Analisi dei requisiti
Il metodo Jackson

Vincenzo Gervasi, Laura Semini
Ingegneria del Software
Dipartimento di Informatica
Università di Pisa

Riassunto lezione precedente
Outline della lezione

- Lezione precedente:
  - Importanza dell'attività di analisi dei requisiti
  - Dominio
    - Comprensione e modellazione
  - Requisiti
    - Acquisizione
    - Analisi
- Questa lezione
  - Il metodo Jackson

Context diagrams, Problem diagrams e Problem frames (Metodo Jackson)

Michael Jackson (not the singer)
Consultancy & Research in Softw

Visi
D

Inde
(I

Il problema della correttezza

$$M, W \models R$$

- M: specifica del sistema (macchina)
- W: descrizione del dominio (mondo, world) in cui opera il sistema
- R: requisiti
- La formula si legge:
  - Il sistema, come specificato da M, se operante in un mondo descritto da W, soddisfa R

Tre elementi

- La macchina
  - Tipicamente hardware+software
- Il mondo
  - Tutto ciò che non è macchina
  - Il contesto/dominio
- Le interfacce
  - I fenomeni condivisi tra macchina e mondo

La macchina e il mondo

La macchina opera attraverso queste interfacce

Spazio della soluzione

La soluzione sta nella macchina

Spazio del problema

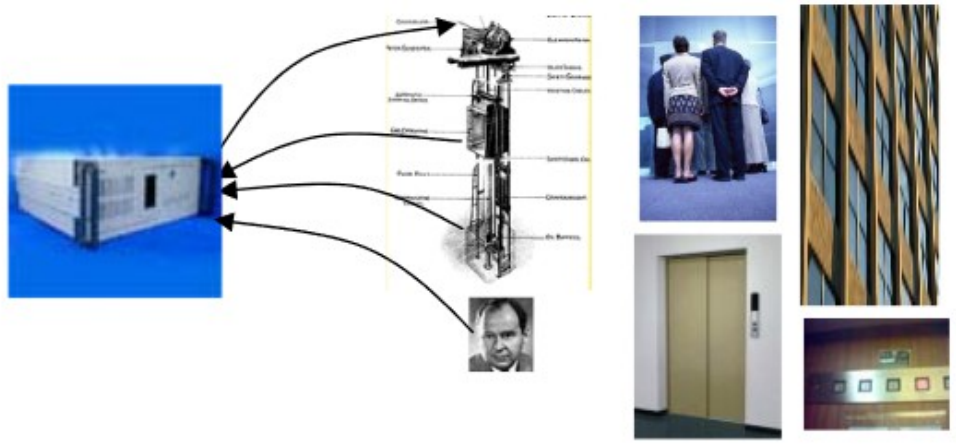
Il problema sta nel mondo

Introduzione con esempi

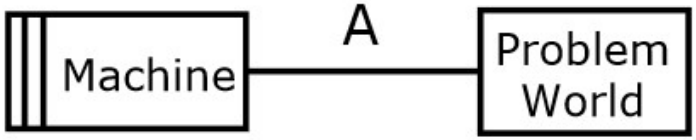
Il mondo (del problema)

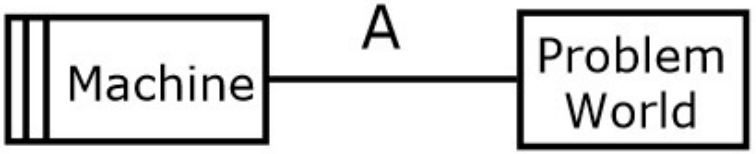
- Capire i requisiti significa capire un problema, che è nel mondo

machine problem world



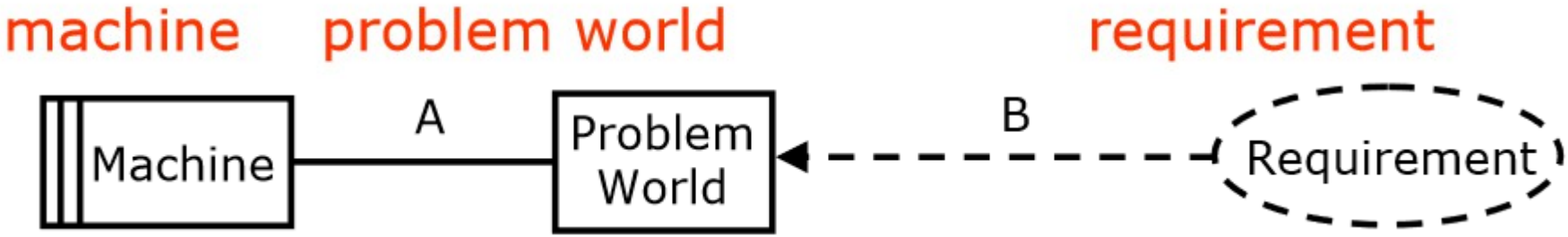
machine problem world





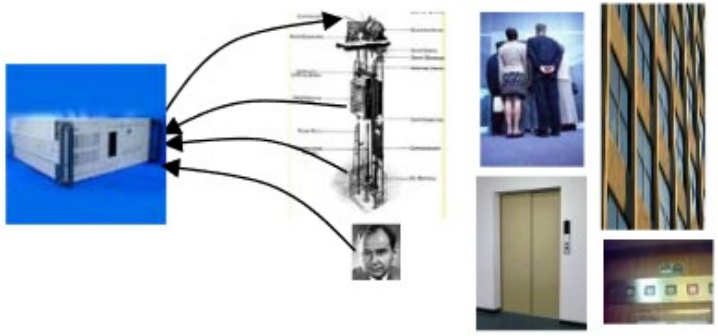
Il mondo e i requisiti

- only members can borrow books
- reserved books available to collect
- reminders sent for overdue loans
- fees and fines are collected
- catalogue is up to date
- management reporting …
- … …



