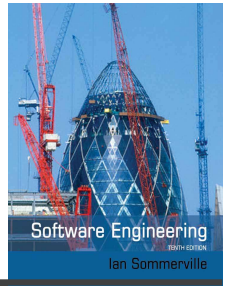
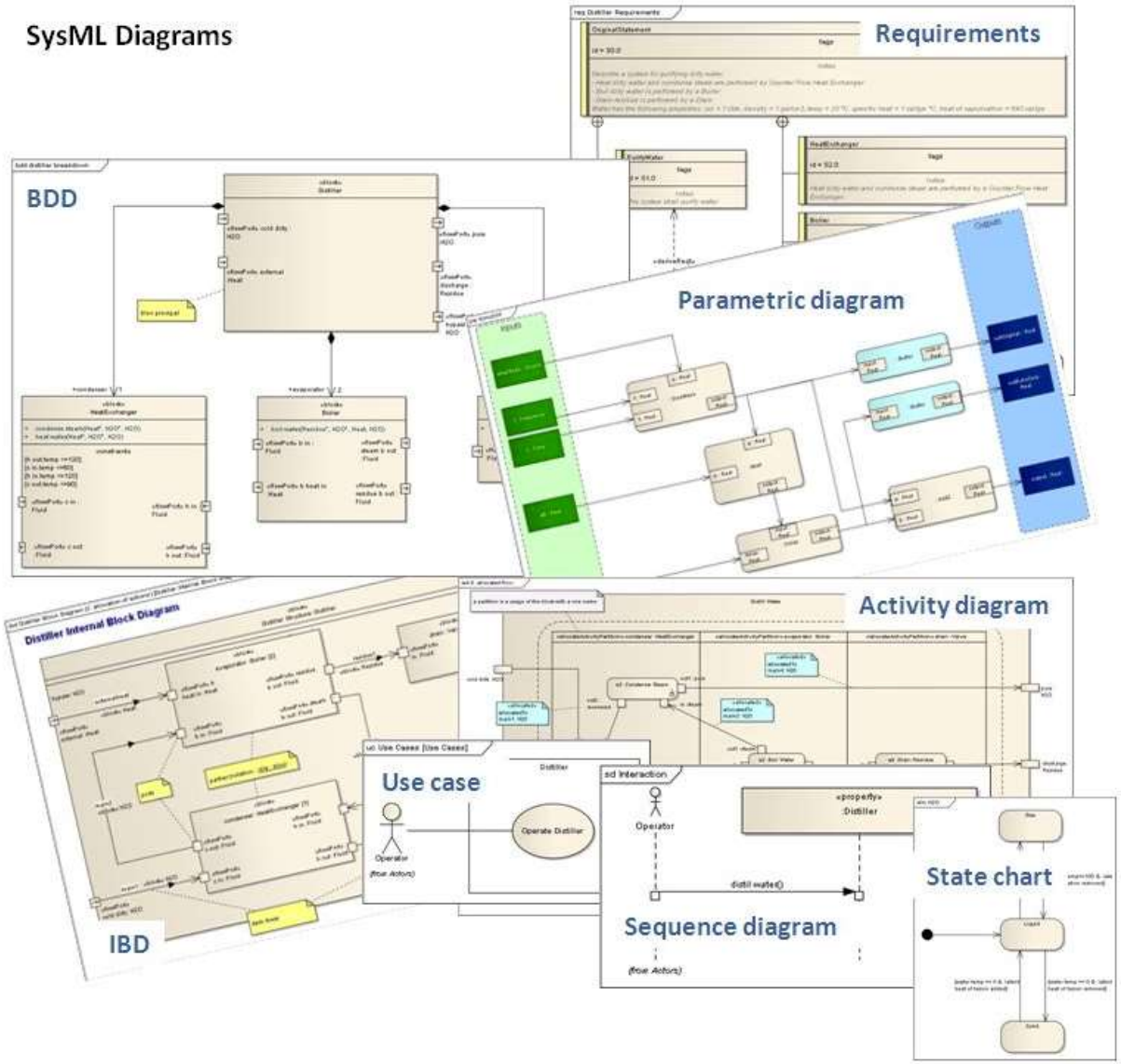
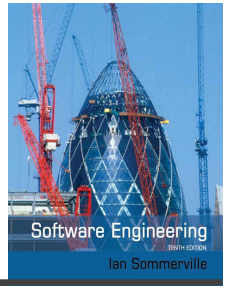


OMG SYSTEMS MODELING LANGUAGE™

SysML diagrams

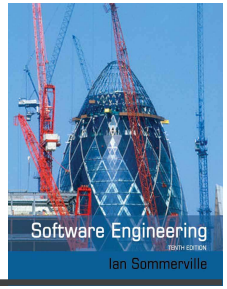


SysML: identity card

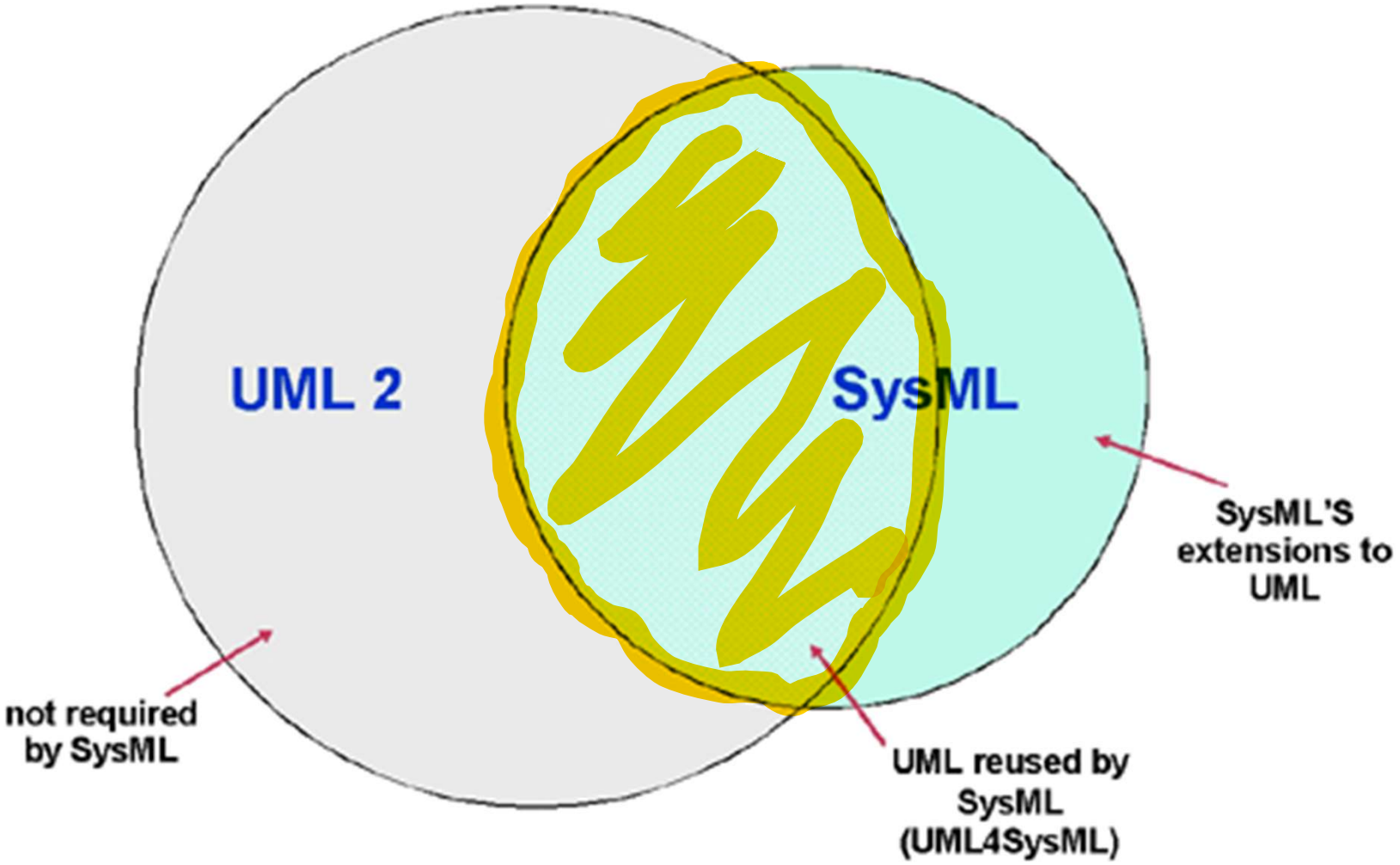


- ✧ Date of birth: 2001!
- ✧ Current version: 1.6 (June 2019)
- ✧ Father: OMG/UML + INCOSE
- ✧ Leading authors
  - Conrad Bock
  - Cris Kobryn
  - Sanford Friedenthal

