

Requirements Models:

- ☐ The Why-What-How model
- ☐ The 4-Variable model
- ☐ The Reference model
- ☐ The Goal-Service-Constraint model
- ☐ The 4-World model

What is RE?

What are requirements?

"... Requirements definition is a careful assessment of the needs that a system is to fulfill.

It must say **why** a system is needed,
based on current and and foreseen conditions,
which may be internal operations or an external market

It must say **what** system features will serve and satisfy this context.

And it must say **how** the system is to be constructed ..."

[Ross77]

Not about the design

What is RE?

What are requirements?

why?

✿ **Enterprise requirements**

for "context analysis" - the reasons why the system is to be created.
(e.g, why IS for BPR, organizational structure, agents, goals)

constraints on the environment in which the system is to function
(e.g, airplane running beyond runway, AT&T Internet service)

the meaning of system requirements
(symbols, relationships, ontology, vocabulary)

what?

✿ **(System) functional requirements**

a description of what the system is to do;
what information needs to be maintained?
what needs to be processes?

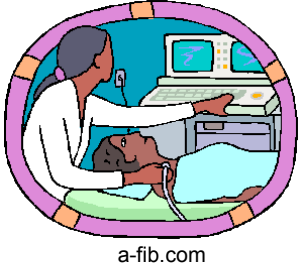
$\{f: I \rightarrow O\}$

how?

✿ **(System) non-functional requirements**

(global) constraints on how the system is to be constructed and function.
E.g., -ilities and -ities

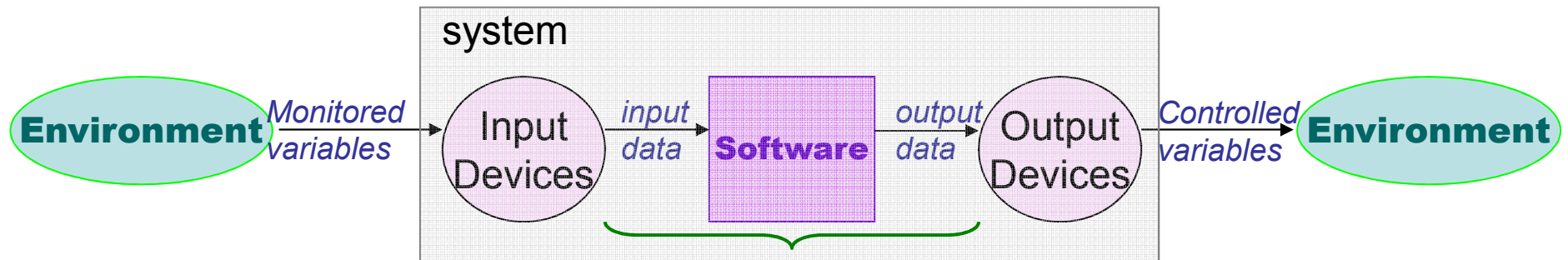
$\{bcfh(f: I \rightarrow O)\}$



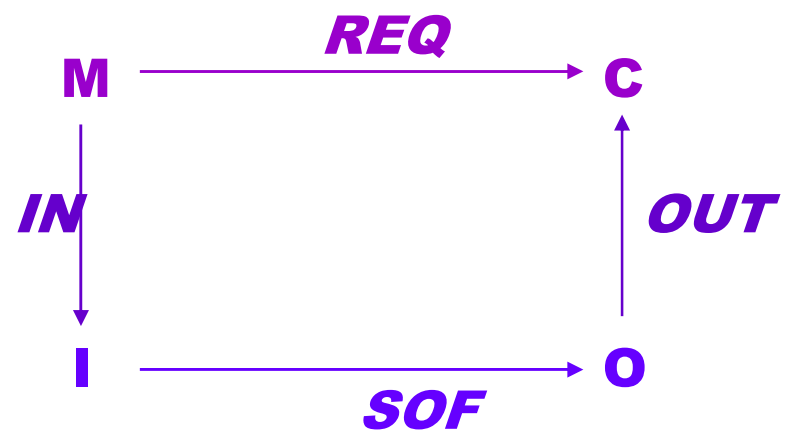
The 4-Variable Model

(the functional documentation model)

D.L. Parnas and J. Madey, "Functional Documentation for Computer Systems,"
Science of Computer Programming, Vol. 25, No. 1, Oct. 1995, pp. 41–61.



S - Specification of software in terms of inputs & outputs
(possibly large in number, and in very complex relationships)

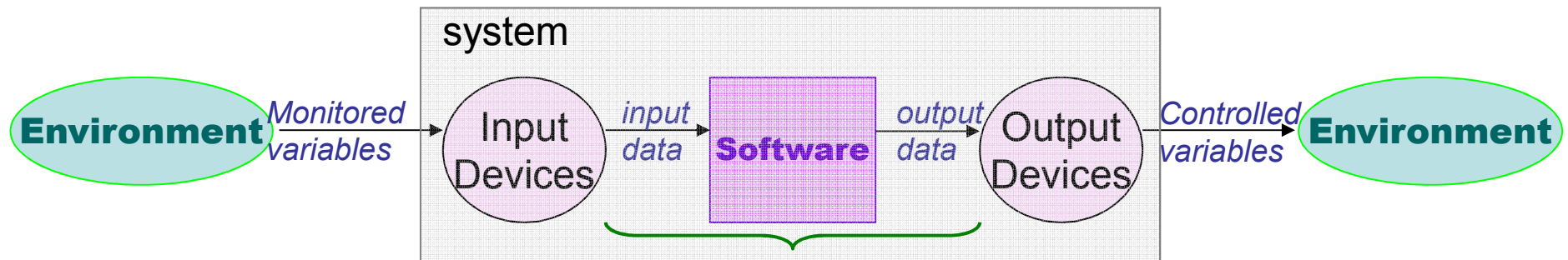




The 4-Variable Model

(the functional documentation model)

D.L. Parnas and J. Madey, "Functional Documentation for Computer Systems," Science of Computer Programming, Vol. 25, No. 1, Oct. 1995, pp. 41-61.



S - Specification of software in terms of inputs & outputs
(possibly large in number, and in very complex relationships)

NAT(m, c): describes nature without making any assumptions about the system;

REQ(m, c): describes the desired system behavior;

IN(m, i): relates the monitored real-world values to their corresponding internal representation;

OUT(o, c): relates the software-generated outputs to external system-controlled values; and

SOF(i, o): relates program inputs to program outputs.

Any issues?

Nat - the range of sounds detected or non-detected by the sensor and the possible range of values of the actuator controlling the buzzer.

Req - A warning that notifies the nurse if the system detects heart stops beating. The document' formalization: if the sound being monitored falls below a certain threshold, then the system sound the buzzer.

In - The input registers holding the data read from the sensor monitoring the sounds of the heart beat.