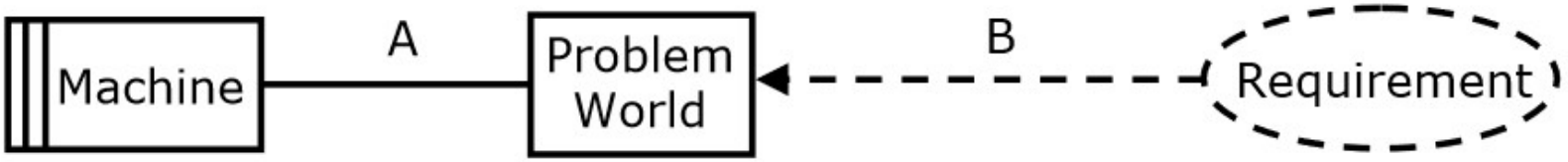
- Un requisito è una condizione nel mondo (contesto del problema)
  - Espressa in termini dei fenomeni B

Il mondo (dell'ascensore) e i requisiti



- lift comes on demand
- lift goes to requested floor
- safety against equipment failure
- gives efficient service
- current car location shown
- adjustable priorities ...
- ... ...

machine problem world requirement

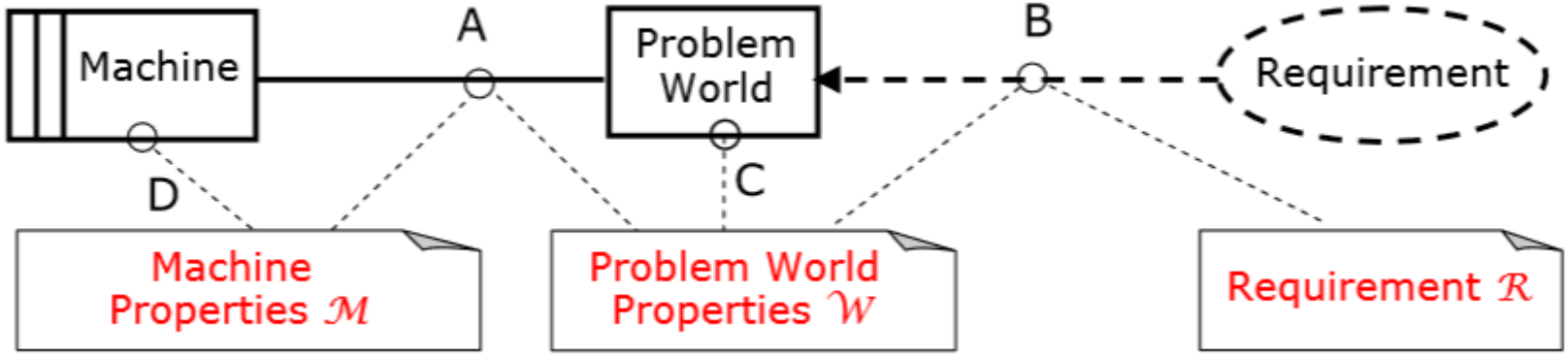


- I requisiti parlano il linguaggio del contesto del problema
- La macchina deve realizzarli
  - Per esempio: l'ascensore arriva su richiesta
- Il mondo ha caratteristiche/comportamenti indipendenti dalla macchina
  - Per esempio: il motore dell'ascensore

Requisiti o proprietà del mondo?

- L’ascensore non si muove se le porte sono aperte
- Il piano n può essere raggiunto da n-2 solo tramite n-1
- Se la freccia in alto è illuminata l’ascensore non si muove verso il basso
- Il pulsante di richiesta al piano è spento quando l’ascensore si ferma al piano
- Le porte al piano e le porte dell'ascensore si aprono e chiudono insieme
- Nel display che indica la posizione dell’ascensore è illuminato un solo