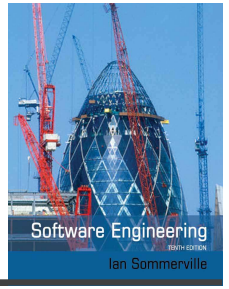
SysML / UML



✧ Relationship between the two

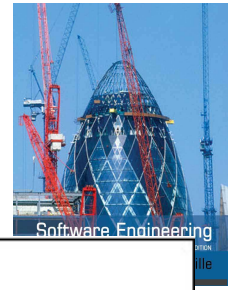


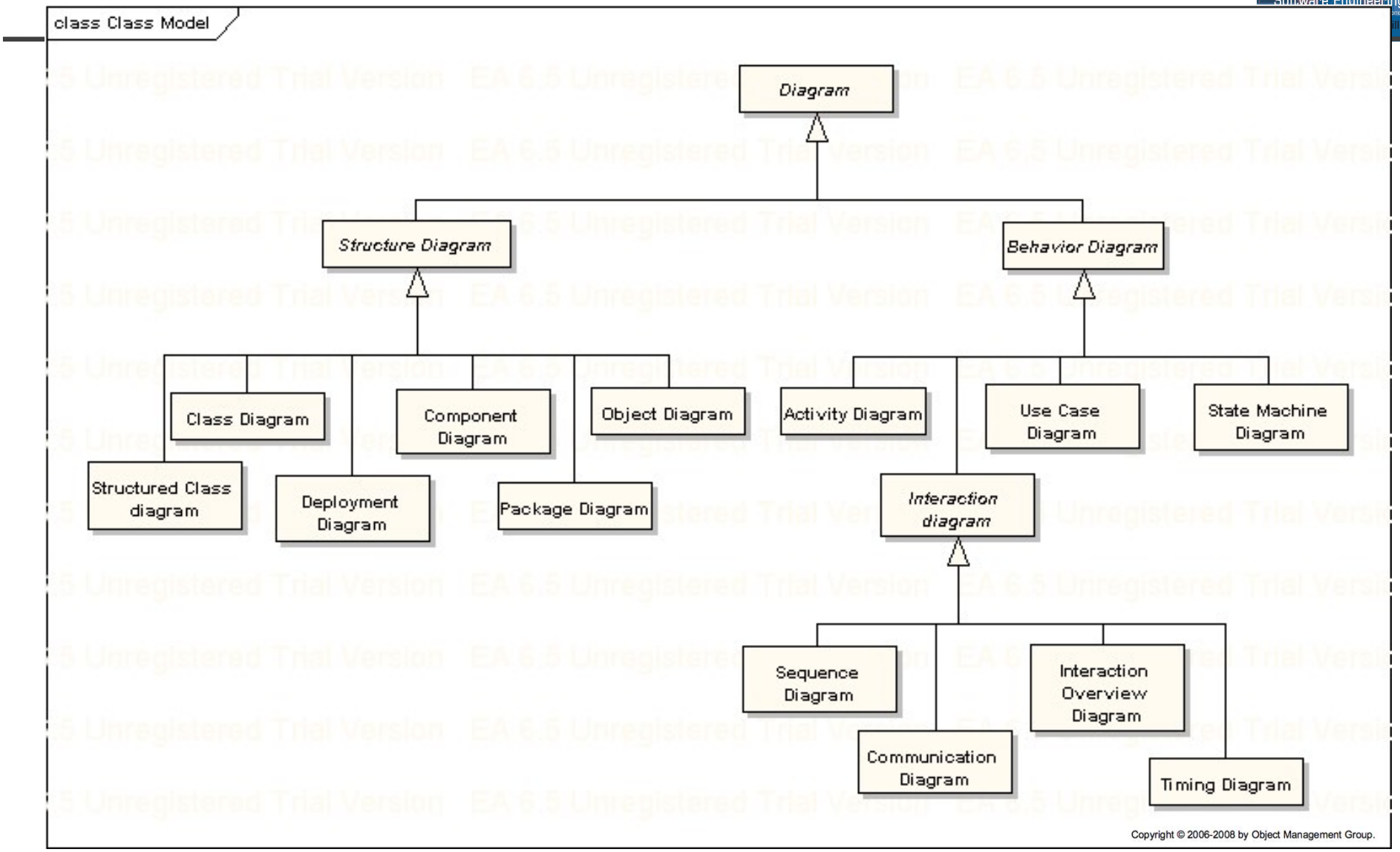
SysML: who's behind



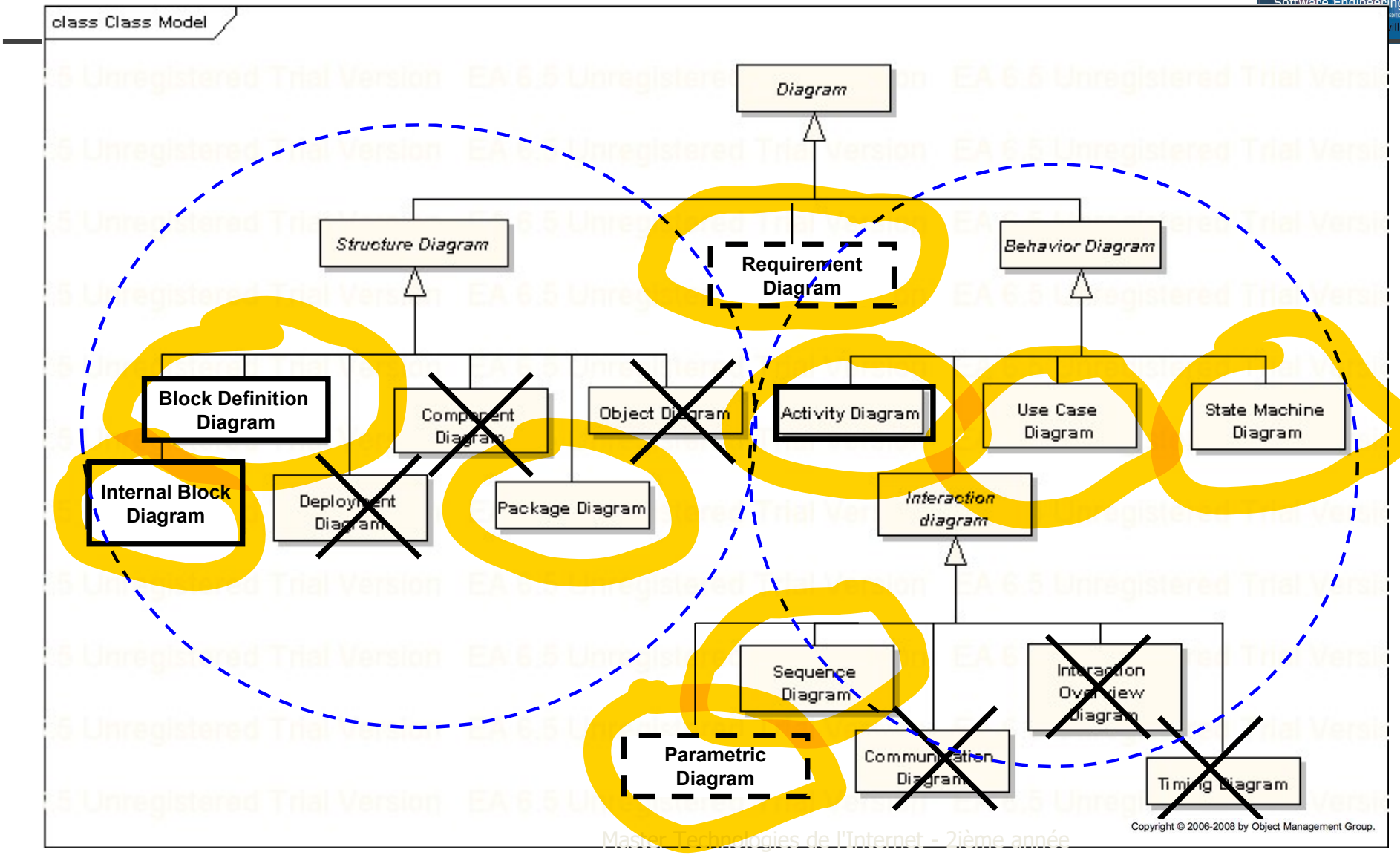
- Industry
  - American Systems, BAE Systems, Boeing, Deere & Company, EADS Astrium, Eurostep, Israel Aircraft Industries, Lockheed Martin, Motorola, NIST, Northrop Grumman, oose.de, Raytheon, Thales, …
- Tool vendors
  - Artisan, EmbeddedPlus, Gentleware, IBM, Mentor Graphics, PivotPoint Technology, Sparx Systems, vitech, …
- Other organisations
  - AP-233, INCOSE, Georgia Institute of Technology, AFIS, …