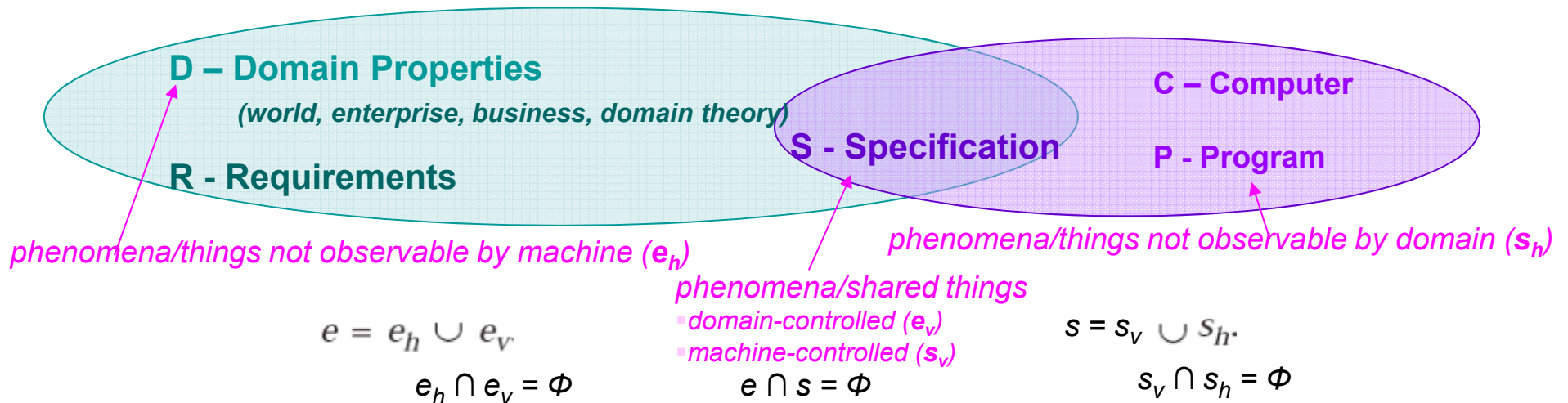
Out - The output registers which are read by the actuator that can sound the buzzer.

SOF - If the input register doesn't show signs of a heart beat for more than some specified time then the output register indicates the alarm to ring.

What are requirements?

[P. Zave and M. Jackson, Four Dark Corners of Requirements Engineering. *ACM Transactions on Software Engineering and Methodology* 6(1) 1-30. ACM Press. 1997]
 [C. Gunter, E. Gunter, M. Jackson and P. Zave, "A Reference Model for Requirements and Specifications," *IEEE Software*, May/June 2000. pp. 37-43]

The WRSPM Model



□ Domain Properties: (indicative, = assumptions=domain knowledge)

things in the environment (application domain) that are true regardless of the proposed system

□ Requirements: (optative)

things in the application domain that we wish to be made true through the proposed system

"Many phenomena not accessible by the machine"

□ Specification:

a description of the behaviors that the program must have in order to meet the requirements

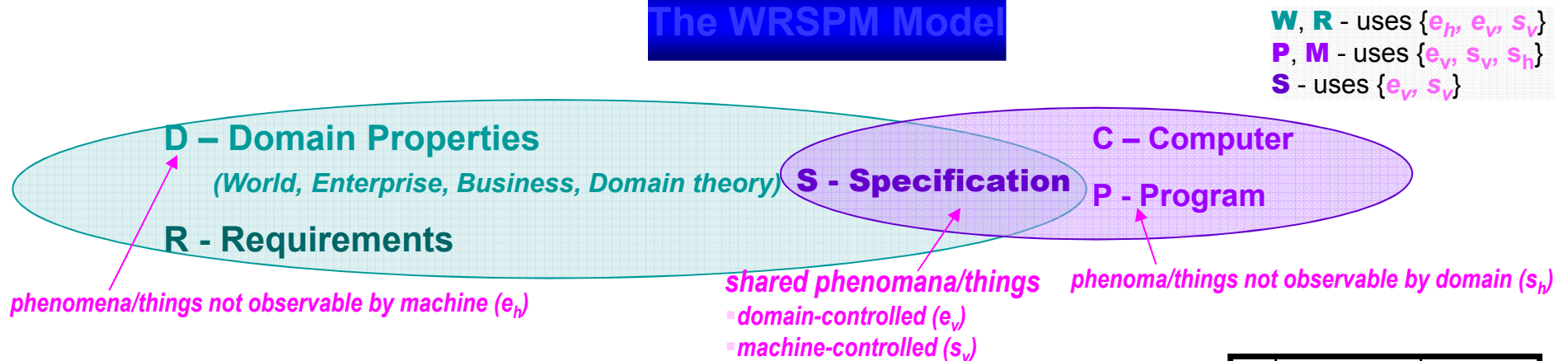
"Can, and should, only be written in terms of shared phenomena"

❖ **Designated Terminology** – names/vocabulary to describe W, (R), S, M in terms of phenoma – typically states or events

Requirements should contain *nothing but* information about the environment.

[P. Zave and M. Jackson, Four Dark Corners of Requirements Engineering. *ACM Transactions on Software Engineering and Methodology* 6(1) 1-30. ACM Press. 1997]

The WRSPM Model



	indicative	optative
e_h	✓	✓
e_v	✓	✓
s_v	✓	✓

- **R** requirements describe what is observable at the interface between the environment and the machine – hence exist only in the **e**nvironment;
- Anything else is regarded as **i**mplementation bias;
- States in **S**pecifications should describe states of the **e**nvironment –
hence, specification languages intended to describe internal (program) states of the machine are inadequate.

Verification: $S, D \models R$

$P, C \models S$

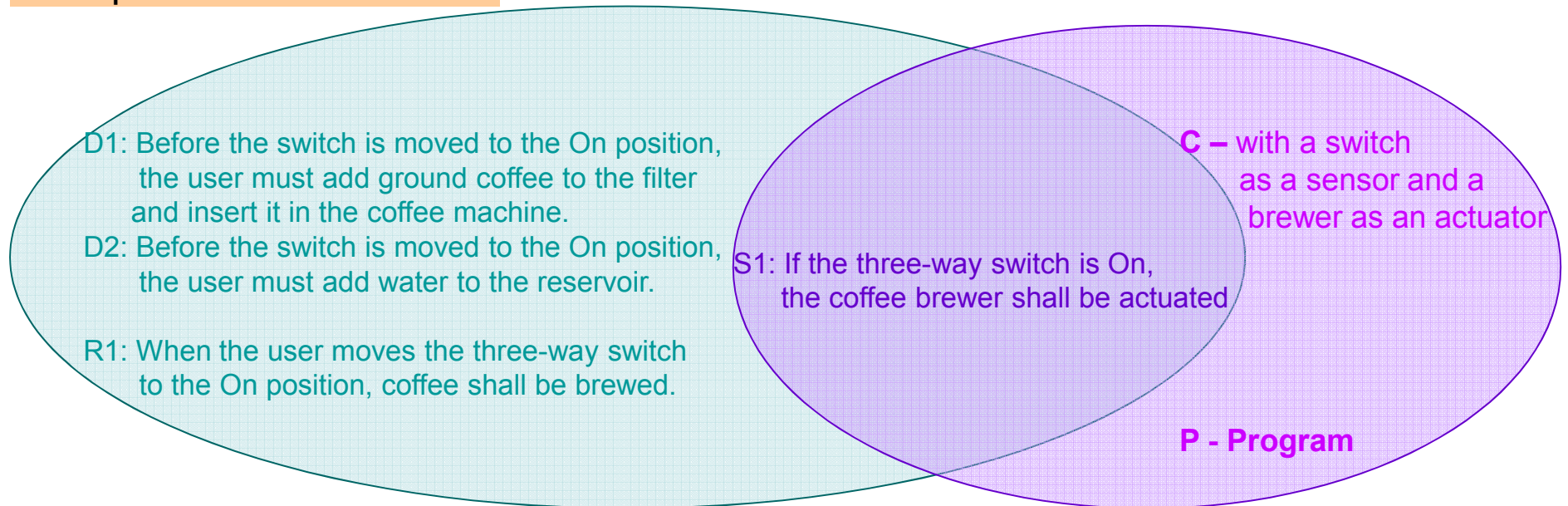
Consequences

- Freedom to collect and record information about the **e**nvironment even before we are sure it will be needed
(i.e., no minimality restriction - there must be nothing that is not necessary to carry out the currently proposed machine functions.);
- Designations refer to the real **w**orld, and **m**achine states may have NO direct correspondence to it



What are requirements?

Example 6: Coffee Machine



Designation Categories:

e_h : ?

e_v : ?

s_v : ?