La macchina e il mondo collaborano per risolvere il problema

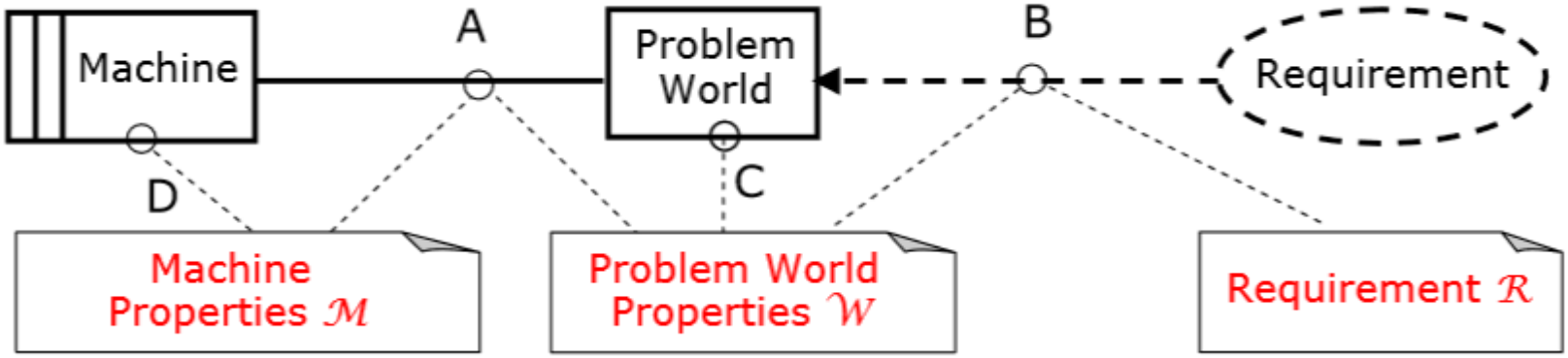


- Fenomeni
  - A: fenomeni condivisi tra la macchina e il mondo
    - MotorOn, sensore[P], DoorMotorOn, TastoPremuto[B], ..
  - B: fenomeni menzionati nei requisiti
    - AscensoreArriva [P], AperturaPorte, Richiesta AscensoreAlPiano[p], ...
  - C, D: fenomeni privati (non condivisi) della macchina e del mondo
- La macchina e il mondo collaborano

$$M, W \models R$$

... per soddisfare il requisito

$\mathcal{M}, \mathcal{W} \models \mathcal{R}$



- Quando il pulsante [2p] è premuto, e Sensore[0..1] è il più recente sensore di piano, e MOTORON è falso:
  - set dirUp e MOTORON;
  - aspetta fino a che sensore [2] è acceso; imposta motore spento; ...

- Il pulsante [2p] è premuto se e solo se un utente chiama l'ascensore al piano 2;
- dirUp e motorOn fanno sì che l'ascensore salga;
- sensore [2] si accende quando l'ascensore è al piano 2;
- quando MOTORON è falso ascensore si ferma;

- Quando l'ascensore è a un piano, e l'utente chiama l'ascensore ad un altro piano, questo si muove; ...

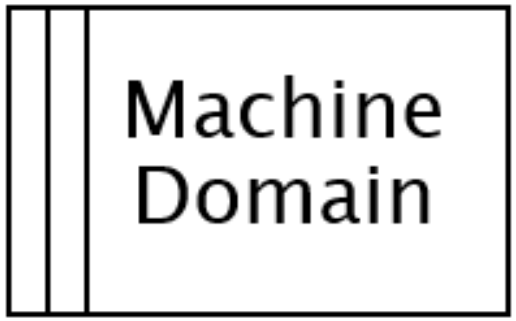
Context diagrams

Context diagrams

- Descrivono la relazione tra:
  - Il contesto del problema (dominio applicativo)
  - La soluzione software
  - Le interfacce tra dominio e soluzione
- Dominio organizzato in sotto-domini:
  - Machine domain
    - Hw e soluzione sw da realizzare
  - Problem domain
    - Mondo reale esterno alla macchina

La macchina

- Rettangolo con due righe verticali
  - Nei context diagrams uno solo in ogni diagramma
  - (Questo vincolo non vale nei problem diagrams)



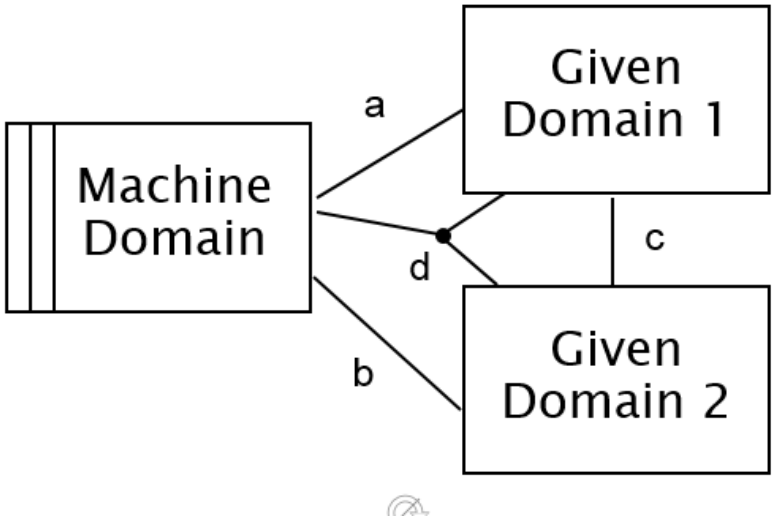
Domini per strutturare il mondo

- Given Domain
  - Il dominio dato
  - Non può essere modificato
  - Un rettangolo
- Designed Domain
  - La rappresentazione del dominio nel sistema
  - Viene costruito costruendo il sistema
  - Rettangolo con una linea verticale