UML: 13 diagrams

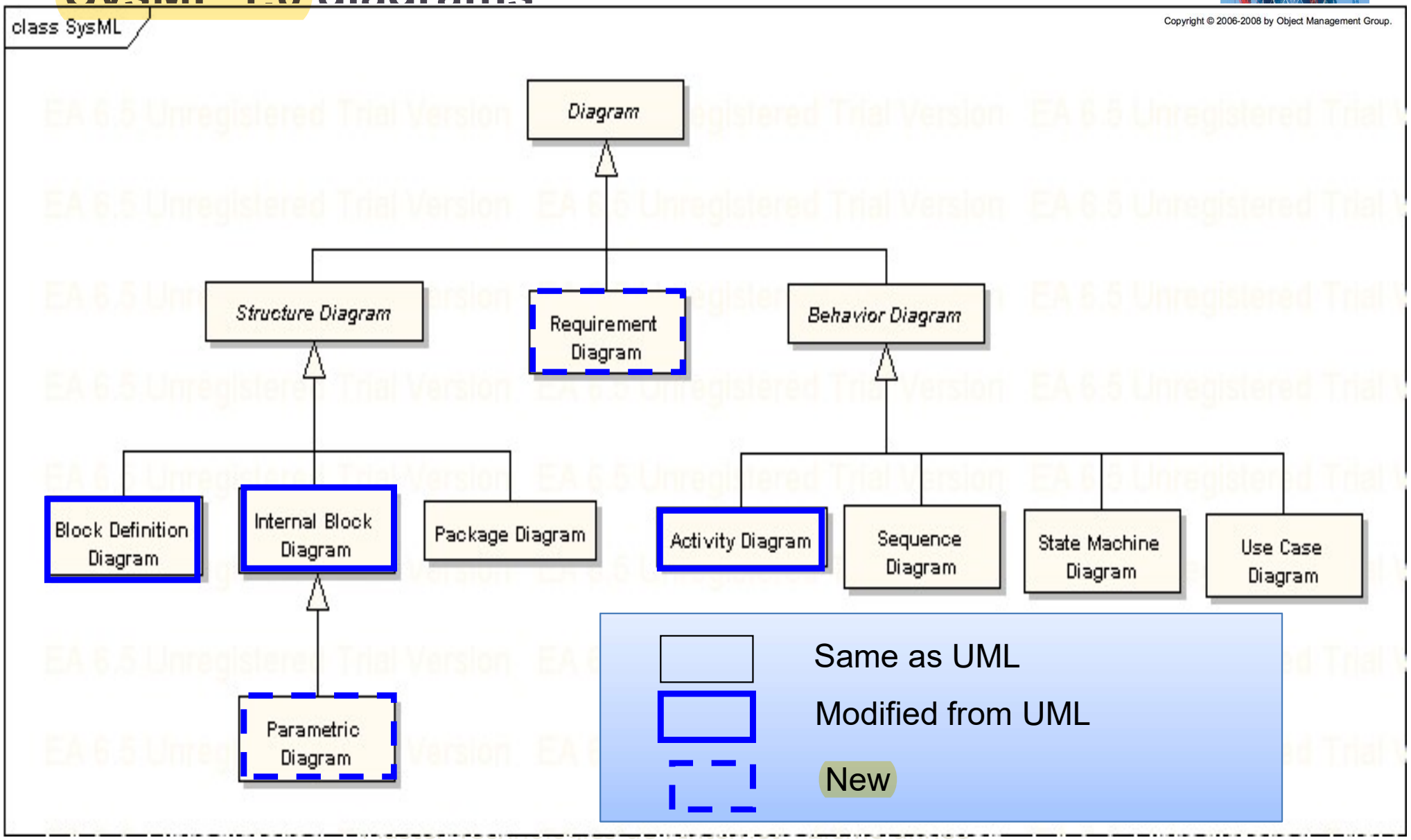




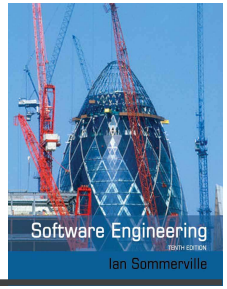
SysML: ~~13~~-7+2=9 diagrams



SysML 1.0 diagrams

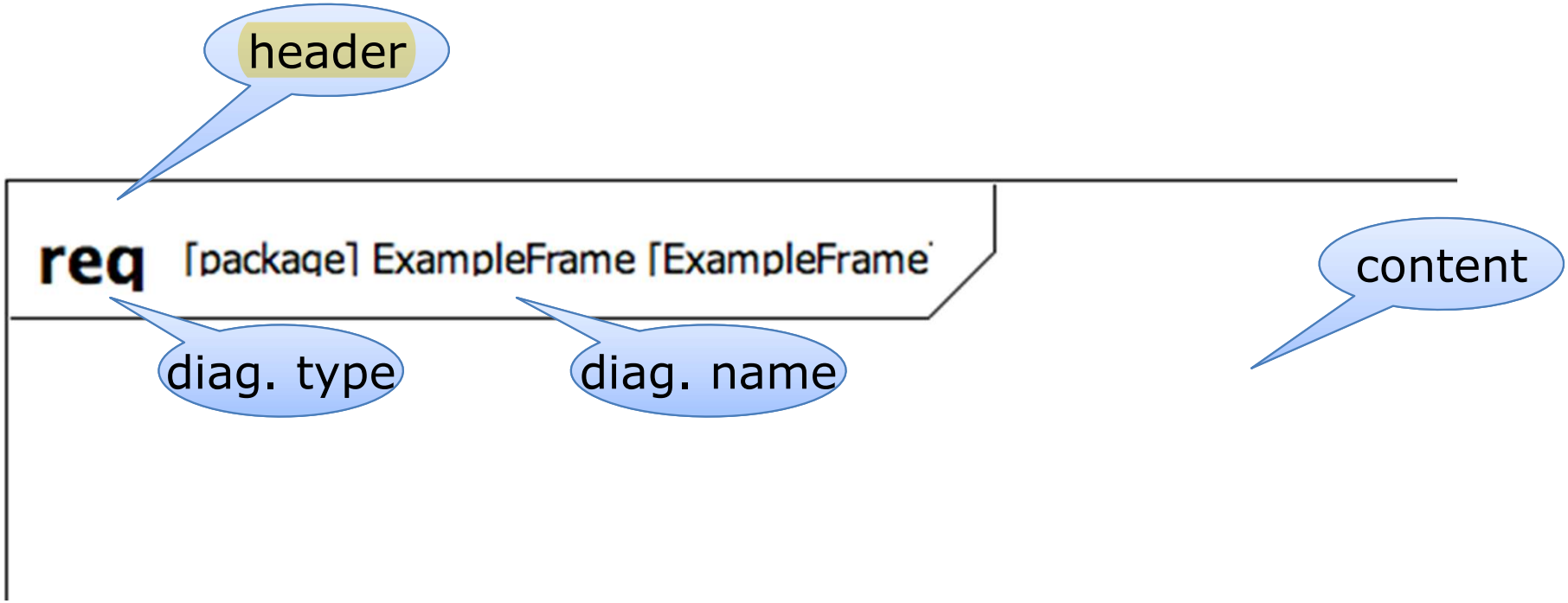


SysML diagram frames

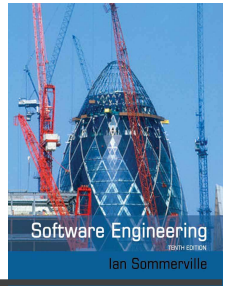


- Each SysML diag. represents a model element
- Each SysML diag. must have a Diagram Frame
- Diagram context is indicated in the header:
  - Diagram kind (req, act, bdd, ibd, sd, etc.)
  - Model element type (package, block, activity, etc.)
  - Model element name
  - User defined diagram name or view name
- A separate diagram description block is used to indicate if the diagram is complete, or has elements elided

SysML diagram frames (e.g.)

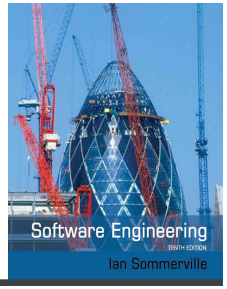


Requirements in SysML



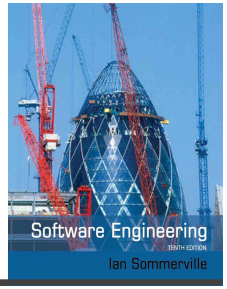
- Requirement diagram
- Other diagrams
  - To link use cases in a Use Case diagram
  - To allocate them to block in a Block Def. Diag.
  - Etc.
- In tables

Why requirements in SysML ?



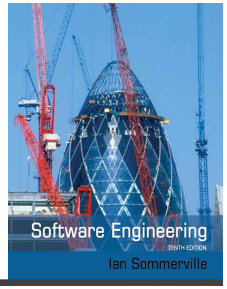
- ✧ Not too much if used only for requirements
- ✧ Added value when related to other elements
- ✧ Easy import/export

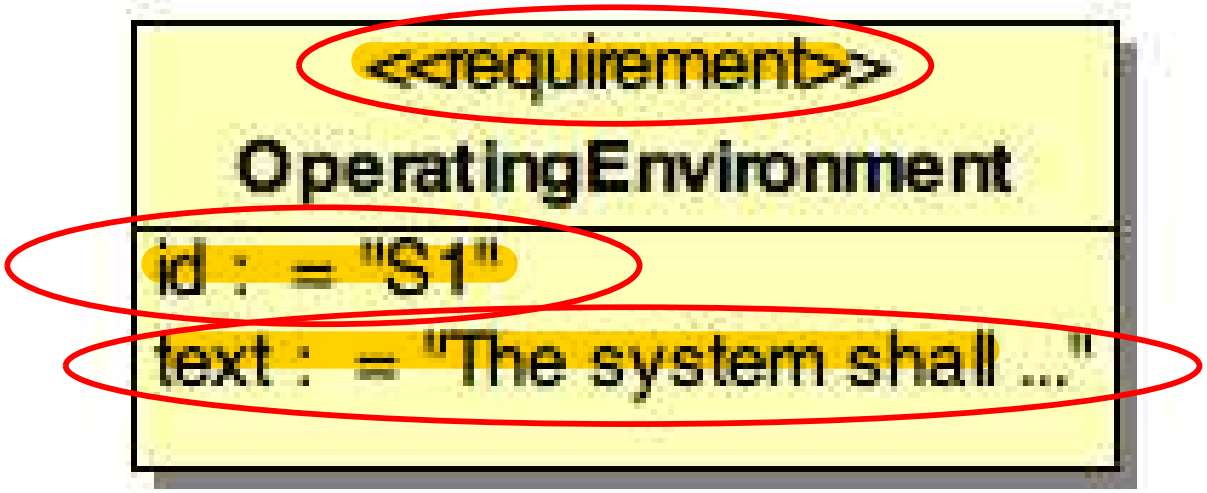
RequirementDiagrams (req)



- <<*requirement*>> allows to represent a text based requirement
  - Includes one identifier *id* and some textual properties
  - Can add user defined properties
  - Can add user defined requirement categories
- Requirements can be
  - decomposed
  - specialized
- Requirement relationships
  - « deriveRqt », « refine »
  - « satisfy », « verify »
  - « trace », « copy »

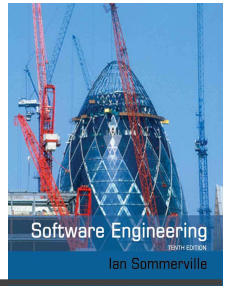
Basic definition





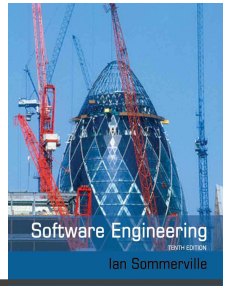
- Stereotype
- Id
- text

A System Requirement

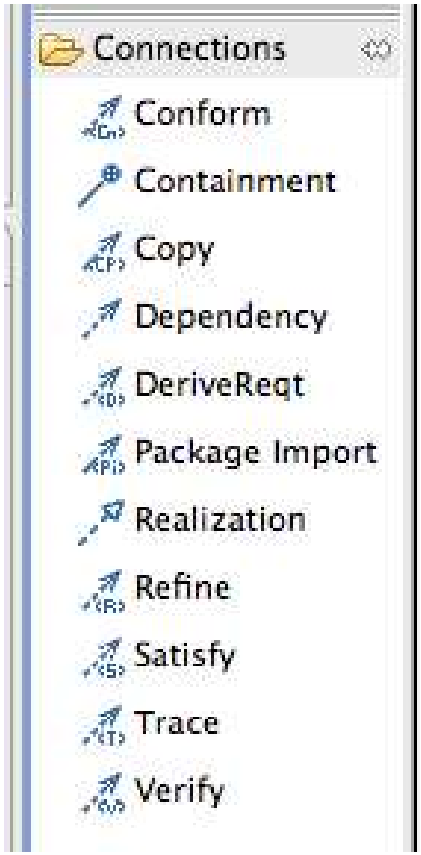


«requirement» **Maximum Acceleration**
Id = "1.4.8" Text = "The vehicle shall accelerate from 0–60 mph in less than 8 seconds under specified conditions"