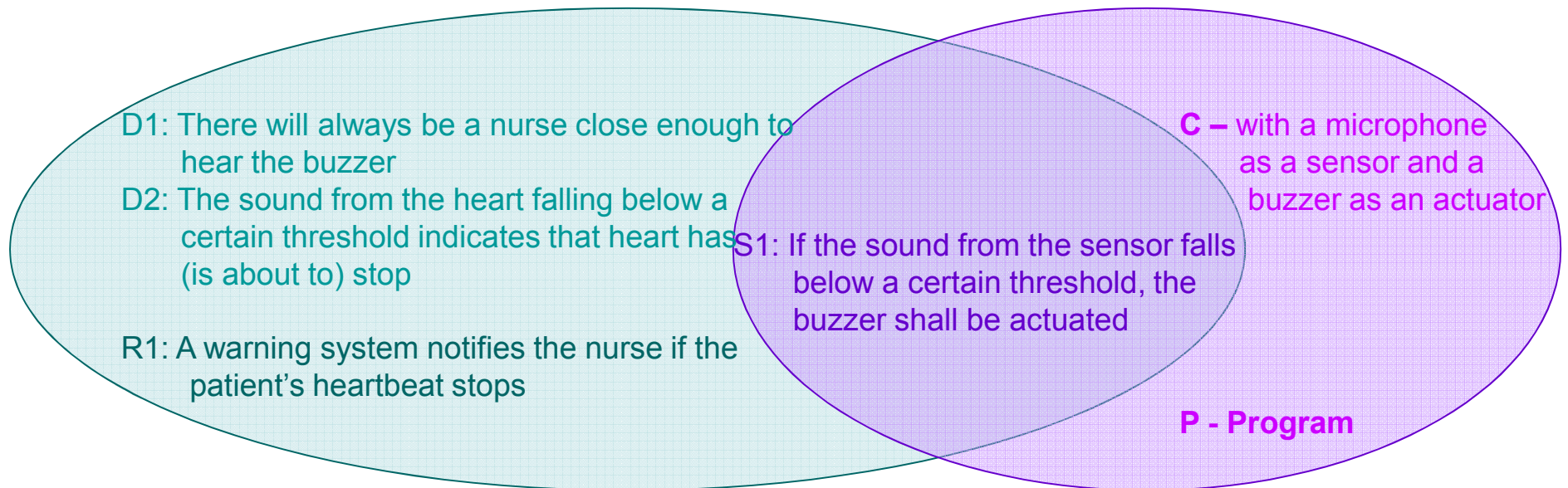
s_h : ?



What are requirements?

[C. Gunter, E. Gunter, M. Jackson and P. Zave, "A Reference Model for Requirements and Specifications," IEEE Software, May/June 2000. pp. 37-43]

Example 1: Patient Monitoring



Designation Categories:

- e_h : the nurse and the heartbeat of the patient.
- e_v : sounds from the patient's chest.
- s_v : the buzzer at the nurse's station.
- s_h : internal representation of data from the sensor.

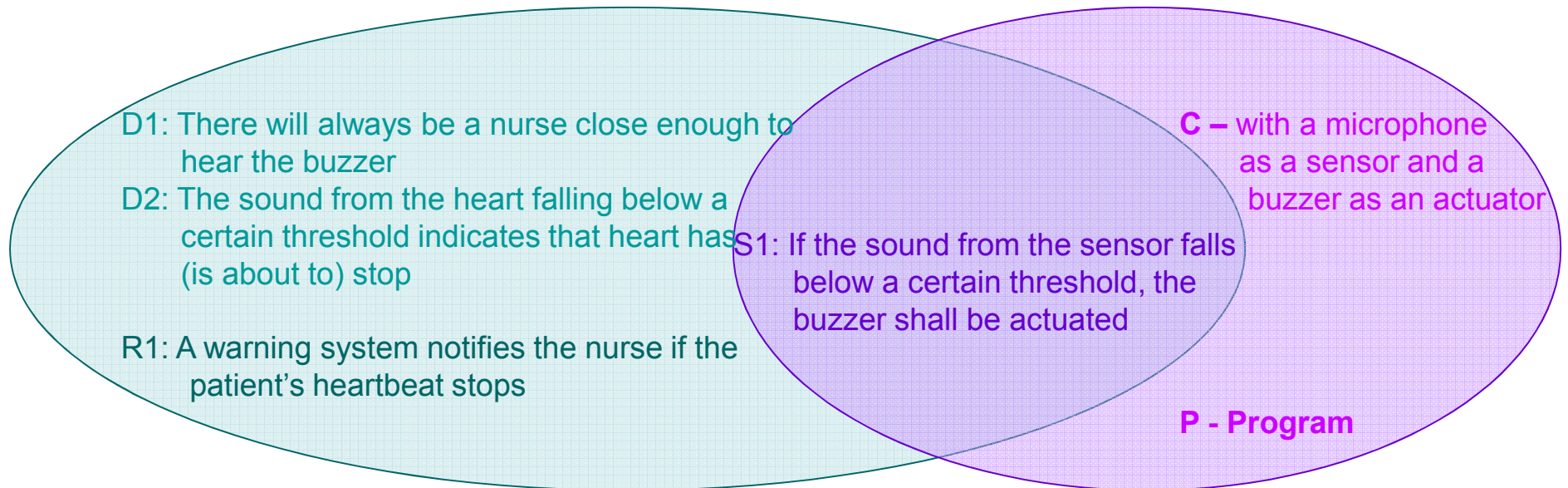
What if the domain assumptions are wrong?



What are requirements?

[C. Gunter, E. Gunter, M. Jackson and P. Zave, "A Reference Model for Requirements and Specifications," IEEE Software, May/June 2000. pp. 37-43]

Example 1: Patient Monitoring



What?

$S, D \models R$

$P, C \models S$

What is RE?

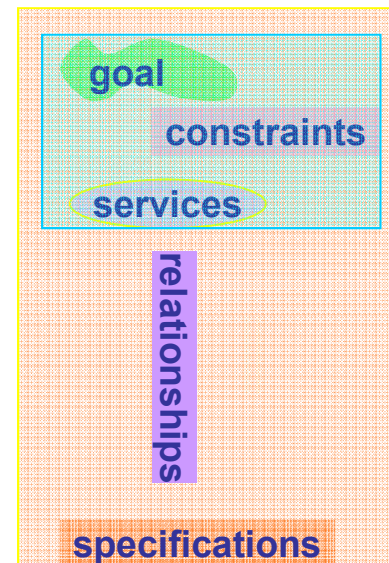
"... Requirements Engineering is the branch of Systems engineering concerned with

real-world goals for,
services provided by, and
constraints on
software systems

Requirements engineering is also concerned with
the relationships of these factors

