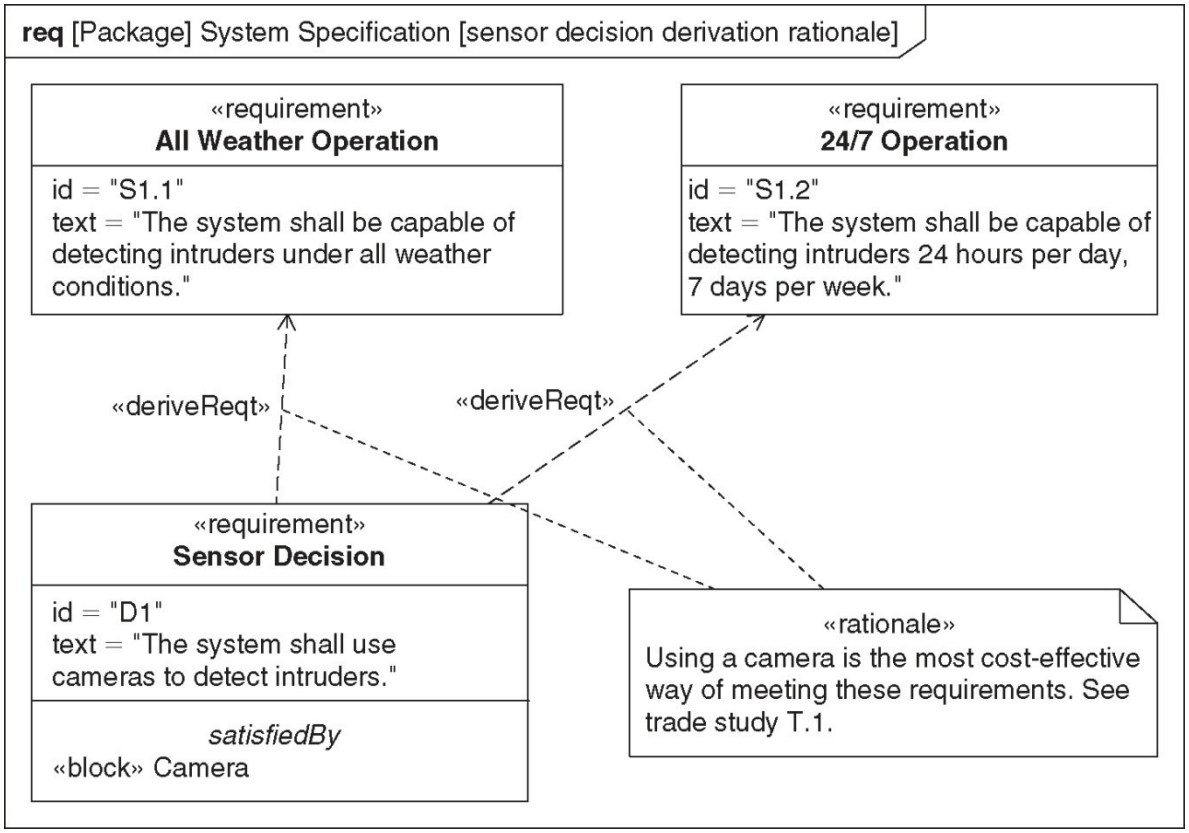
Copy Relationship

- Relates a copy of a requirement to its original
- Arrowhead points to the source requirement
- Typically used when the source requirement is contained in a technical standard.



