CS445 / ECE451 / CS645 / SE463

Software Requirements Specification & Analysis

Specifications

Lufthansa Flight DLH 2904 (1993)

Req: reverse thrust should be enabled iff aircraft is moving on the runway

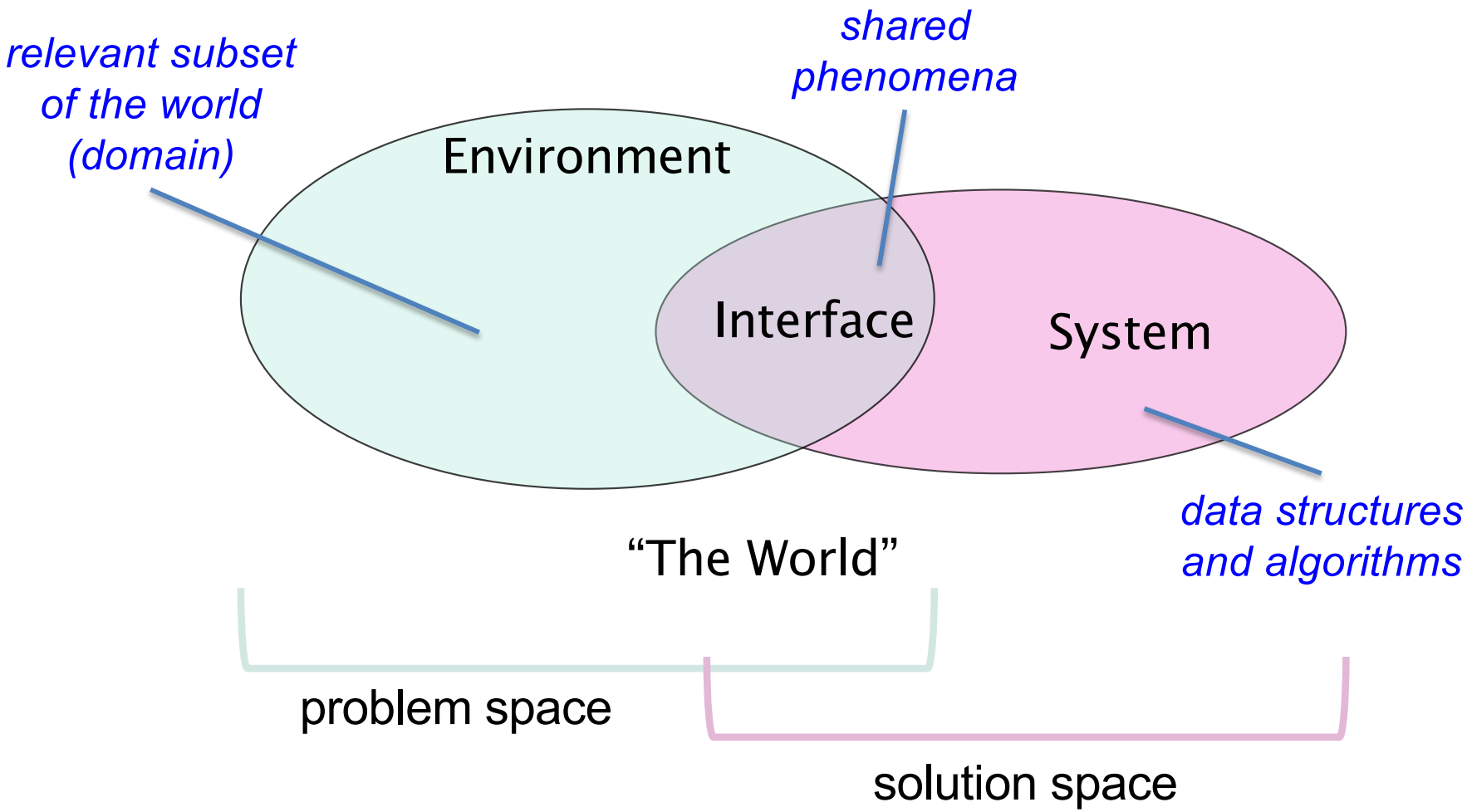
⇓

System Spec: reverse thrust enabled iff wheel pulses are detected

~~**Assume 1:** aircraft is moving on the runway iff wheels are turning~~