Strutturare il mondo
Esempio: infrastrutture di comunicazione

- Alcune infrastrutture di comunicazione fanno parte del mondo
  - Internet
  - reti cellulari
  - linee che collegano i negozi alle banche
- Altri possono essere progettati e sono quindi parte del sistema
  - Il formato esatto degli SMS inviati da un dispositivo di monitoraggio antifurto di una macchina

Le interfacce



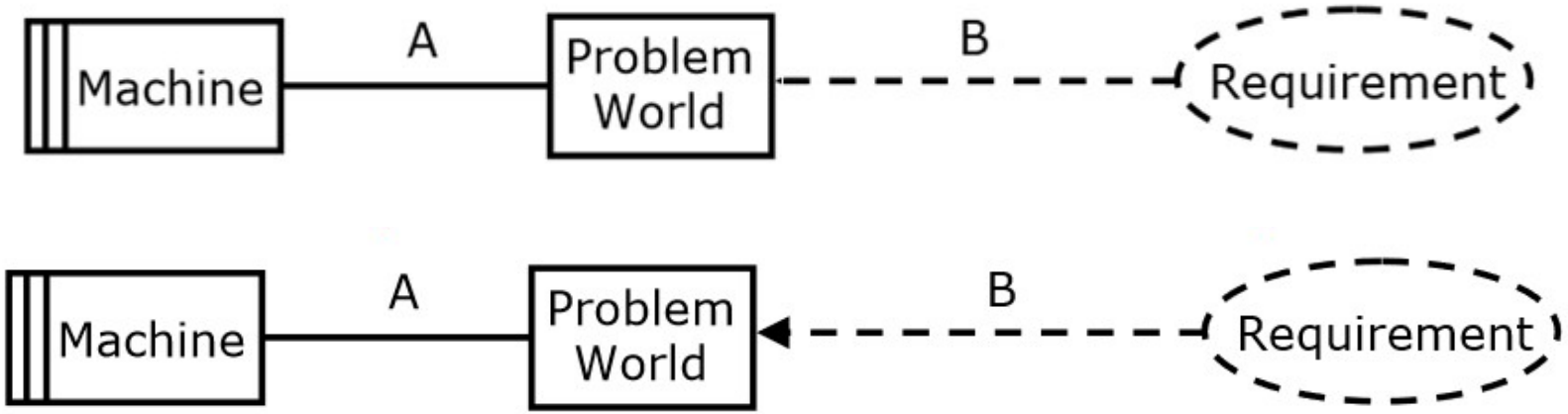
- Le relazioni tra domini (interfacce)
  - Quando due (o più) domini condividono un fenomeno
  - I fenomeni possono essere:
    - Stati
    - Valori
    - Eventi

Problem diagrams

Problem diagrams

- Estendono i context diagrams introducendo i requisiti
  - Mostrano le relazioni tra requisiti e sotto-domini del dominio applicativo.

Problem diagrams



- I requisiti sono ovali tratteggiati
- Linee tratteggiate
  - Indicano che il requisito parla del mondo
- Frecce tratteggiate
  - indicano che il requisito impone dei vincoli sul