Requirements Relationships



- ✧ Between requirements
  - Containment
  - Refine
  - Derive
  - Specialize
  - Copy
  - Trace
- ✧ Between requirements and others
  - Satisfy
  - Verify
  - Refine
  - Trace



Requirements Tables

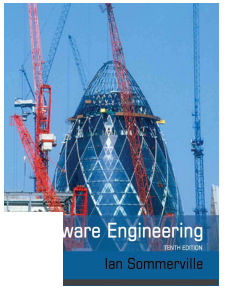


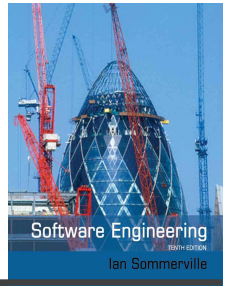
table [requirement] Performance [Decomposition of Performance Requirement]

id	name	text
2	Performance	The Hybrid SUV shall have the braking, acceleration, and off-road capability of a typical SUV, but have dramatically better fuel economy.
2.1	Braking	The Hybrid SUV shall have the braking capability of a typical SUV.
2.2	FuelEconomy	The Hybrid SUV shall have dramatically better fuel economy than a typical SUV.
2.3	OffRoadCapability	The Hybrid SUV shall have the off-road capability of a typical SUV.
2.4	Acceleration	The Hybrid SUV shall have the acceleration of a typical SUV.

table [requirement] Performance [Tree of Performance Requirements]

id	name	relation	id	name	relation	id	name
2.1	Braking	deriveReqt	d.1	RegenerativeBraking			
2.2	FuelEconomy	deriveReqt	d.1	RegenerativeBraking			
2.2	FuelEconomy	deriveReqt	d.2	Range			
4.2	FuelCapacity	deriveReqt	d.2	Range			
2.3	OffRoadCapability	deriveReqt	d.4	Power	deriveReqt	d.2	PowerSourceManagement
2.4	Acceleration	deriveReqt	d.4	Power	deriveReqt	d.2	PowerSourceManagement
4.1	CargoCapacity	deriveReqt	d.4	Power	deriveReqt	d.2	PowerSourceManagement

Relationship with tables

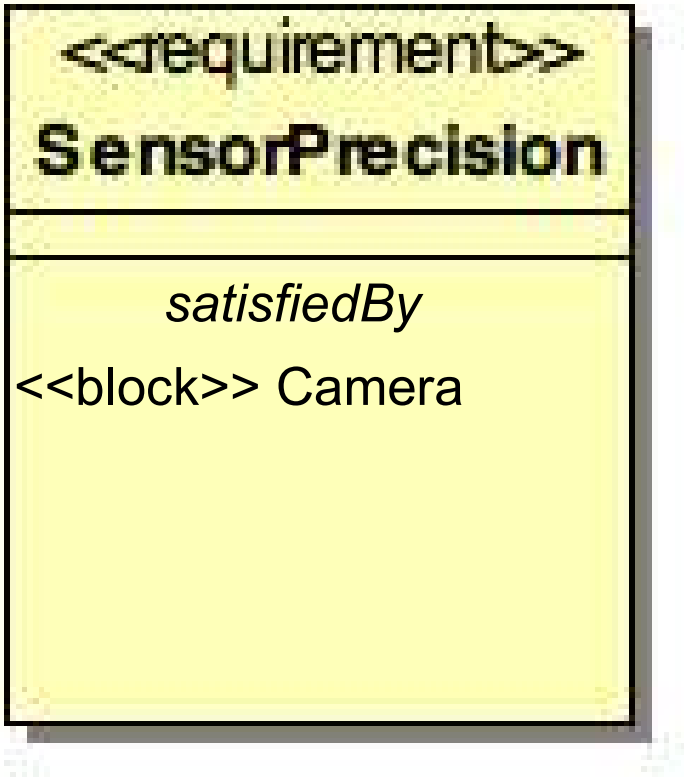


Id	Name	RelatesTo	RelatesHow	Type
TMC17	Traffic management configuration information	{TMC15, TMC16}	deriveReqt	Functional
TMC20	Make function/context obvious	{TMC18, TMC19}	deriveReqt	Functional
TMC21	Education material	{TMC18, TMC19}	deriveReqt	Functional

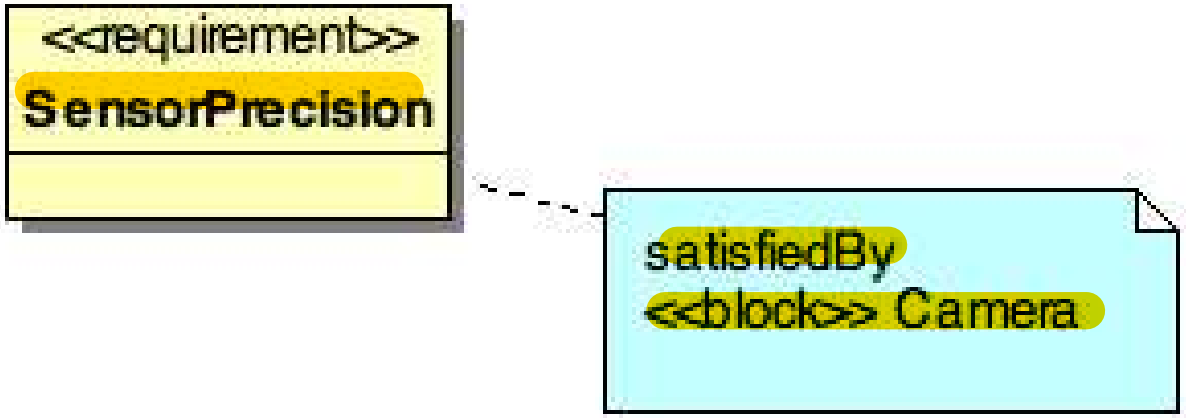
http://ojs.academypublisher.com/index.php/jsw/article/viewFile/03065768/988