Assume 2: wheel sensor detects pulses iff wheels are turning

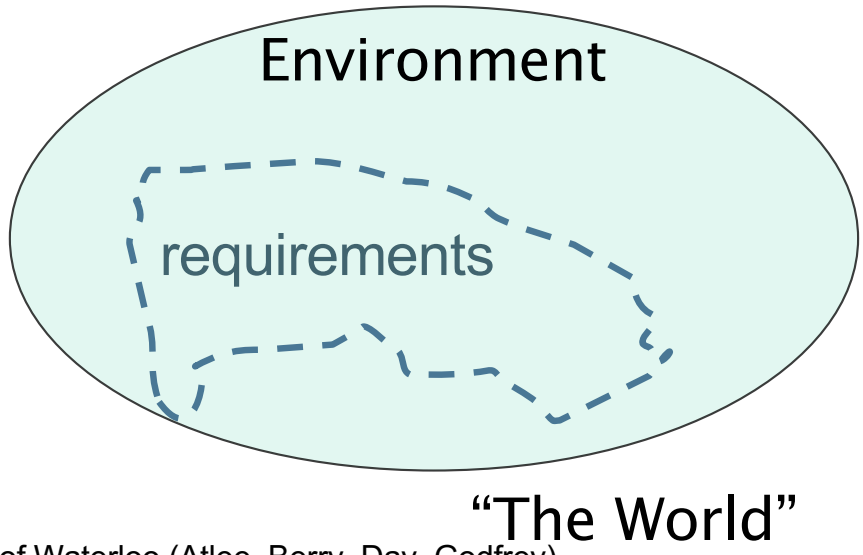
Problems vs. Solutions



Requirements

A requirement is a condition or capability that must be achieved

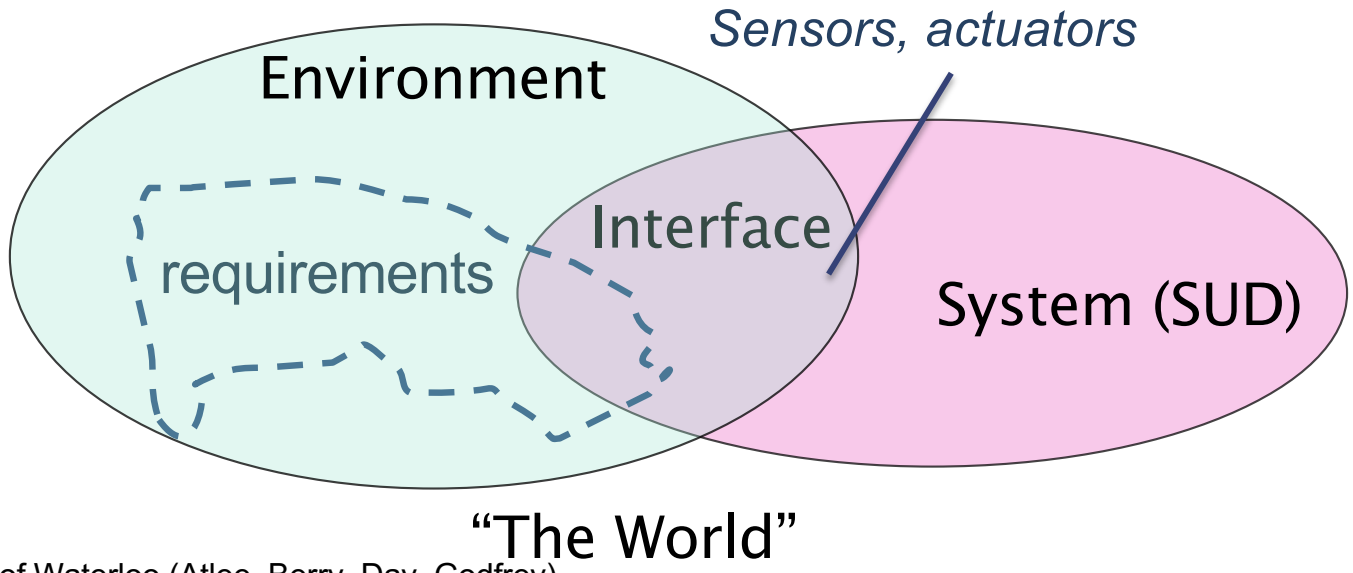
- desired changes to the World
- expressed in terms of environmental phenomena