Depicting Rationale

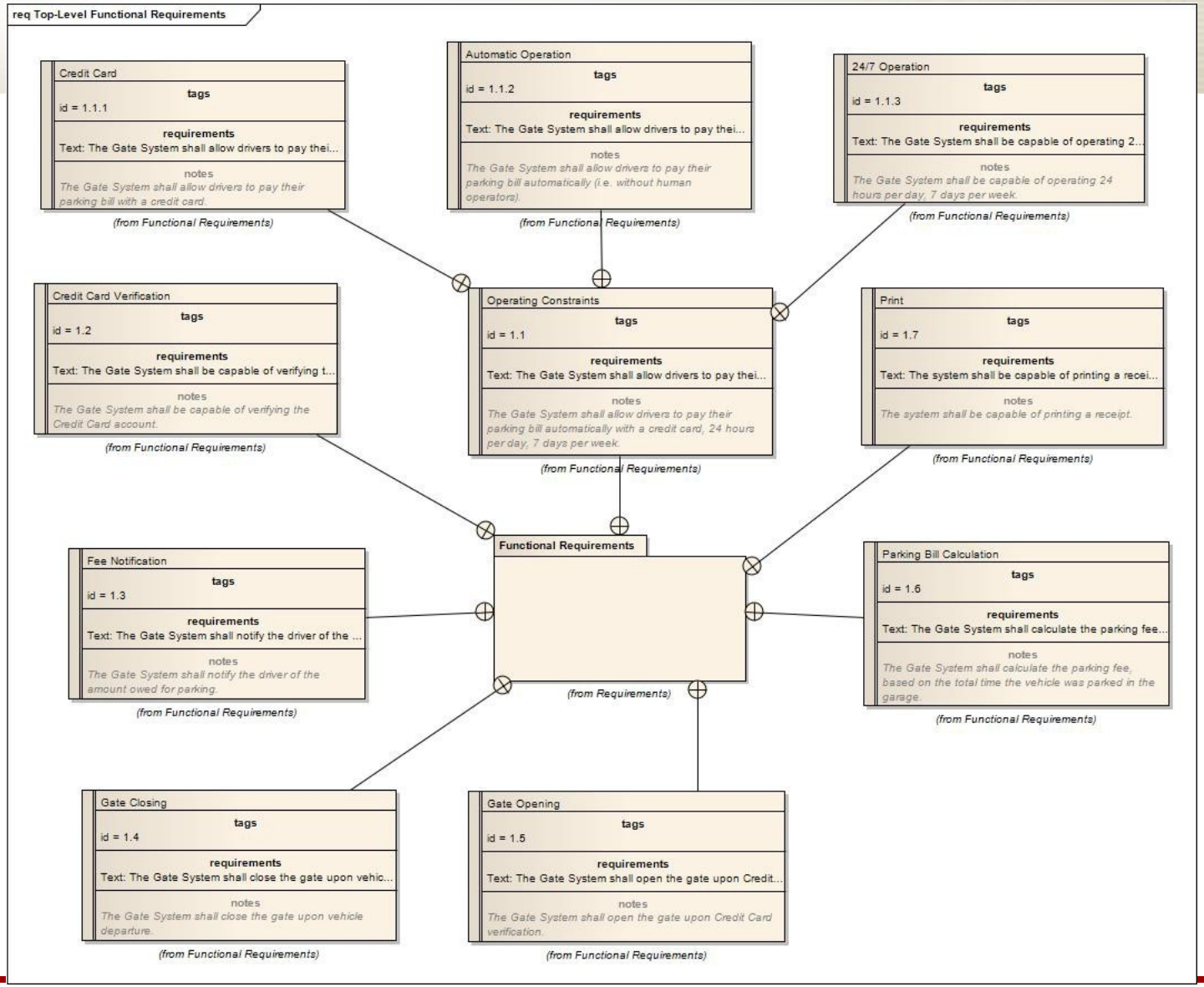
- Used to explain or justify a requirement or a requirement relationship



Requirements Modeling for In-Class Project

- Build Requirements Model for Parking Garage Gate Project using EA
- Parking Garage Gate System Description
  - System is used to collect parking fees from drivers exiting the garage.
  - System reads the Parking Ticket provided by the driver to obtain garage entry date and time.
  - System calculates payment amount based on time the vehicle was in the garage.
  - System has an ‘automated’ process to read and bill the driver’s credit card, (i.e. no attendant is required)
  - System will verify Credit Card account, before billing card.
  - System will automatically open the gate after the credit card account is verified.
  - System will automatically print out a receipt.

Parking Garage Requirements Model



Summary

- Requirement modeling graphically depicts:
  - Hierarchy between requirements
  - Relationships between requirements and the rest of the model elements
- There are three types of notation used to depict requirement relationships: Direct, Compartment, and Callout
- There are seven types of requirement relationships in SysML:
  - Containment
  - Satisfy
  - Verify
  - Derive
  - Refine
  - Trace
  - Copy