SE463
Software Requirements Specification & Analysis

RE Reference Model

Readings:

Jackson, M. and Zave, P. "Deriving specifications from requirements: an example". in *Proceedings of the 17th international Conference on Software Engineering (ICSE),* 1995, pp. 15-24.

Objectives

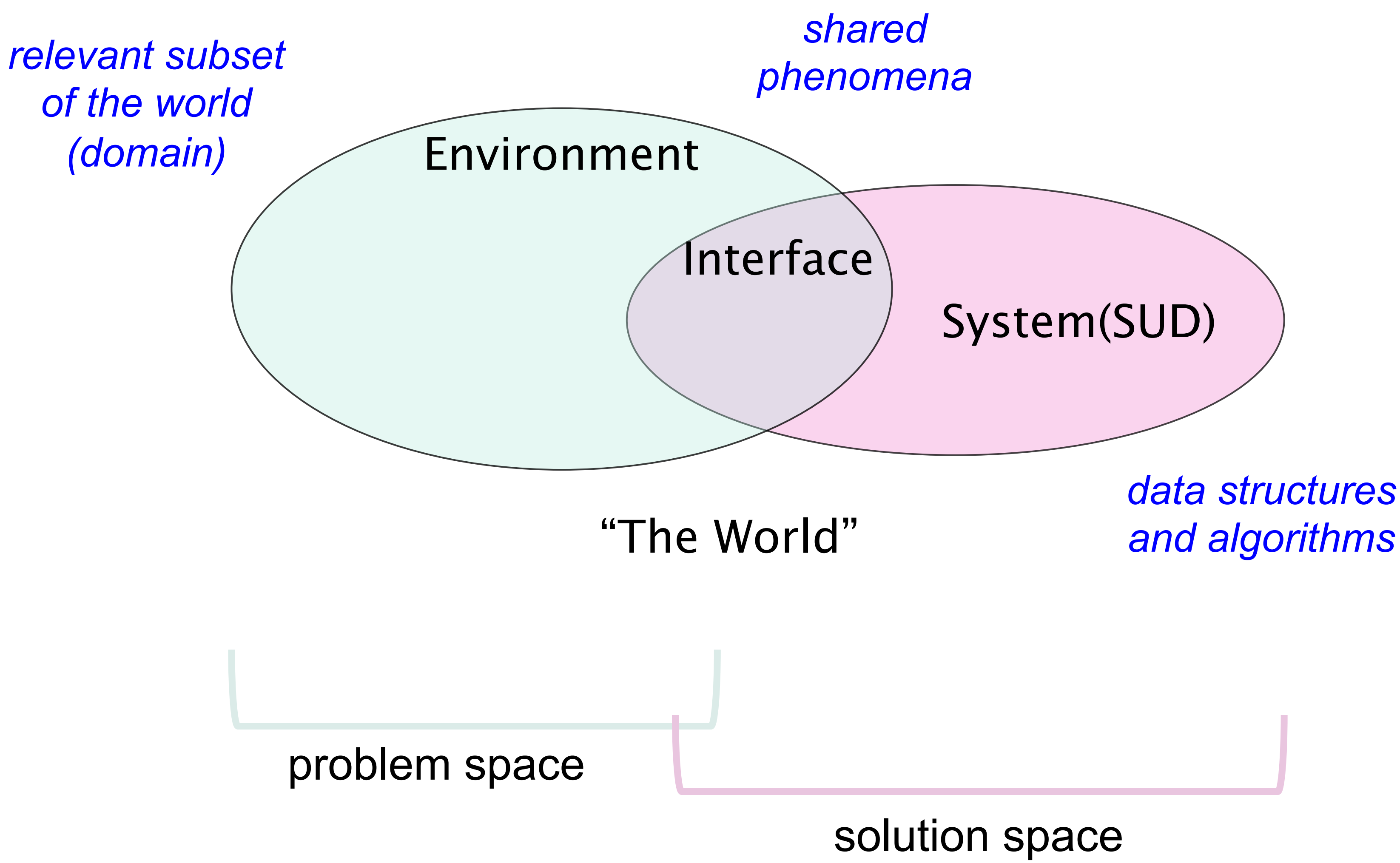
Want to identify and articulate

- Requirements - Conditions and capabilities that describe a *problem* – to be met by a *solution*, for the solution to be acceptable

- Specification - A complete, precise, verifiable expression of requirements of a *software* or *system* solution.

in a way that avoids implementation bias.