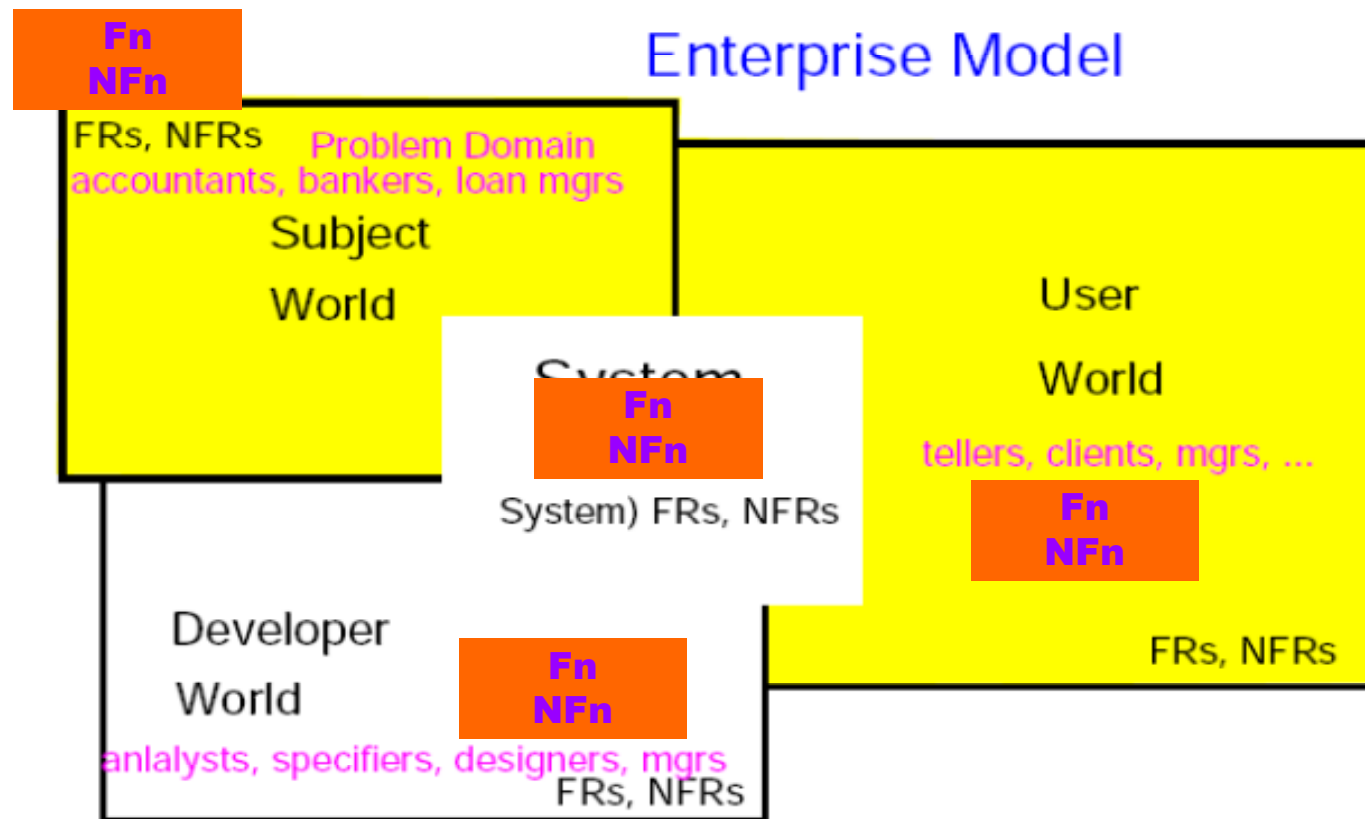
to precise specifications of system behavior
and
to their evolution over time and across system families..."

[Zave94]



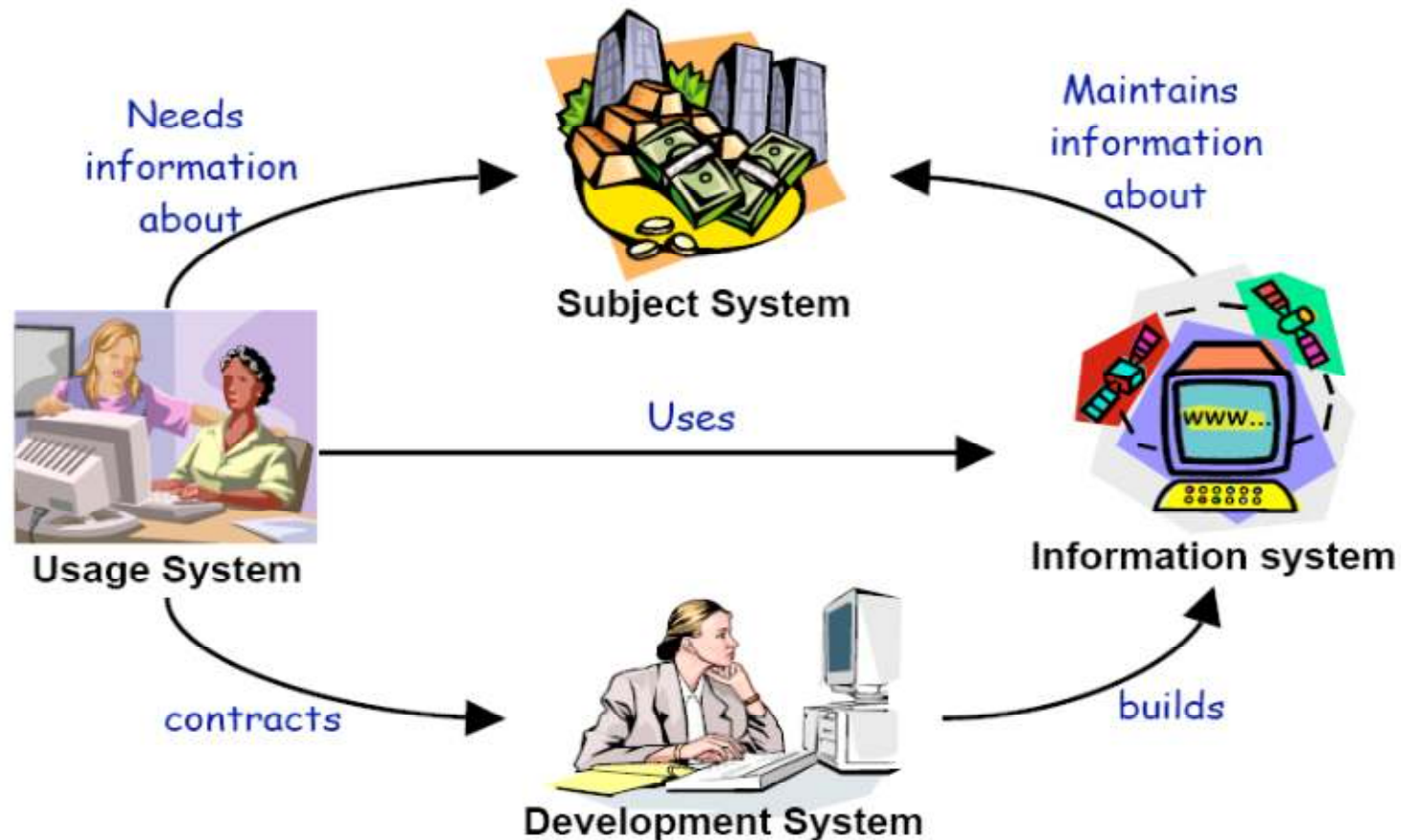
What to elicit?