Decomporre il problema

E decomporre il mondo in domini

La macchina

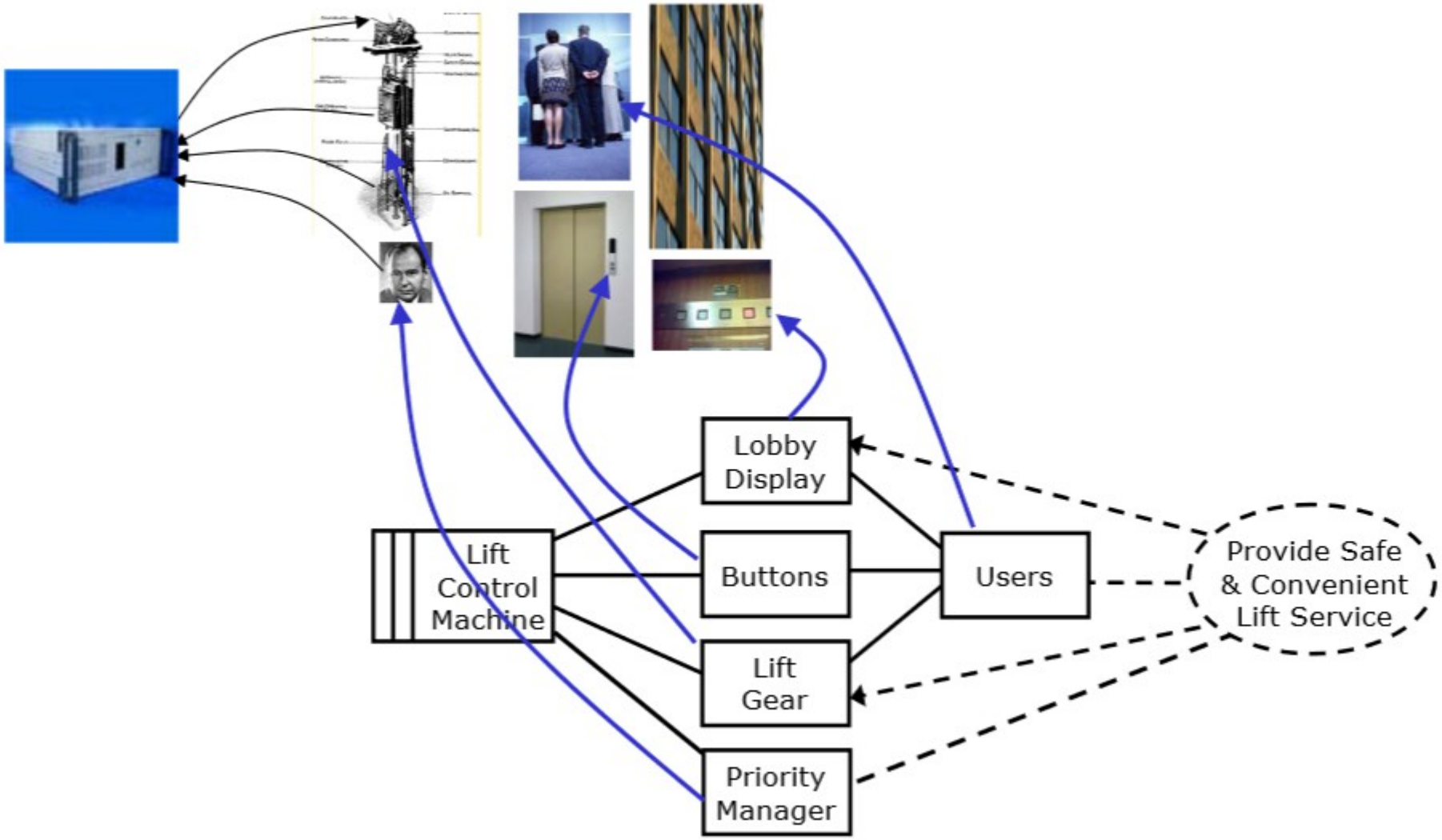
- Dato un problema:
  - La macchina è quello che dobbiamo sviluppare
  - Il mondo è ciò che è dato
- Una macchina è un vista parziale delle funzioni del software
- Ogni problema ha una macchina
  - Nel sistema che sarà costruito, una macchina può rappresentare ...
    - ... tutto o parte di ...
    - ... uno o più computer + software

La macchina è relativa ai problemi

- > 1 macchina significa > 1 problema

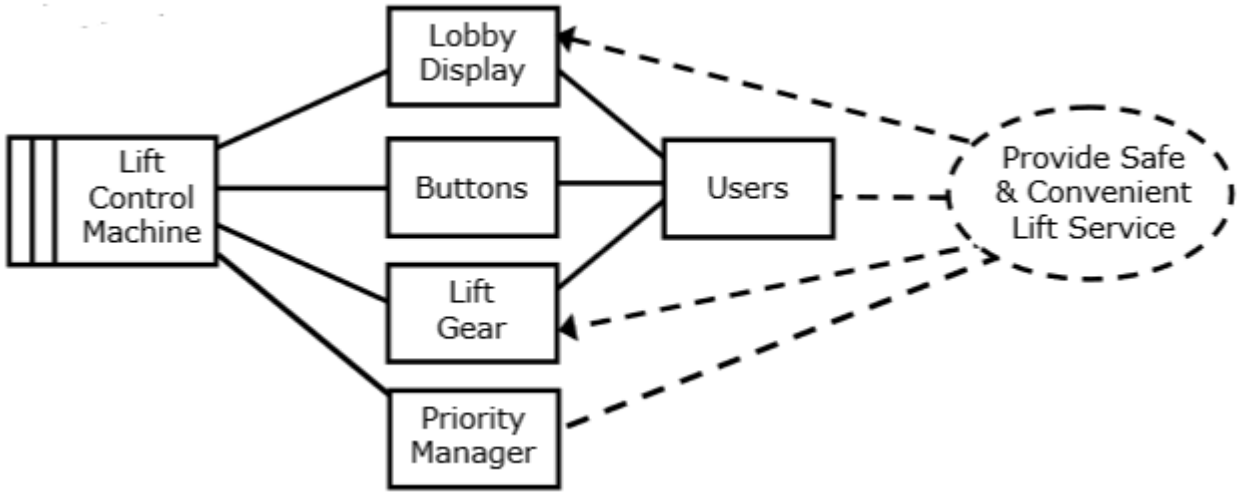
- La macchina in un problema ...
  - ... Può essere un dominio in un altro problema
- Si possono scomporre i problemi in sottoproblemi
  - Un sottoproblema è un problema più semplice
  - Un sottoproblema è chiamato componente (del problema)

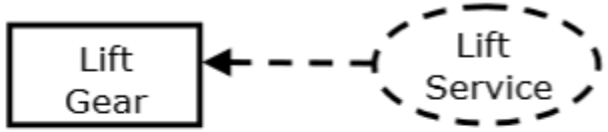
Esempio


Lobby Display
Lift Control Machine
Buttons
Users
Provide Safe & Convenient Lift Service
Lift Gear
Priority Manager