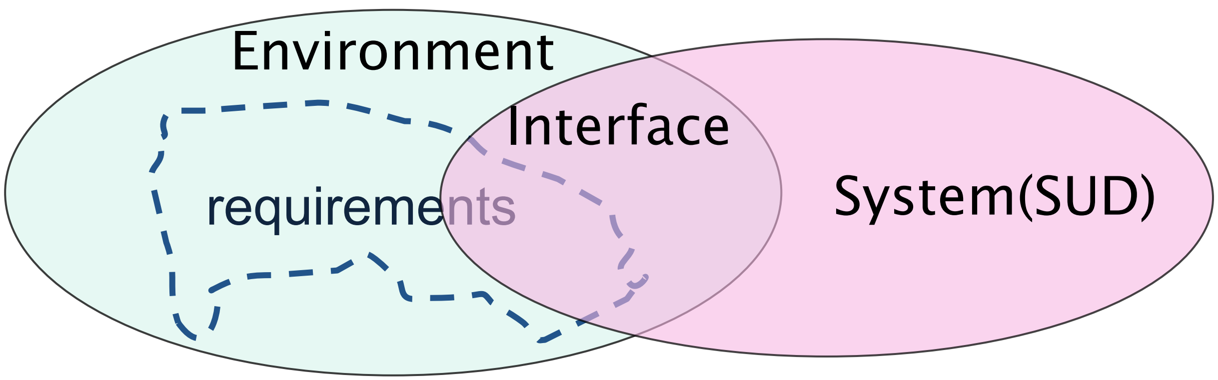
Problems vs. Solutions



Requirements

A requirement is a condition or capability that must be achieved

- desired changes to the World
- expressed in terms of environmental phenomena

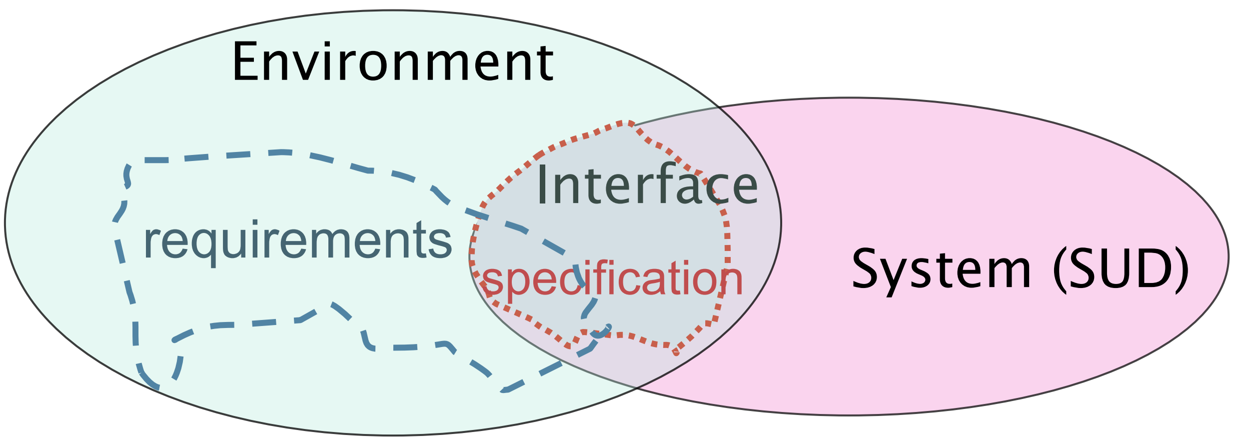


"The World"

Specification

A specification is a description of the proposed system

- desired changes to the world
- requirements re-expressed in terms of interface phenomena
- places no constraints on the design or implementation of the system giving the designer maximum freedom



"The World"

Example: Park User Fees

Suppose that the city of Waterloo decides to raise funds by instituting users fees for public parks.

Requirements:

R1: Collect $1 fee from each user on entry to the park.

R2: Ensure that anyone who has paid may enter the park.

R3: Ensure that no one may enter park without paying.