Four Worlds of RE



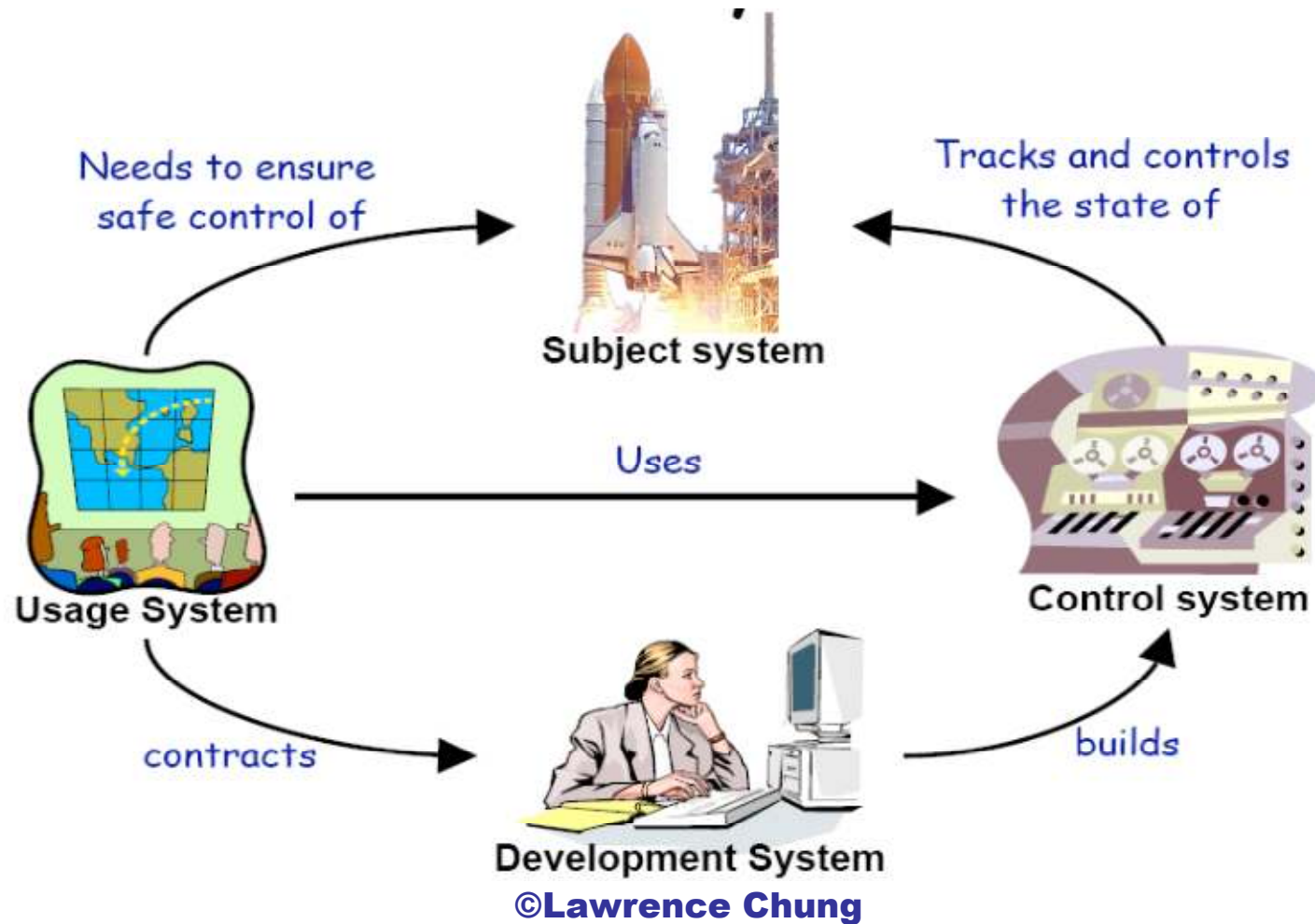
Four Worlds of RE for Information Systems

Adapted from [LK1995, p73] [S. Easterbrook, 2000-20



How does this relate to RE process?

Four Worlds of RE for Control Systems



Recall: Models about Requirements Revisited

The Why-What-How Model

The WRSPM Model

Requirements should contain *nothing but* information about the environment.

W, R - uses $\{e_v, e_v, s_v\}$
 P, M - uses $\{e_v, s_v, s_v\}$
 S - uses $\{e_v, s_v\}$

D - Domain Properties

(World, Enterprise, Business, Domain theory)

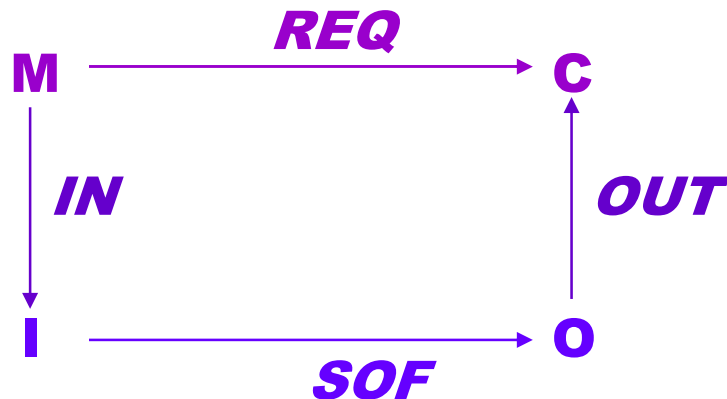
R - Requirements

S - Specification

C - Computer

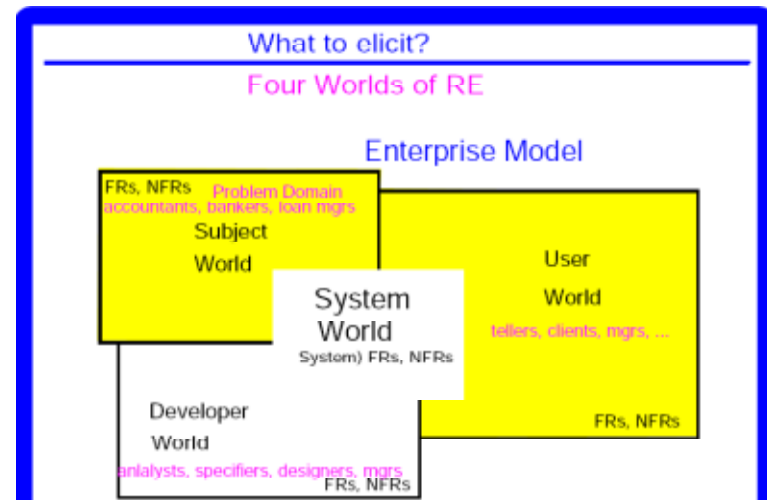
P - Program

▪ The 4-variable model:



▪ The goal-service-constraint model:

©Lawrence



Does the reference model capture all the above?

Where are goals, services and constraints? [Zave94]

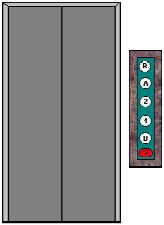
Which is about S, D \models R?

Which is about technical feasibility, component reuse, etc.?

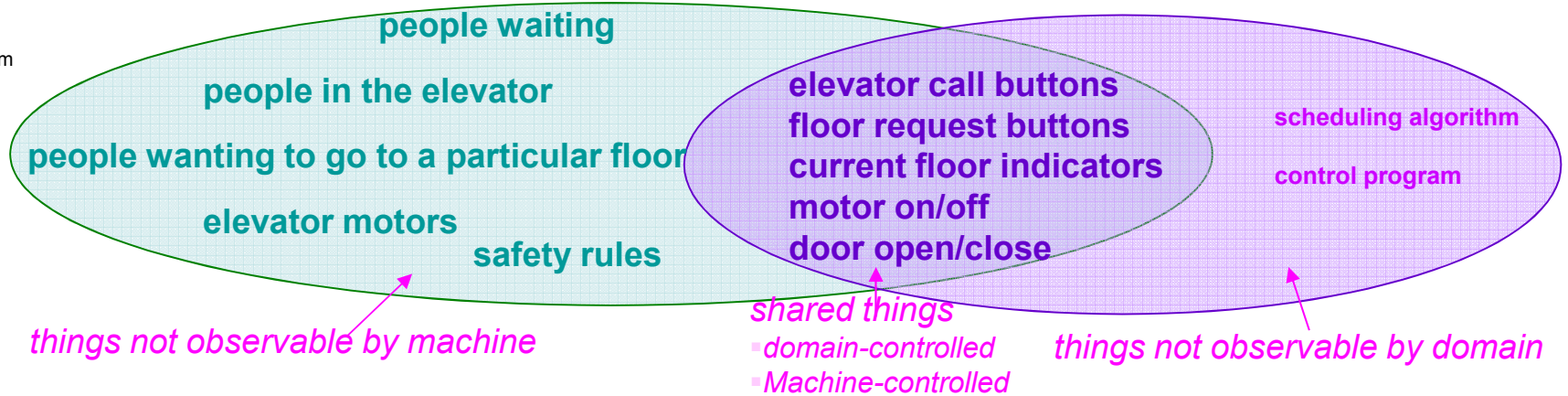
Where is traceability?