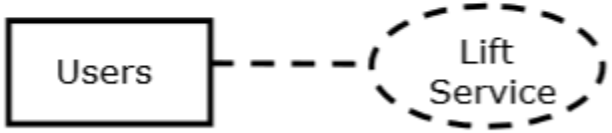
Un requisito nomina o vincola un sottoinsieme dei domini che formano il mondo

- In generale, un requisito vincola alcuni domini del mondo e non altri





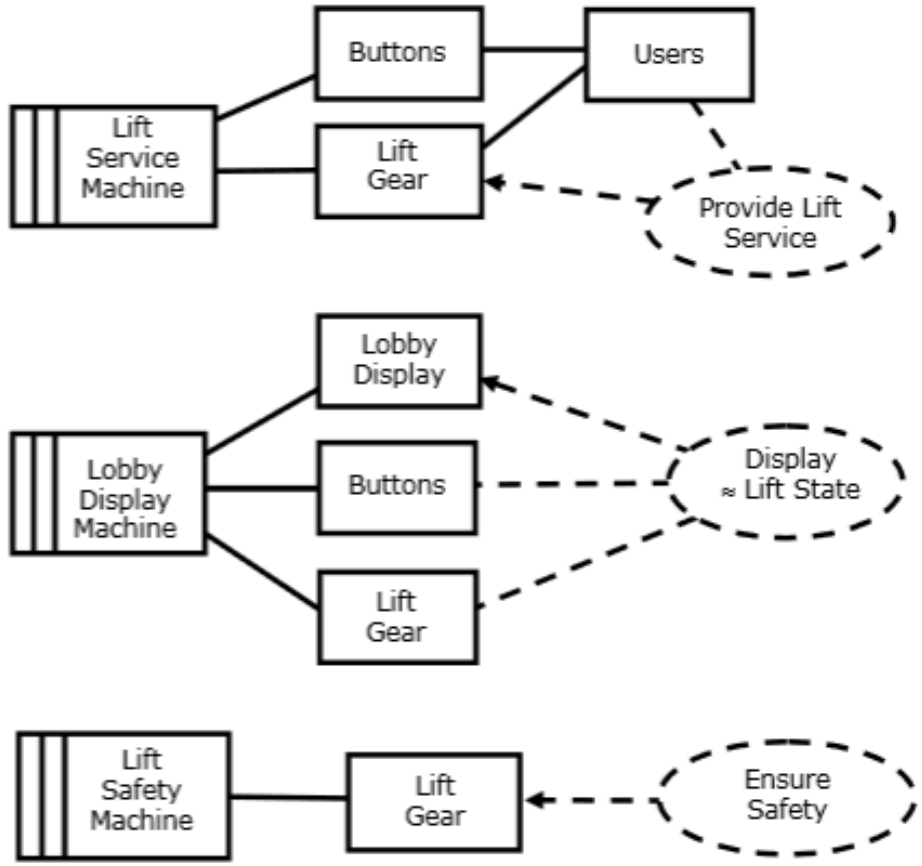
- The requirement constrains ◀- - - - this domain



- The requirement only refers to - - - - this domain

Sottoproblemi

- Fornire un servizio di ascensore agli utenti
- Mostrare sul display richieste e posizione dell'ascensore
- In caso di malfunzionamento tirare il freno di emergenza