Domain Knowledge

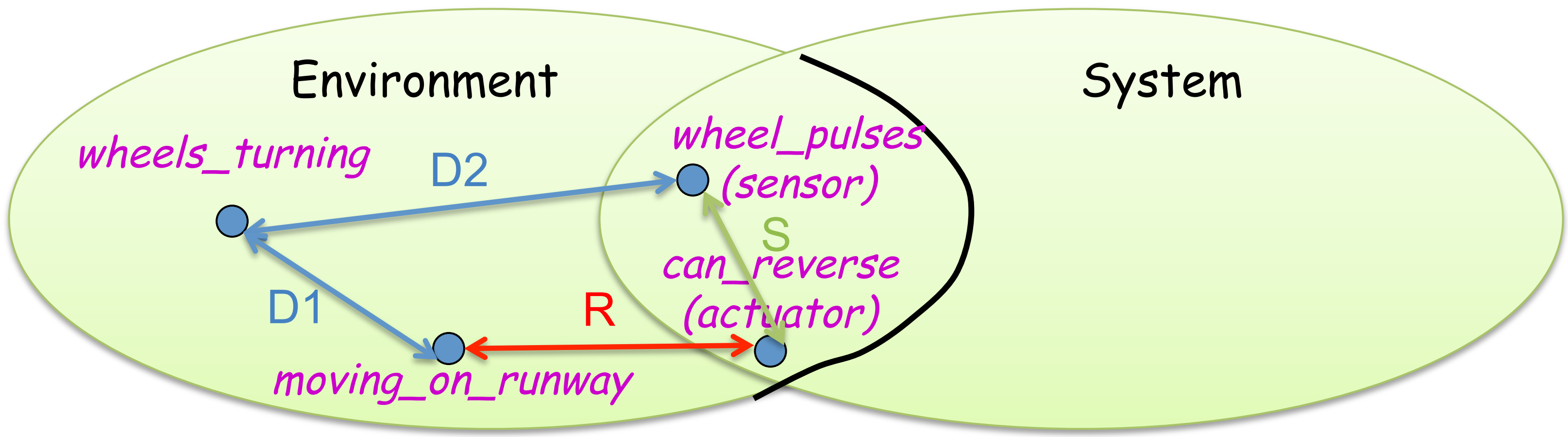
Ideally, we want to be able to show that the specifications imply the requirements:

$$\text{Spec} \models \text{Req}$$

Often we cannot do so without making some assumptions about how the environment behaves.

$$\text{Dom} \subseteq \text{Env}$$

Real World Example



R: *An airplane may engage reverse thrust iff it's moving on the runway*

D1: *Moving on runway iff wheels turning*

D2: *Wheel pulses detected iff wheels turning*

S: *Can reverse iff wheel pulses detected*

Michael Jackson, *Software Requirements and Specification*, 1995

Park User Fees Example

Requirement	Interface	Specification
Collect $1 fee from each user on entry to the park.	Coin slot	(Env) coin inserted into slot
		(Sys) senses coin
Ensure that anyone who has paid may enter the park	Barrier	(Sys) unlocks barrier upon sensing a new coin
		(Env) visitor pushes unlocked barrier
Ensure that no one may enter park without paying	Barrier	(Sys) detects entry
		(Sys) relocks barrier

RE Reference Model

The fundamental law of requirements:

$$\text{Dom, Spec} \models \text{Req}$$

Must be able to argue that the specification of the system plus the assumptions are enough to satisfy the requirements.

Deriving Specifications

For each requirement Req

- Determine how system will monitor / control the environment
  - may need to introduce interface (i.e., sensors, actuators)
  - recast Req in terms of interface phenomena (the result is a Spec)
- Determine whether Req constrains environmentally-controlled phenomena
  - if so, identify domain assumptions (Dom) that link the environmentally-controlled phenomena to system-controlled phenomena
- Check that $\text{Dom}, \text{Spec} \models \text{Req}$
  $\text{Dom} \land \text{Spec}$ is satisfiable

Example

Example #1: Thermostat

R: Want to keep the air temperature at or above the set temperature.

Example

Example #2: Traffic Light

R: Allow car traffic to cross an intersection safely, without colliding with traffic travelling in other directions.

Example

Example #3: Patient Monitor

R1: Retain records of the patient's vital signs

R2: Warn intensive-care nurse if the patient's readings exceed the safe ranges.

[Stevens, Myers, and Constantine, "Structured Design", *IBM Systems Journal*, 13(2), 1974.]

Summary

Reference model for requirements engineering

- Terminology
- Deriving specifications from requirements
- Spec, Dom $\models$ Req