Boundaries are not fixed

Example 1: Elevator control system



www.geocities.com



E.g. Add some sensors to detect when people are waiting
This changes the nature of the problem to be solved

An Integrated Model

-Ross:

- why (context – Environment/Enterprise, Domain, Business, World)
- what (Fn)
- how (NFn)

- Reference model: W/D. R. S

- Zave: Goal

	Fn ^G	NFn ^G
E/D/B/W		
R		
S		

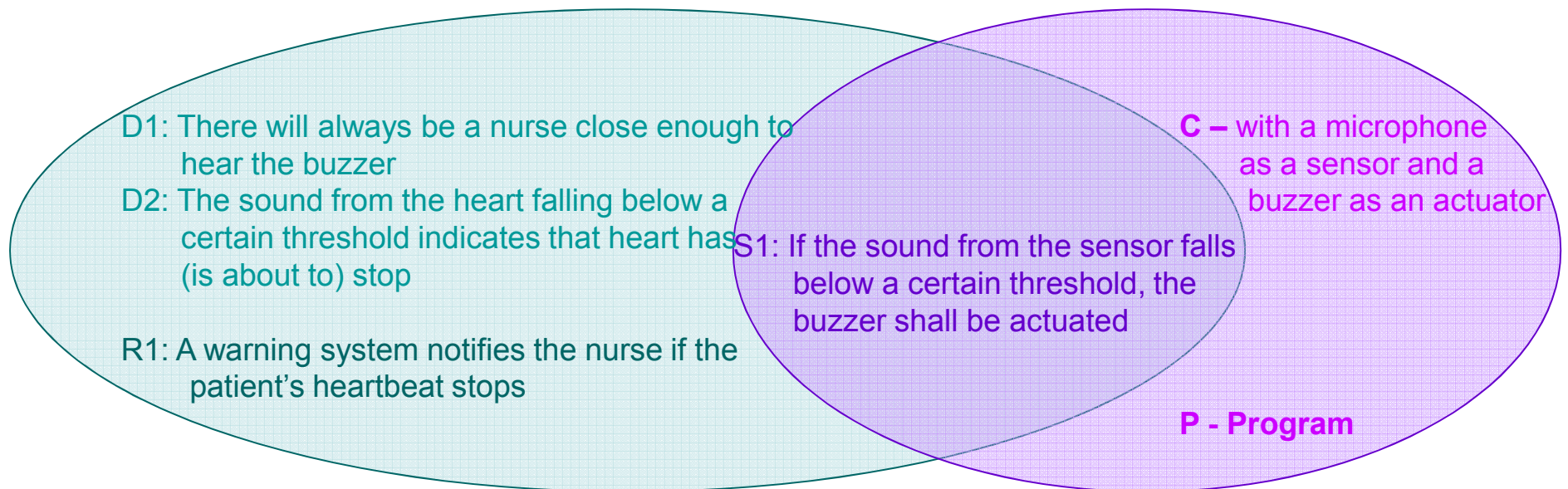
traditional emphasis here or in S + D

More models & Goal-orientation – these later in Elicitation & Modeling)

Some Questions

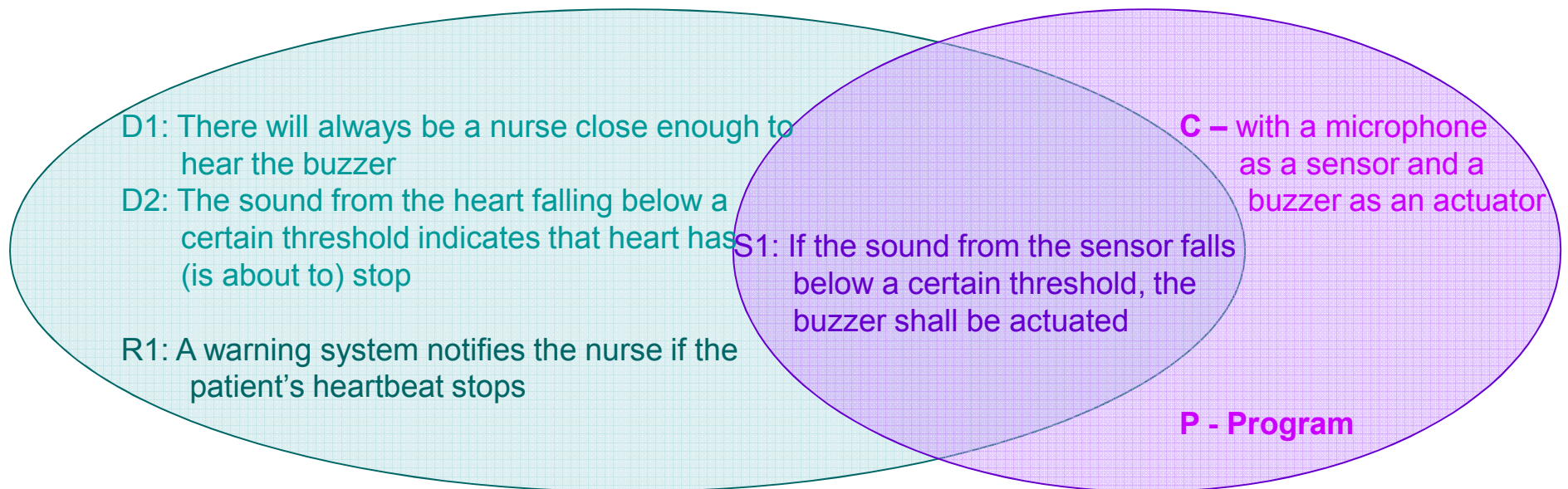
Revisiting the Reference Model

Any issues with this requirement?



Revisiting the Reference Model

Any issues with this requirement?

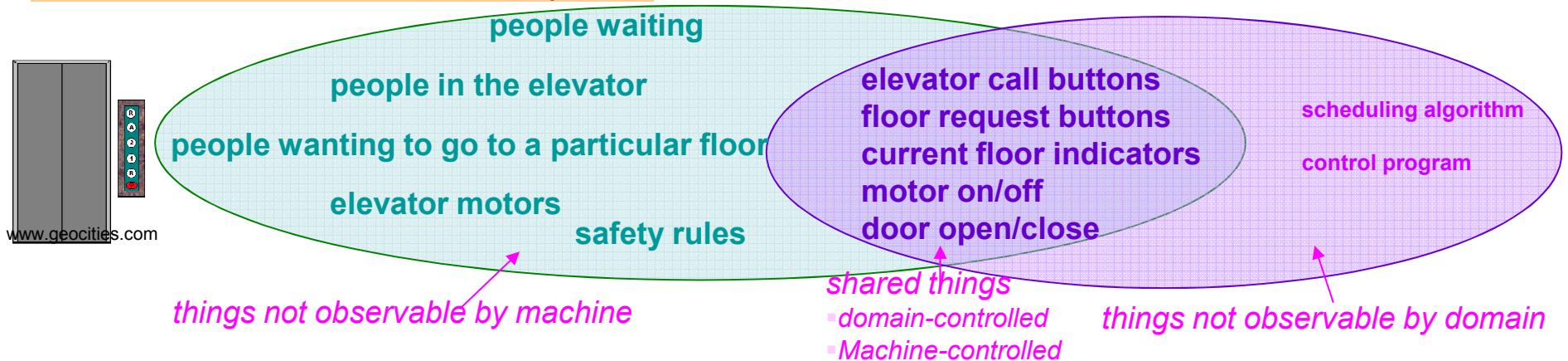


What if a warning system notifies the nurse one year after the patient's heartbeat stops?

Is this warning system really what we want?