Depicting relationship directly

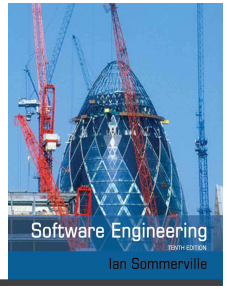
Relationship in compartments

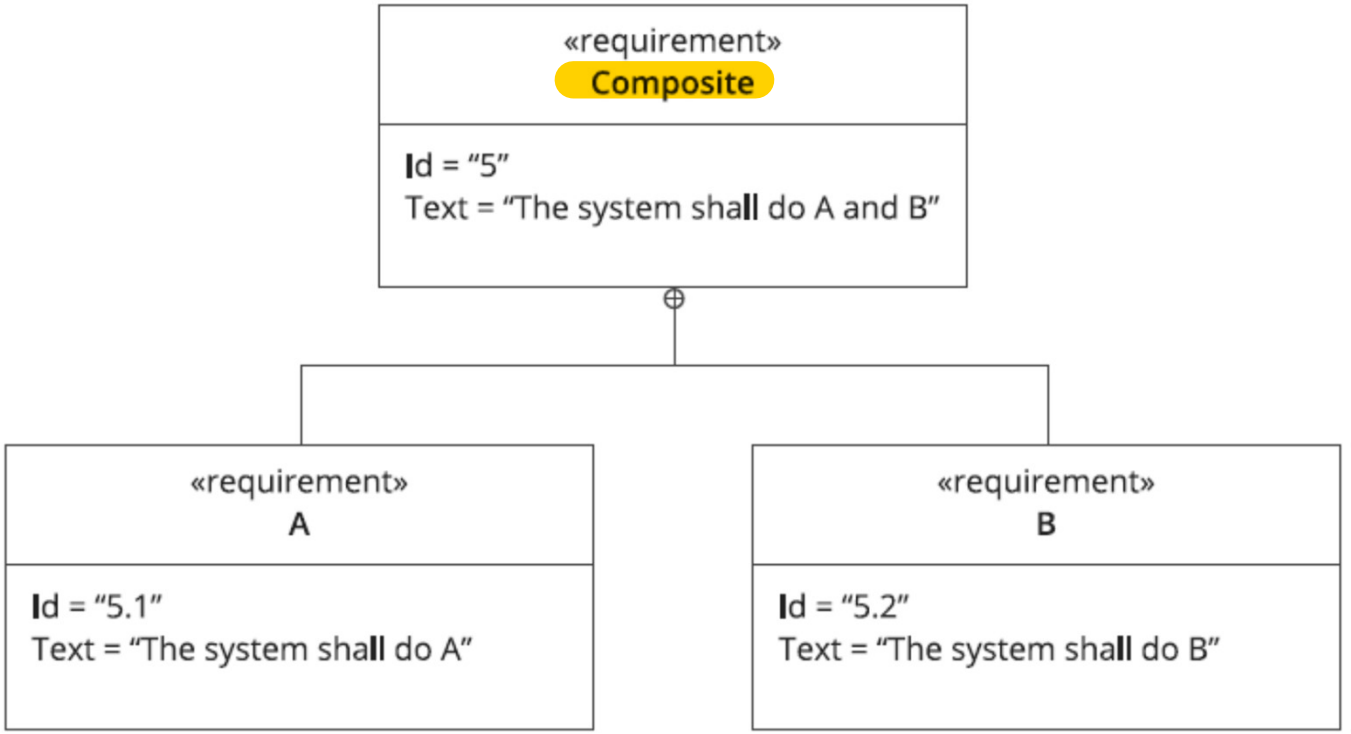


Relationship with callout

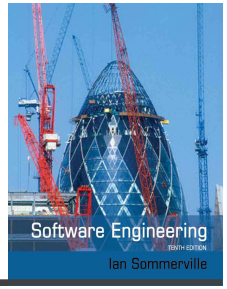


Requirements hierarchies: Containment or Composite

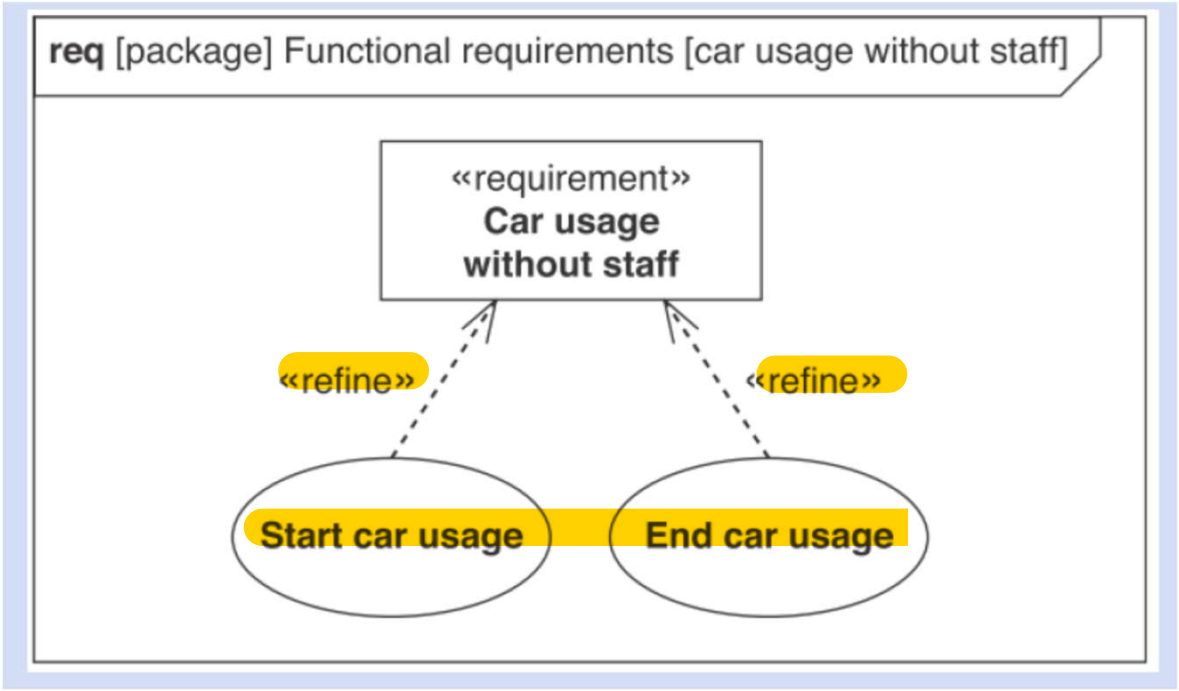




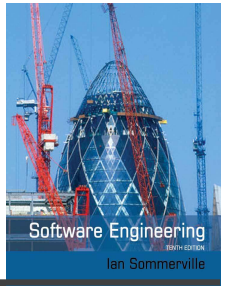
Refine Relationship

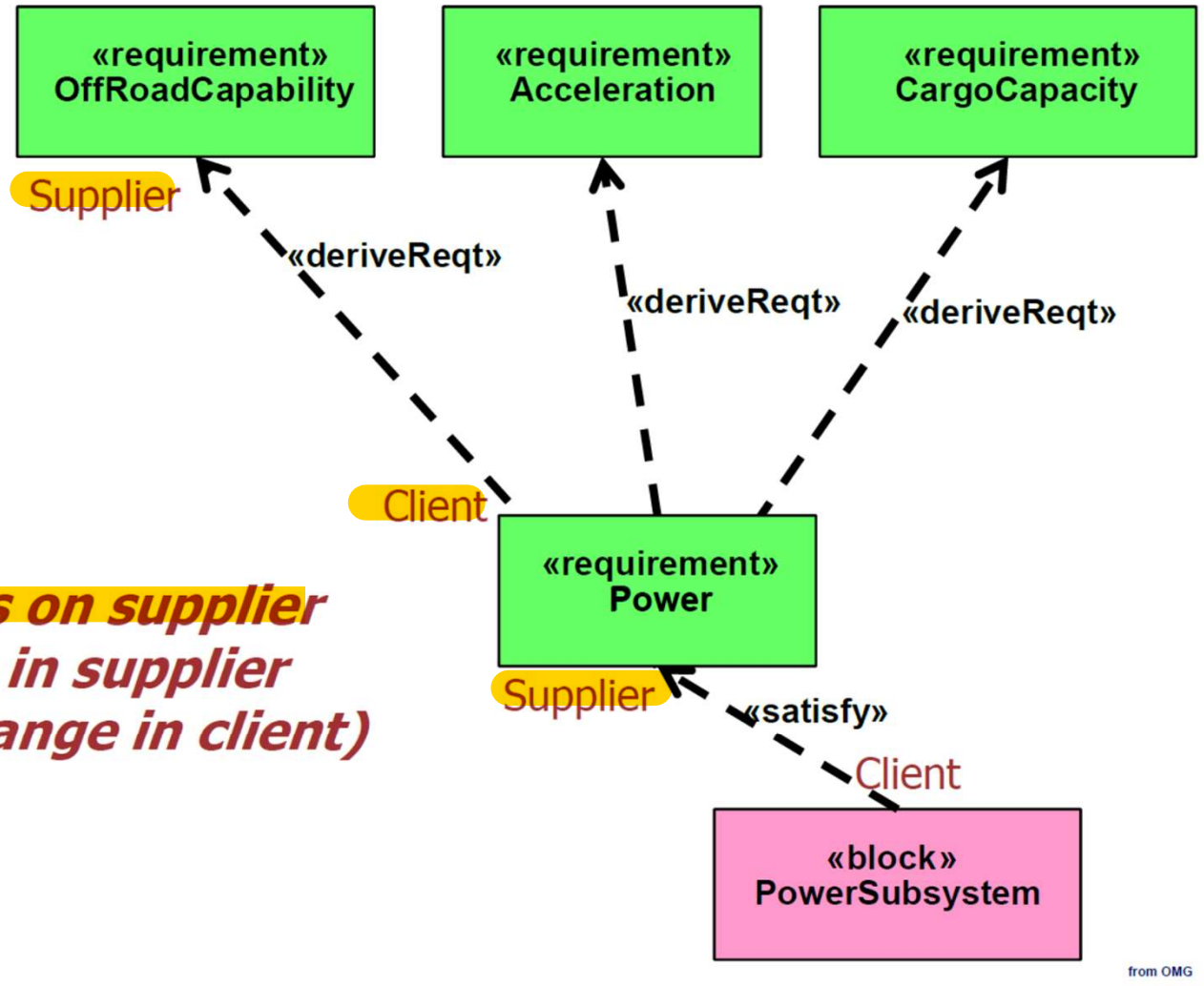


- The refine requirement relationship can be used to describe how a model element, or set of elements, can be used to further refine a requirement.
- For example, a use case or activity diagram may be used to refine a text-based functional requirement.



DeriveReqt relationship

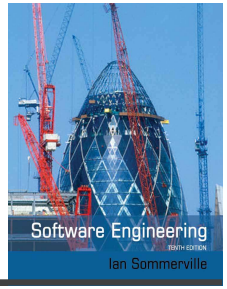


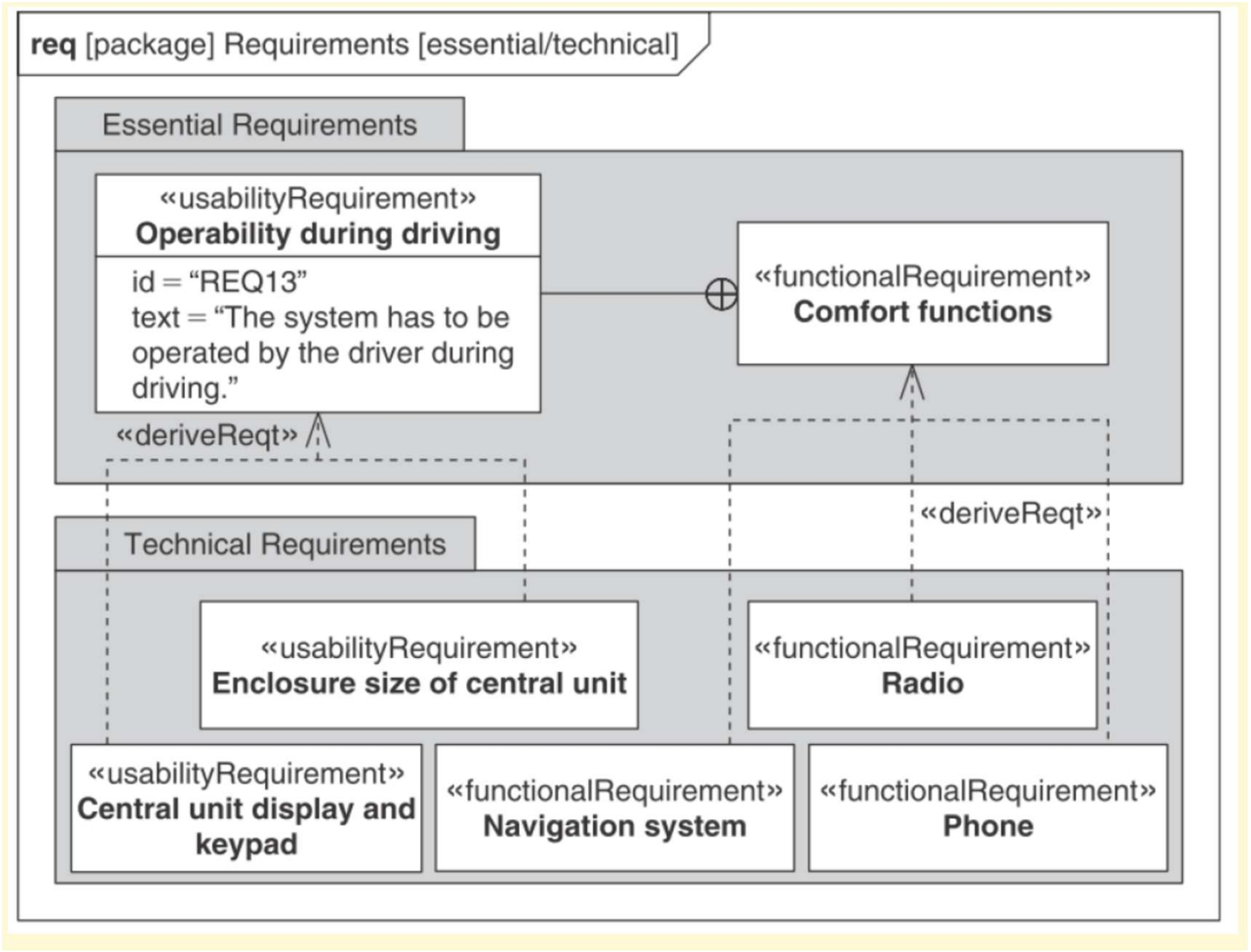


Client depends on supplier (i.e., a change in supplier results in a change in client)