System

The system is our proposed solution for achieving requirements

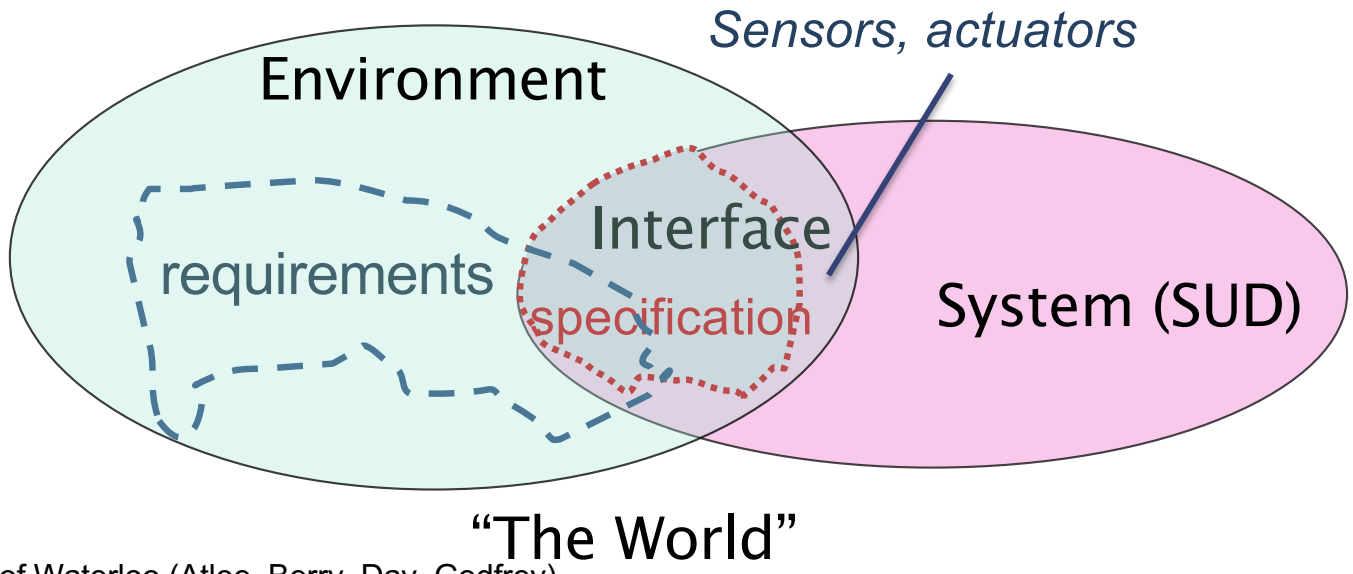
- May not be able to directly access environment phenomena
- Need to devise means by which system can access phenomena
  - Sensors to monitor environment (e.g., keyboard, mouse clicks)
  - Actuators to control environment (e.g., display screen, reports)



Specification

A specification is a description of the proposed system

- requirements re-expressed in terms of Interface Phenomena
- places no constraints on the design or implementation of the system giving the designer maximum freedom



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.

Possible Systems

- Honour system (collection box)
- Manned kiosk
- Automated system

Park User Fees Specification

Req	Interface	Specification
R1	Coin slot	(Env) coin inserted into slot
		(Sys) senses coin
R2	Turnstile	(Sys) unlocks turnstile upon sensing a new coin
		(Env) visitor can detect that turnstile is unlocked, can push turnstile
R3	Lock on Turnstile, Park fence	(Sys) detects entry
		(Sys) relocks turnstile