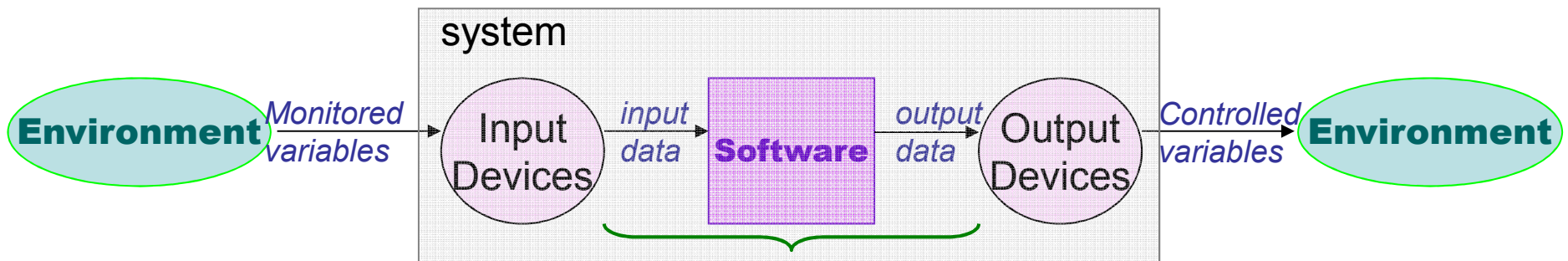
Boundaries are not fixed

Example 1: Elevator control system



E.g. Add some sensors to detect when people are waiting
This changes the nature of the problem to be solved

Example 2: The 4-variable model



S - Specification of software in terms of inputs & outputs
(possibly large in number, and in very complex relationships)

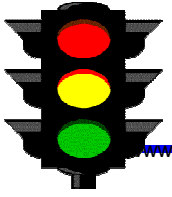
Systems engineer decides - what application domain phenomena are shared
- the boundaries by designing the input/output devices
- I/O data as proxies for the monitored and controlled variables

Boundaries are not fixed

- **Consider coffee brewing machine**
- **Consider patient monitoring system**



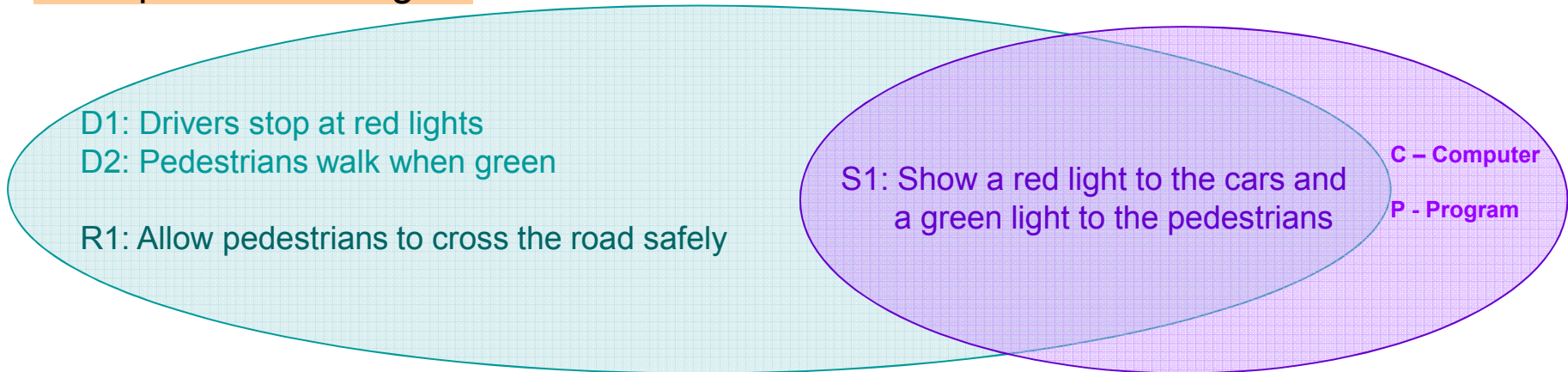
Appendix



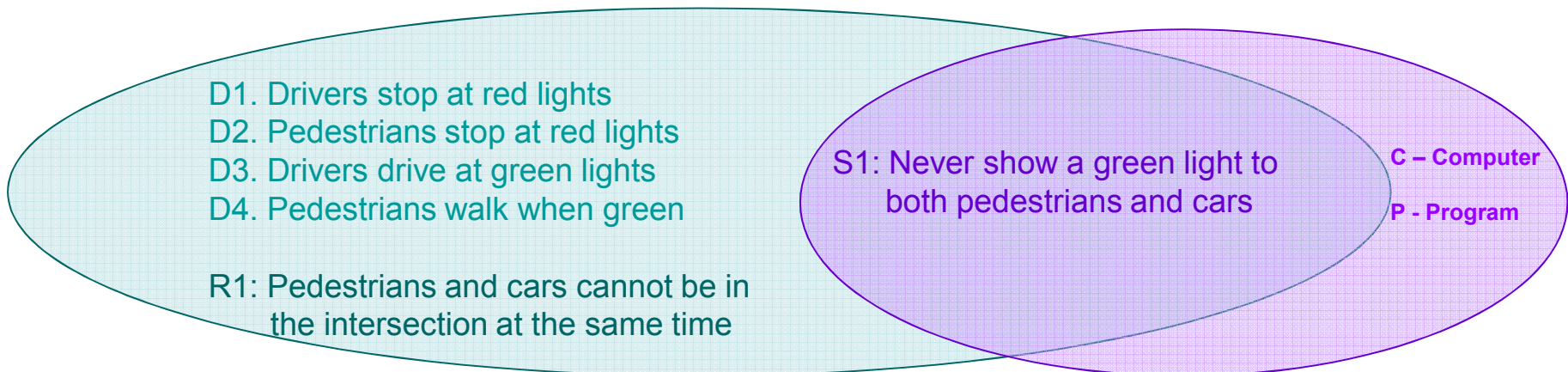
www.sergolf.com

What are requirements?

Example 2: Traffic lights



Example 3: Traffic Lights - Safety

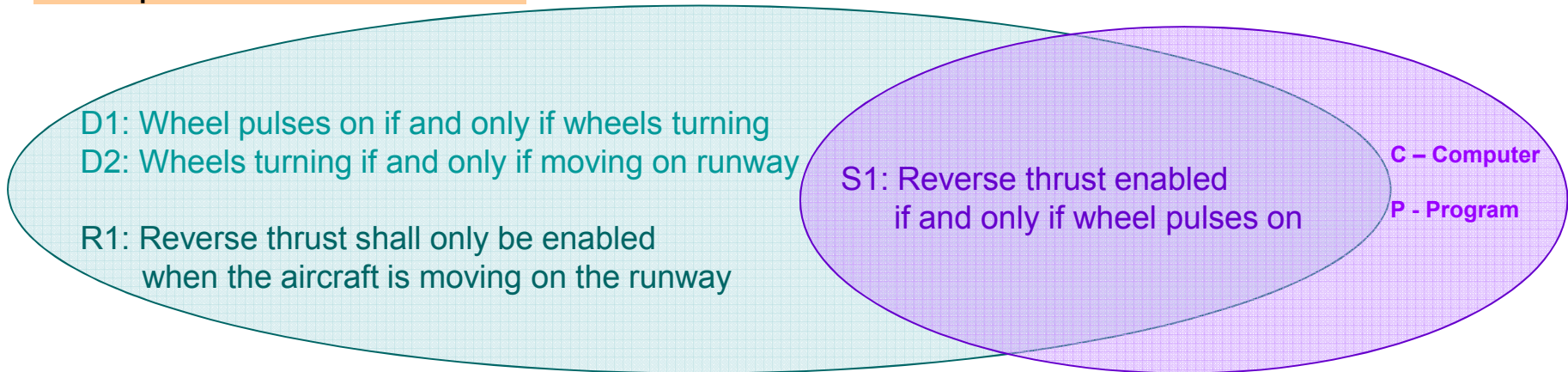


What if the domain assumptions are wrong?