Rivediamo tutto attraverso un esempio

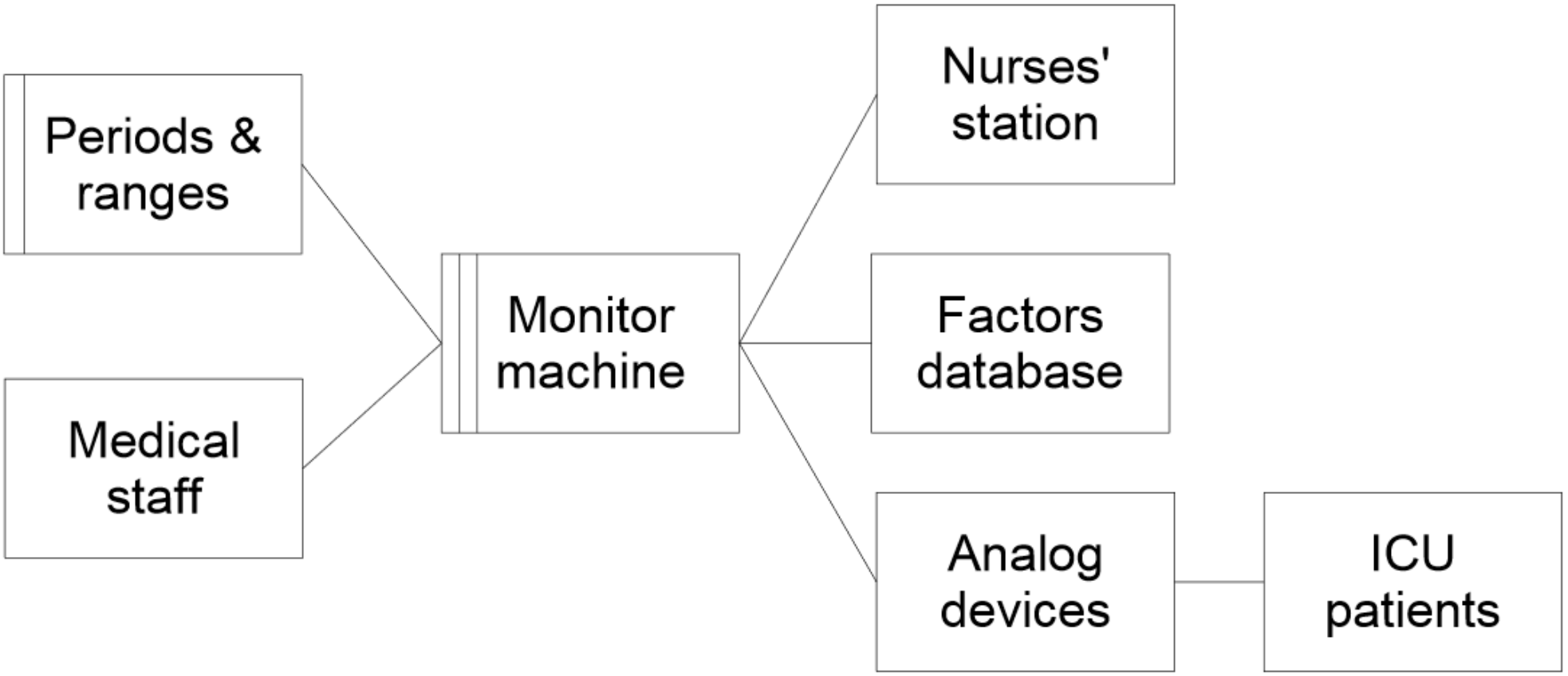
Un esempio: Monitoraggio di un paziente

Per la terapia intensiva in un ospedale è necessario un programma di monitoraggio dei pazienti. Ogni paziente viene controllato da un dispositivo analogico che misura fattori quali battito , temperatura e pressione sanguigna. Il programma legge questi fattori sulla base periodica (specificati per ogni paziente) e memorizza i fattori in un database (pre-esistente). Per ogni paziente sono specificati dal personale medico gli intervalli di rifermento, per ciascun fattore. Se il fattore cade al di fuori del range di sicurezza del paziente , o se un dispositivo analogico fallisce, viene notificata la postazione degli infermieri