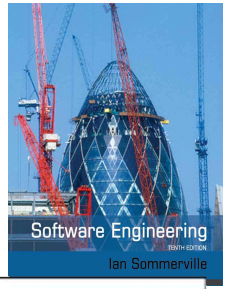
Arrow Direction Opposite Typical Requirements Flow-Down

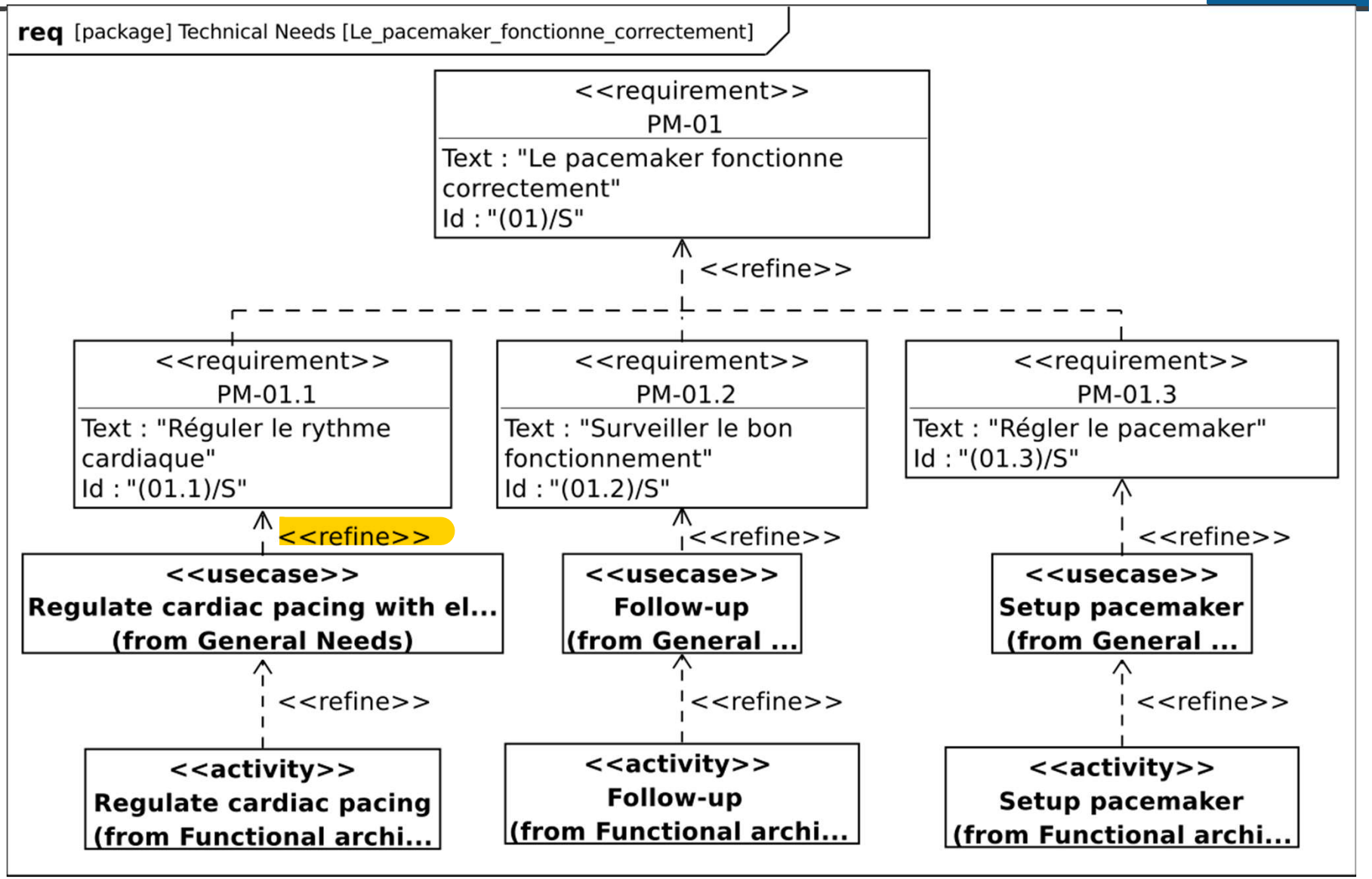
DeriveReqt relationship



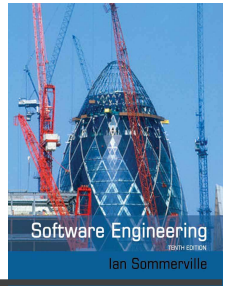


Reducing ambiguity

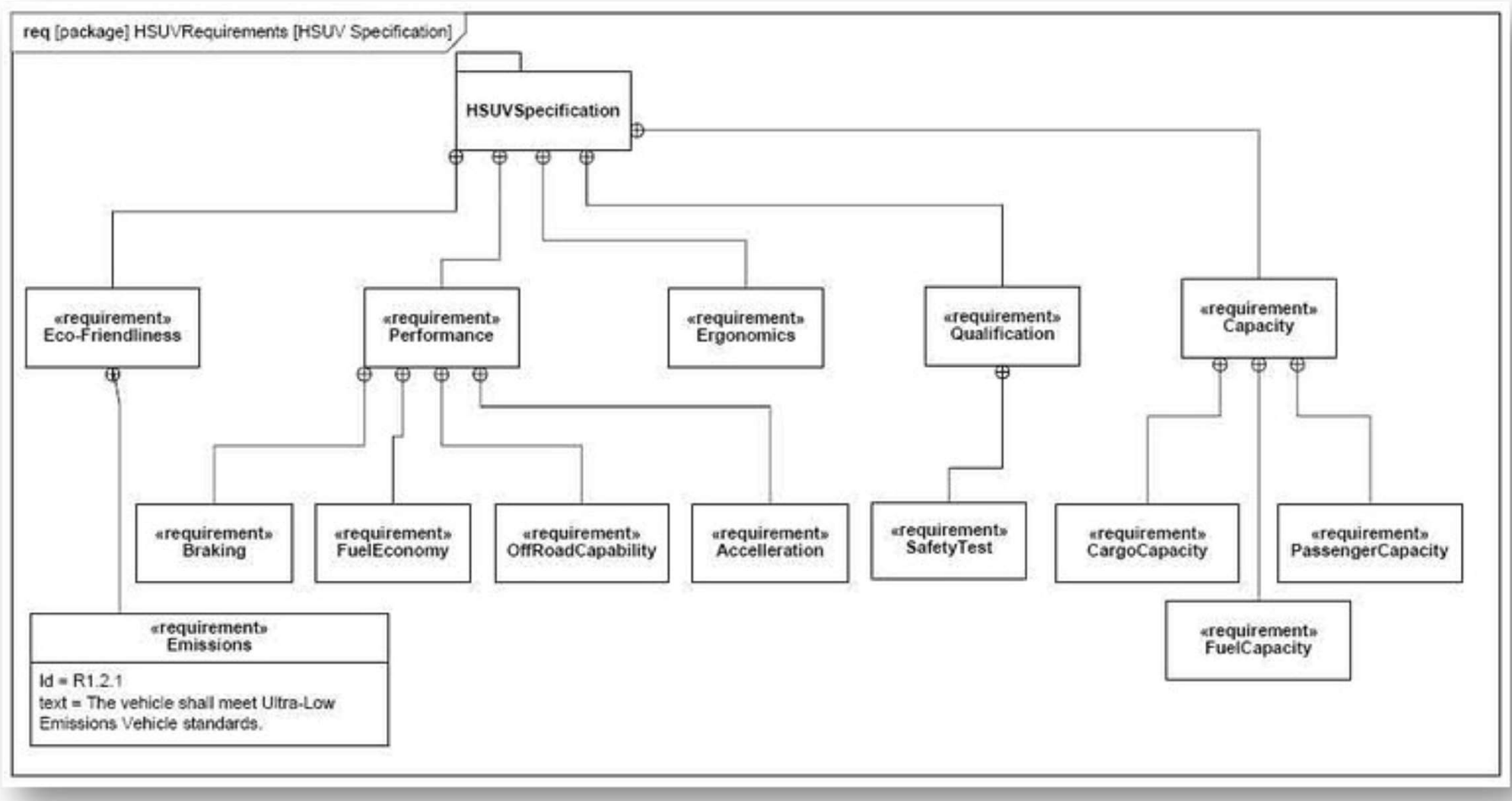




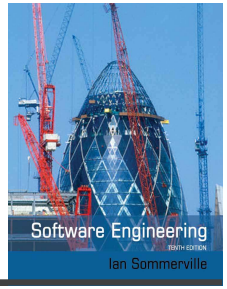
Requirements Package and Containement



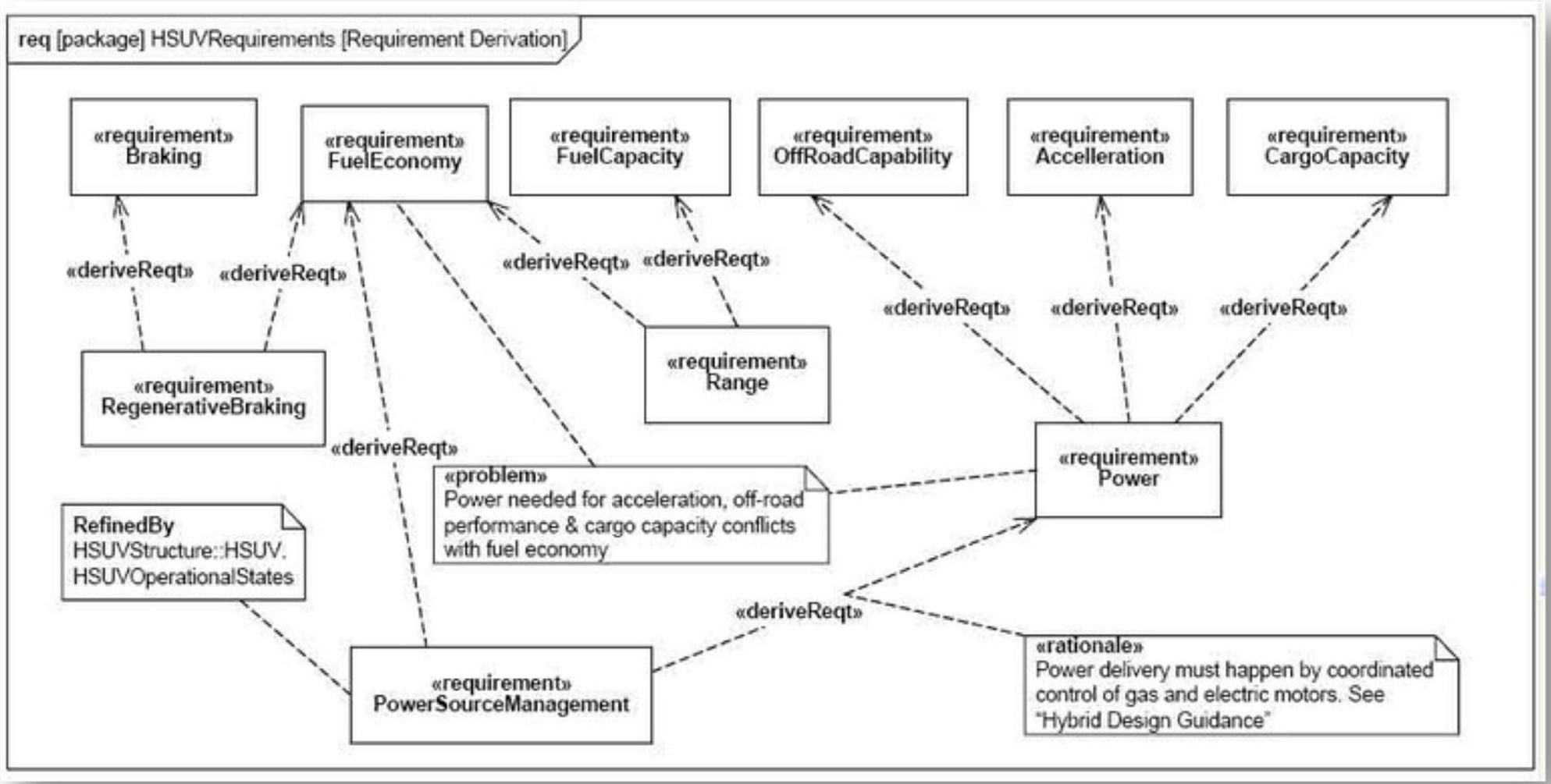
- Requirements organized into **package structure and using containement**