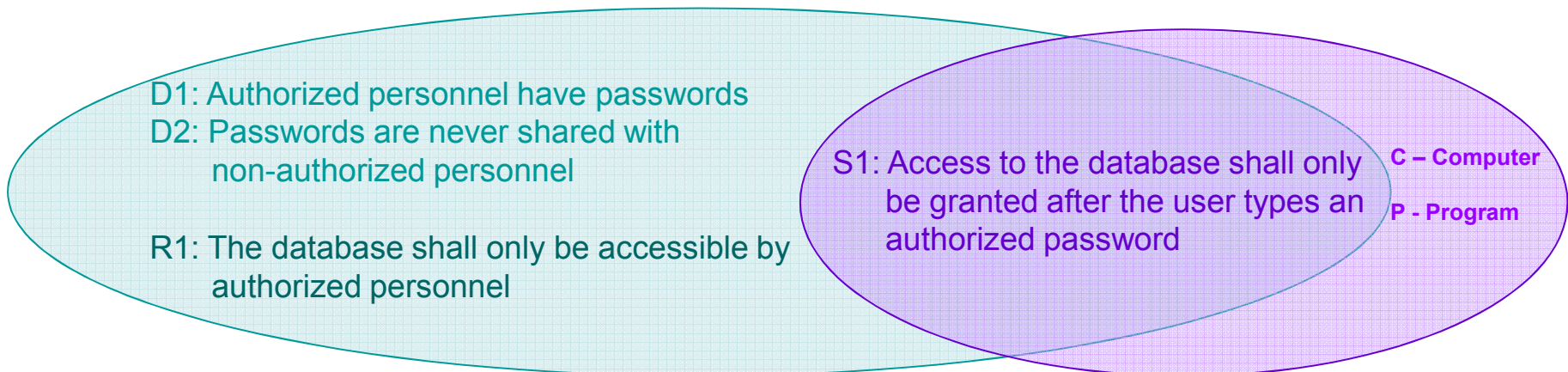


What are requirements?

Example 4: Aircraft Control



Example 5: Security



What if the domain assumptions are wrong?

What Are Requirements?

Example 7: In a single-customer banking environment,

- Notation: equational logic
- $action(a)$: atomic and sequential;
- $earlier(a1, a2)$: a nondense total order on actions;
- $pause(p)$: a unique pause between each adjacent pair of actions
- $begins(a,p)$: action a precedes pause p immediately in the temporal sequence
- $ends(a,p)$: action a succeeds pause p immediately in the temporal sequence
- $(*i \mid R(i) : P(i))$: the accumulation of values $P(i)$, using operator $*$, over all values i for which predicate $R(i)$ holds.

D

- $deposit(a,m)$: a is an action in which amount m is deposited
- $withdrawal-request(a,m)$: a is an action in which a withdrawal of amount m is requested
- $withdrawal-payout(a,m)$: a is an action in which amount m is paid out as a withdrawal
- $balance(b,p)$: during pause p the balance is amount b ;

At any time, the balance is equal to the sum of the amounts of all the previous deposits, minus the sum of the amounts of all the previous withdrawal payouts:

$$(\forall b, p \mid : balance(b, p) \equiv (b = (+m \mid (\exists a \mid : deposit(a, m) \wedge earlier(a, p)) : m) - (+m \mid (\exists a \mid : withdrawal-payout(a, m) \wedge earlier(a, p)) : m)))$$

R

A withdrawal request leads to a withdrawal payout, if the requested amount is less then the current balance

$$(\forall a, m, p, b \mid withdrawal-request(a, m) \wedge ends(a, p) \wedge balance(b, p) \wedge b \geq m : (\exists a' \mid : withdrawal-payout(a', m) \wedge earlier(a, a')))$$

S – a requirement (R), which is implementable, hence a specification

$$(\forall a, m, p \mid withdrawal-request(a, m) \wedge ends(a, p) \wedge m \leq ((+m \mid (\exists a \mid : deposit(a, m) \wedge earlier(a, p)) : m) - (+m \mid (\exists a \mid : withdrawal-payout(a, m) \wedge earlier(a, p)) : m)) : (\exists a' \mid : withdrawal-payout(a', m) \wedge earlier(a, a')))$$



What is RE?

What are requirements?

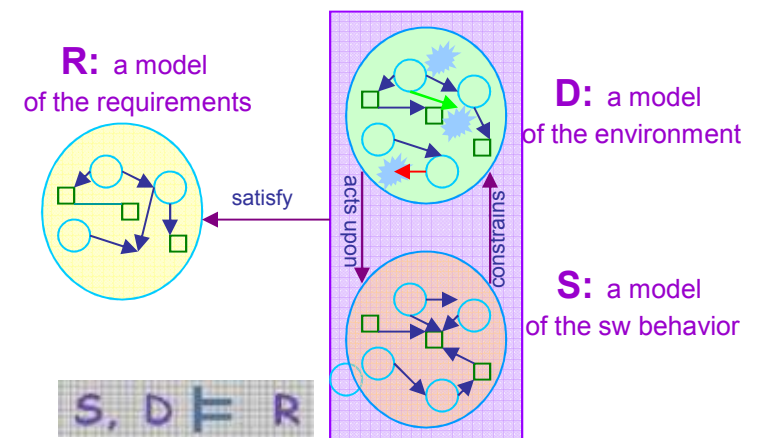
"... Requirements definition is a careful assessment of the needs that a system is to fulfill.

It must say **why** a system is needed,
based on current and and foreseen conditions,
which may be internal operations or an external market

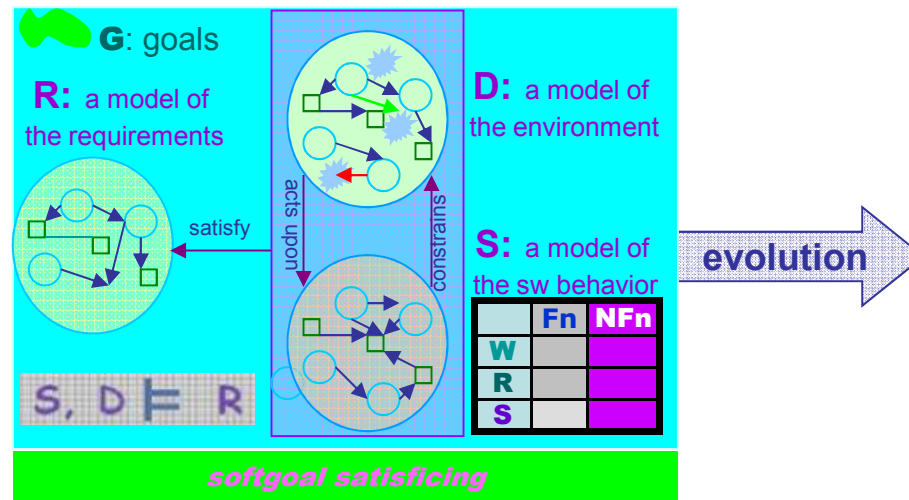
It must say **what** system features will serve and satisfy this context.

And it must say **how** the system is to be constructed ..."
[Ross77]

	Fn	NFn
E/D/B/W		
R		
S		



$$M^G, \text{Prog}^G \models S^G; S^G, D^G \models R^G; R^G, D^G \models G; (G \models \neg P) \vee (G \models \neg P)$$



$$M^G, \text{Prog}^G \models S^G; S^G, D^G \models R^G; R^G, D^G \models G; (G \models \neg P) \vee (G \sim \neg P)$$