Assumptions

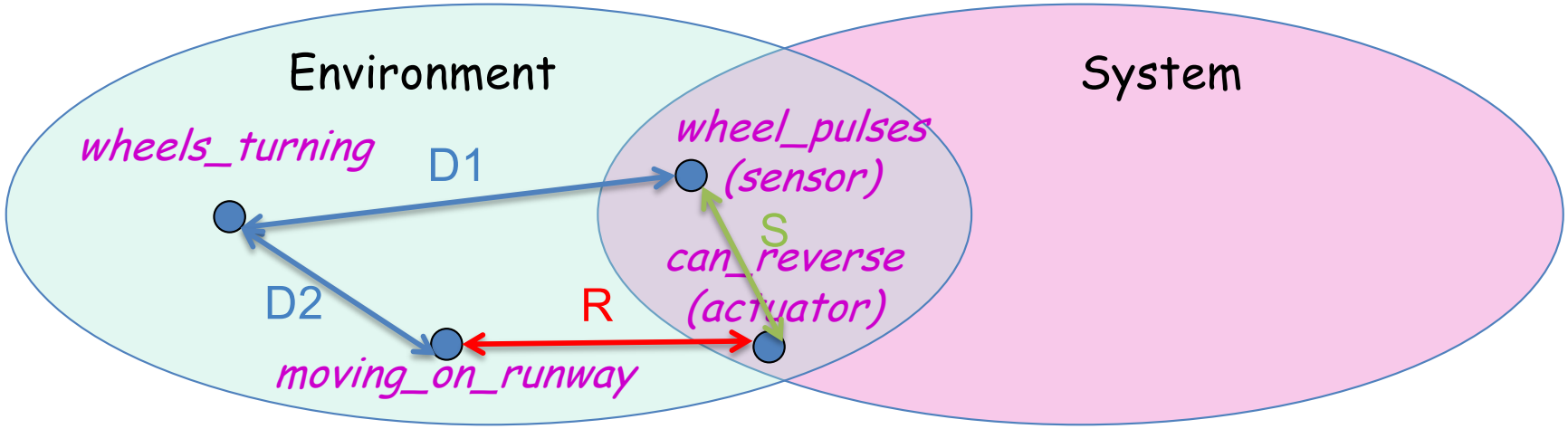
Ideally, we want to be able to show that the specifications imply the requirements:

$$Spec \models Req$$

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

$$Dom \subseteq Env$$

Lufthansa Flight DLH 2904 (1993)



R: reverse thrust is enabled iff the aircraft is moving on the runway

S: Can reverse iff wheel pulses detected

D1: wheel pulses detected iff wheels turning

D2: aircraft is moving on runway iff wheels are turning

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
Ensure that anyone	Turnstile	(Sys) unlocks turnstile upon sensing a new coin
		(Env) visit
Ensure that no one may enter park without paying	Turnstile	(Sys) detects entry
		(Sys) relocks turnstile

Asmp: Coin is worth $1

Req: #payments ≥ #entries

Asmp: Payee enters park

RE Reference Model

1) Must show that the specification of the system plus the assumptions are enough to satisfy the requirements.

$$Dom, Spec \models Req$$

2) Must show that the assumptions are not overly strong

$$Dom \wedge Spec \neq \mathit{false}$$

3) If we cannot, we have 3 options
   - Strengthen the Specification
   - Strengthen the Assumptions (and prove they are valid)
   - Weaken the Requirements (in consultation with customer)

Example

Example #1: Thermostat

R: Want to keep the air temperature at or above the user-desired temperature.

Phenomena:

air temp (sys senses through a thermometer)
user-set temp (sys senses through some UI)
furnace (sys turns on and off)

Specifications:

• if (SetTemp > Temp) then turn on furnace

Assumptions: temperature rises when the furnace is on

Example

Example #2: Traffic Light

R: Allow N/S and E/W traffic to cross a 4-way intersection without colliding

Phenomena:

Light signals to communicate when it is safe to cross the intersection (green, yellow, red signals in every direction of traffic)

Assumptions:

- Drivers stop at red lights
- Drivers stop (if they can) at yellow lights
- Drivers proceed when the light is green
- Vehicles can safely stop from 50kph in 5 sec

Specification:

- N lights are the same as S lights
- E lights are the same as W lights
- If NS lights are green or yellow, then EW lights are red
- If EW lights are green or yellow, then NS lights are red
- When NS (or EW) lights turn green, they stay green for 55 sec and then turn yellow
- When NS (or EW) lights turn yellow, they stay yellow for 5 sec (which is long enough for 3-ton vehicles to stop from 50 kph) and then turn red
- When NS (or EW) lights turn red, they stay red for 1 min and then turn green

Deriving Specifications

For each requirement Req

- Decide how the system will monitor / control the environment
  - may need to introduce interface for phenomena outside of the system (i.e., sensors, actuators)
  - may need to make Dom assumptions about how accurately the interfaces sense and control environmental phenomena
  - recast Req as system actions over interface phenomena (= Spec)

- Check that $Dom, Spec \models Req$
  $Dom \wedge Spec$ is satisfiable
  - may need to introduce additional Dom assumptions about how environmental phenomena (e.g., users, operators) behave
  - alternatively may need to strengthen Spec or relax Req

References

Michael Jackson. ”The Meaning of Requirements”, in *Annals of Software Engineering,* vol. 3, 1997, pp. 5-21.