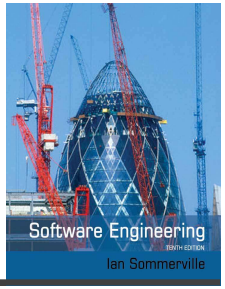
Requirement Derivation

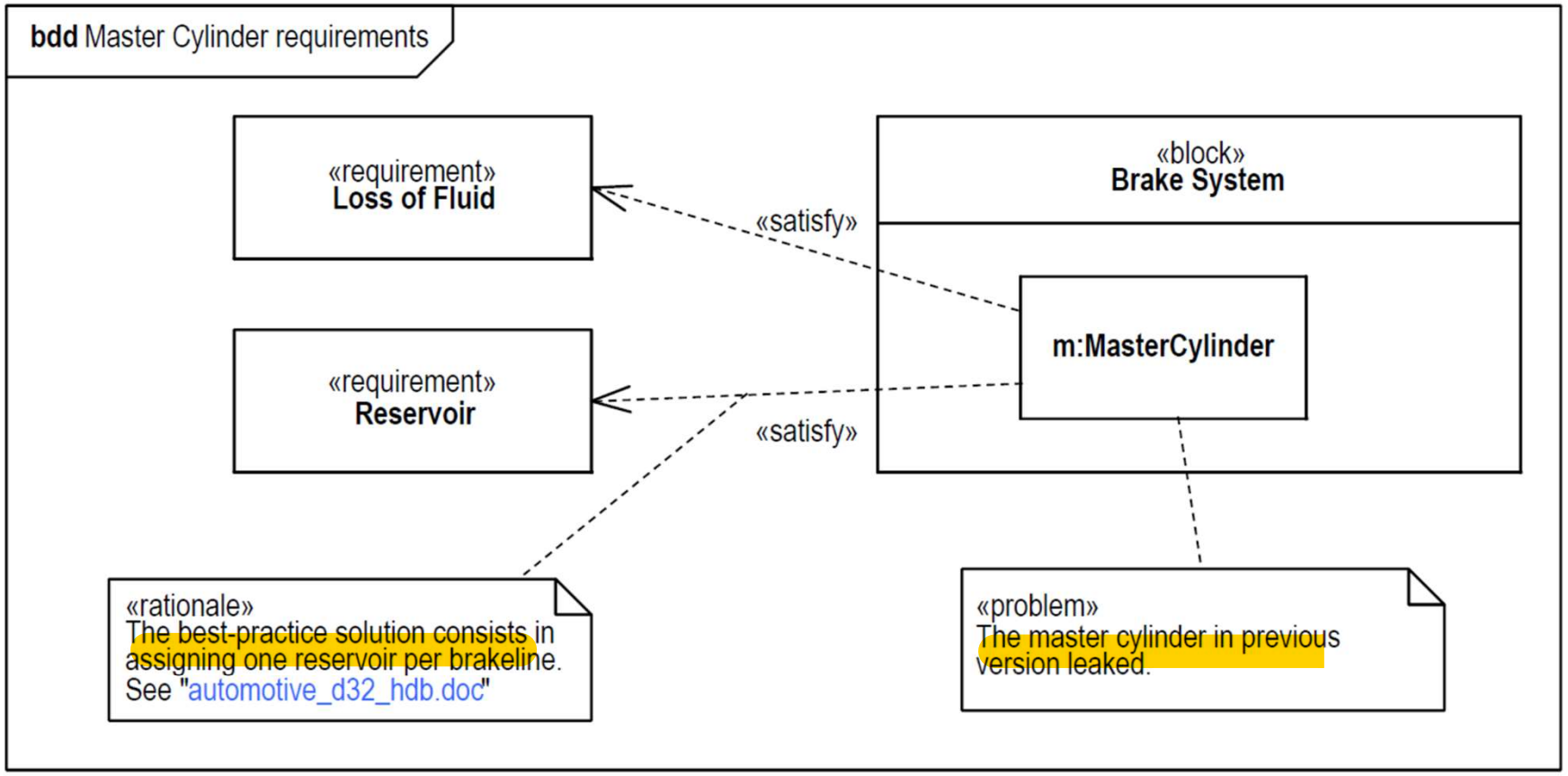


- Implies an analysis



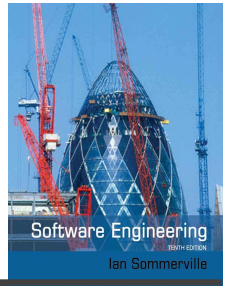
Problem and rationale

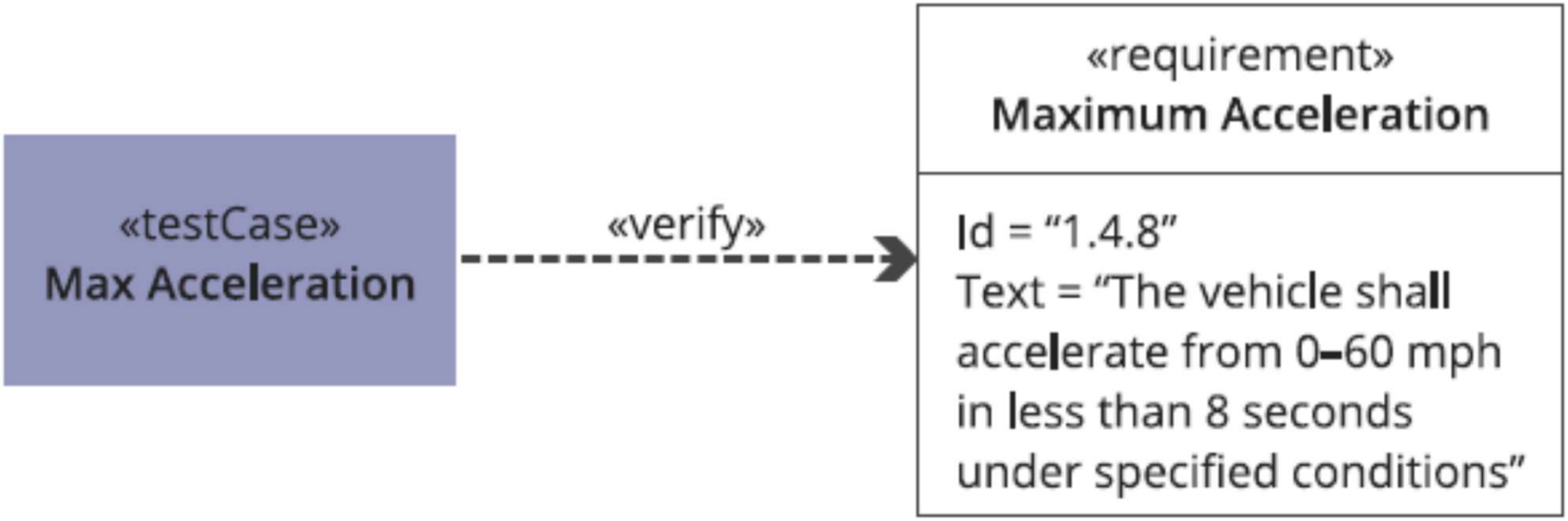




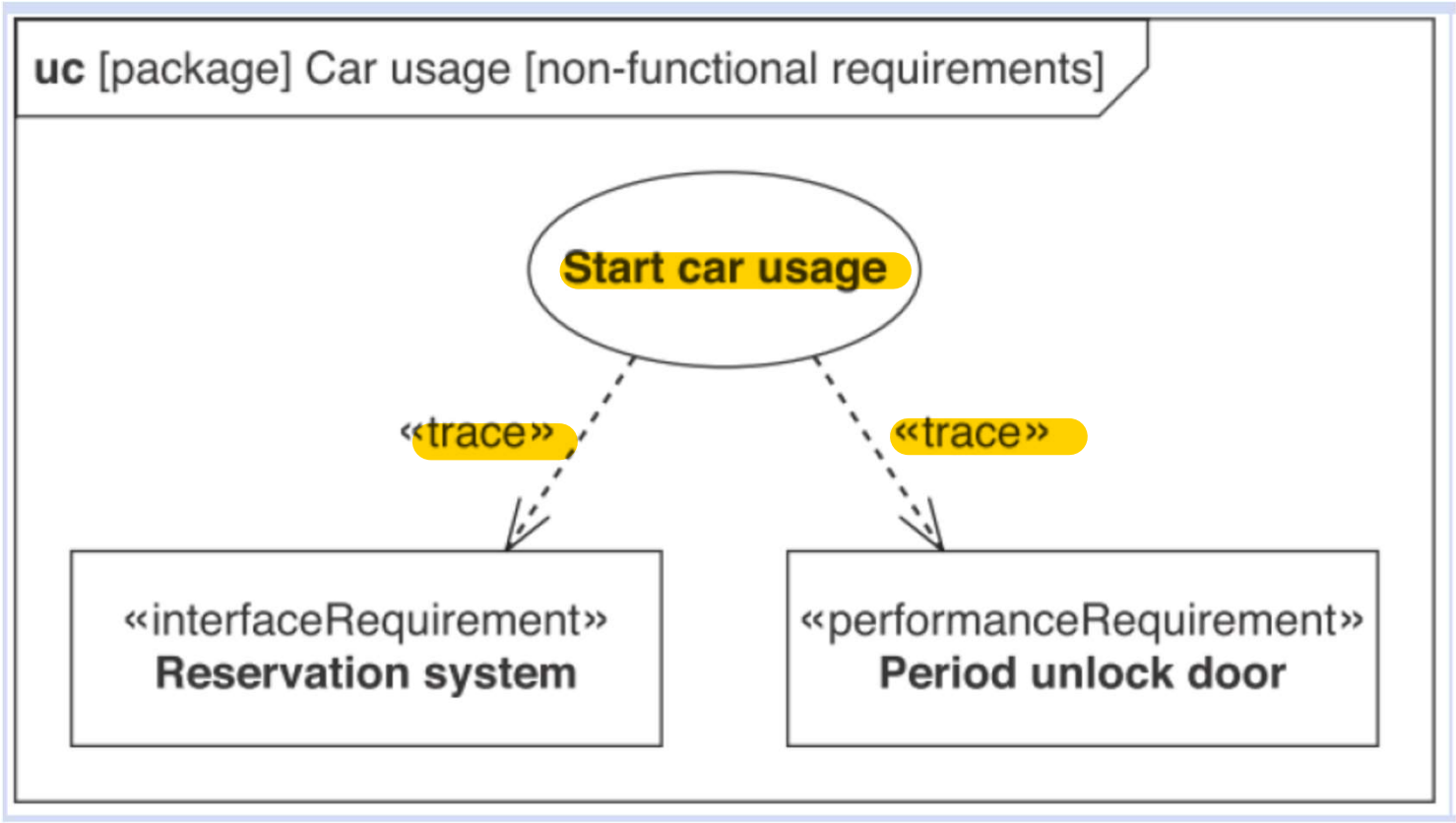
Problem and Rationale can be attached to any Model Element to Capture Issues and Decisions

Verify relationship

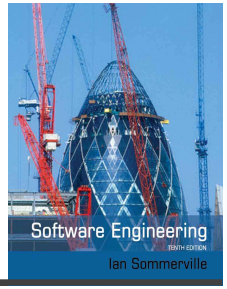




Trace relationship

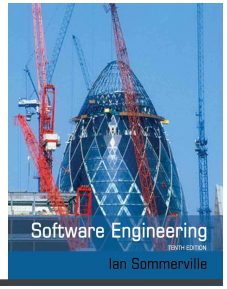


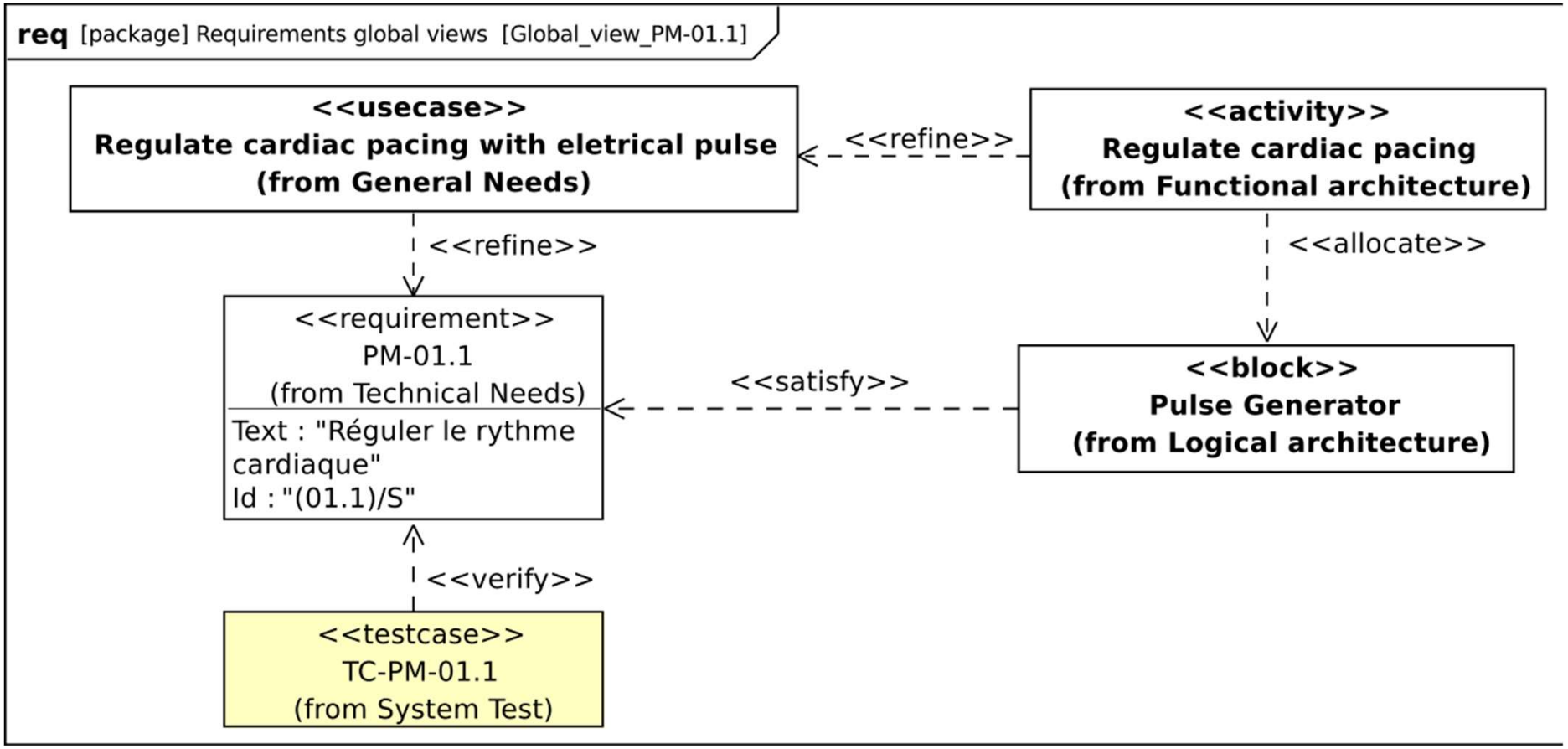
Copy relationship



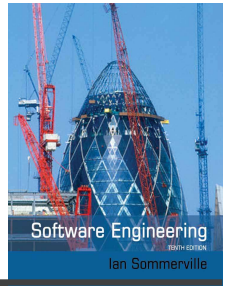
- A Copy relationship is a dependency between a slave Requirement and a master Requirement that specifies that the txt property of the slave Requirement is a read-only copy of the text of the supplier Requirement.
- Typical scenarios are regulatory, statutory or contractual requirements that are applicable across products and/or projects and requirements that are reused across product families

Tracing between requirements



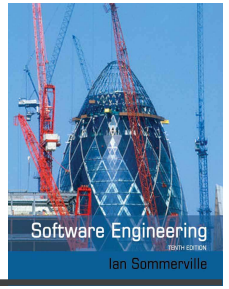


Mapping KAOS models into SysML models



- Mapping Modeling concepts
  - Goal → <<requirement>>
  - Requirement → <<requirement>> (system)
  - Expectation → <<requirement>> (user)
  - Resolutions → <<requirement>> (system or user)
  - Entity → Block
  - Operation → activity or Block operation
  - Environment Agents → Actors
  - System Agents → Blocks

Mapping KAOS models into SysML models (ctd.)



- Relationships
  - Decomposition
    - Or → multiple <<refine>>
    - And → composition
  - Concerns → <<satisfy>>
- No direct mapping
  - Obstacles
  - Conflicts

Mapping KAOS/SysML: example