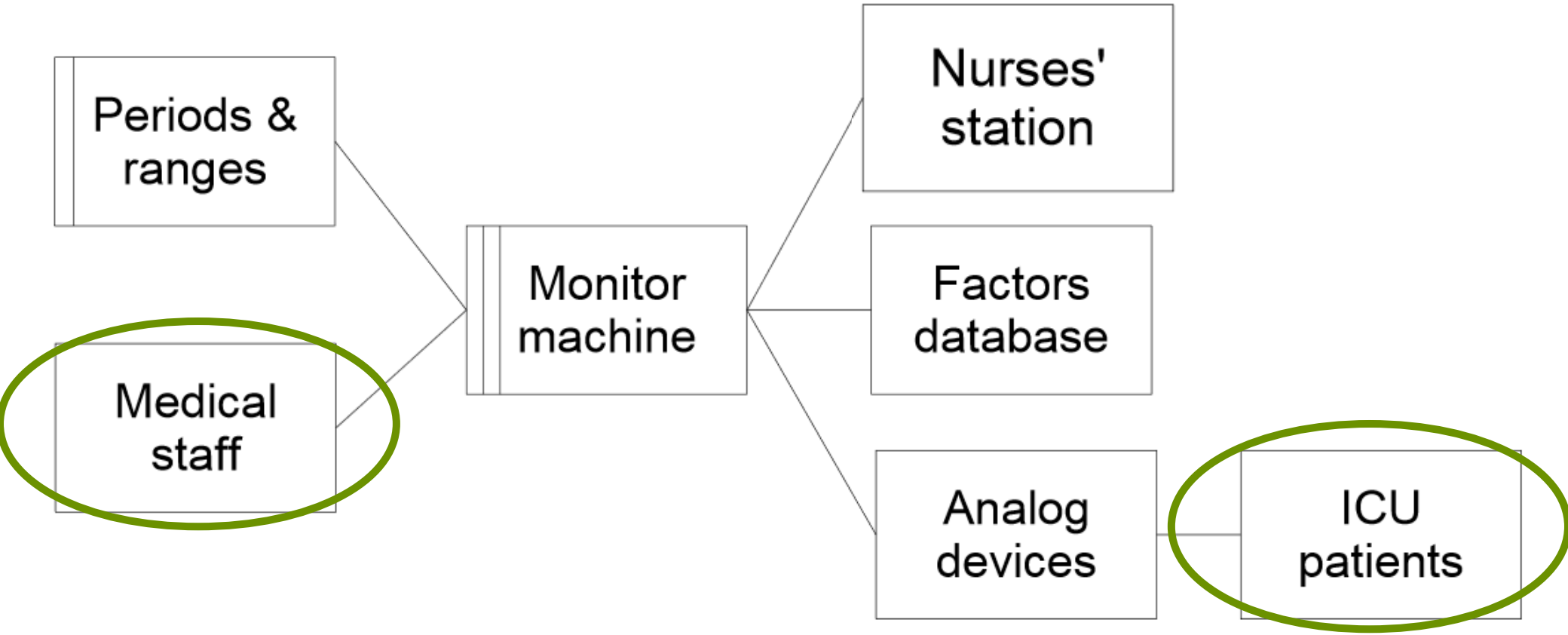
Un esempio: Monitoraggio di un paziente

Per la terapia intensiva in un ospedale è necessario un programma di monitoraggio dei pazienti. Ogni paziente viene controllato da un dispositivo analogico che misura fattori quali battito , temperatura e pressione sanguigna. Il programma legge questi fattori sulla base periodica (specificati per ogni paziente) e memorizza i fattori in un database (pre-esistente). Per ogni paziente sono specificati dal personale medico gli intervalli di rifermento, per ciascun fattore. Se il fattore cade al di fuori del range di sicurezza del paziente , o se un dispositivo analogico fallisce, viene notificata la postazione degli infermieri.

Context diagram



Utenti e interfacce utente



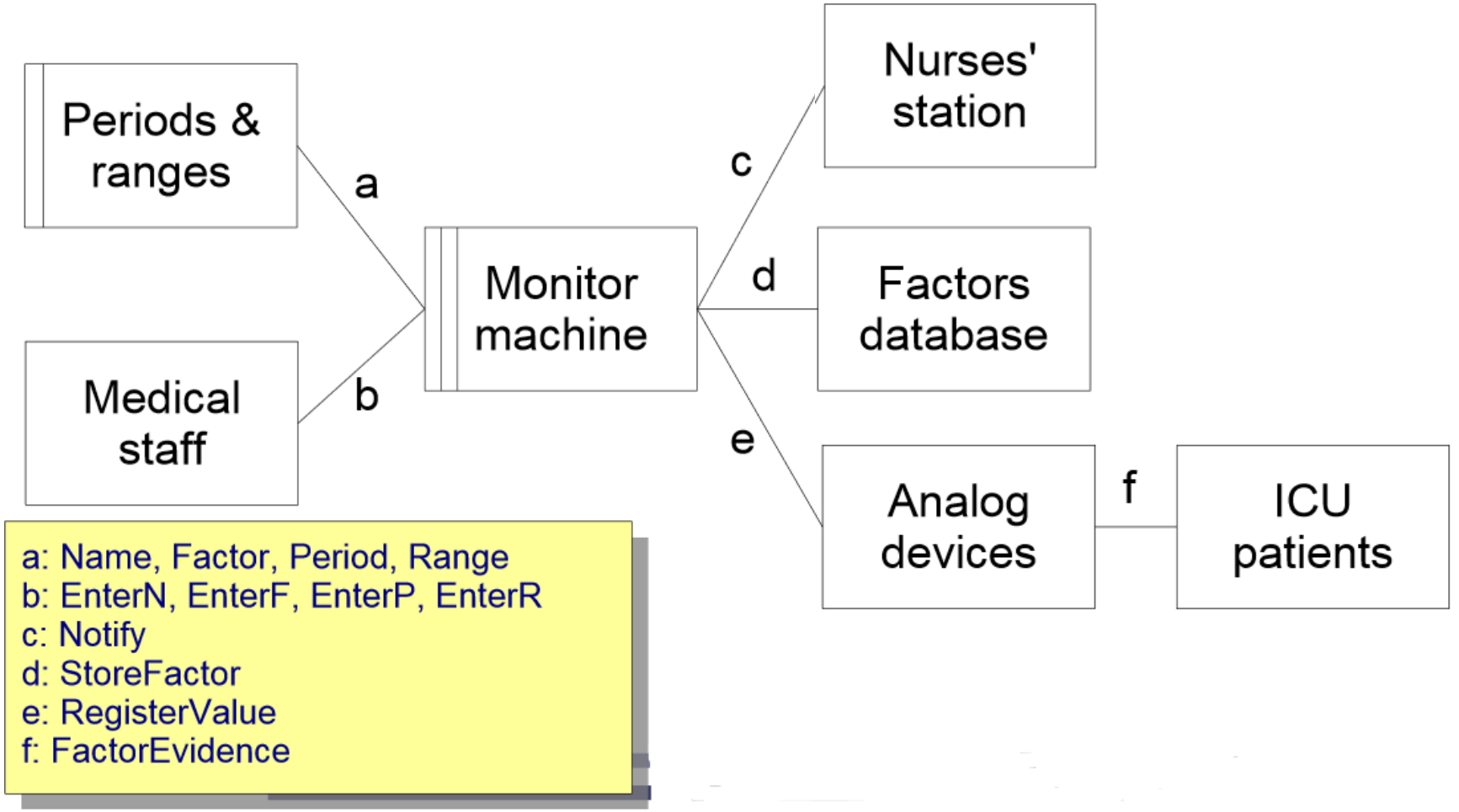
Interfacce e fenomeni condivisi

- E’ importante distinguere la nozione di interfaccia tra domini da quello di interfaccia utente
- Un'interfaccia tra domini è un insieme di fenomeni condivisi (Eventi , stati, valori , ...)
  - Tutti i domini di un'interfaccia condividono i fenomeni
  - Solo un dominio può causare un dato fenomeno
  - I fenomeni possono trasportare informazioni (parametri)

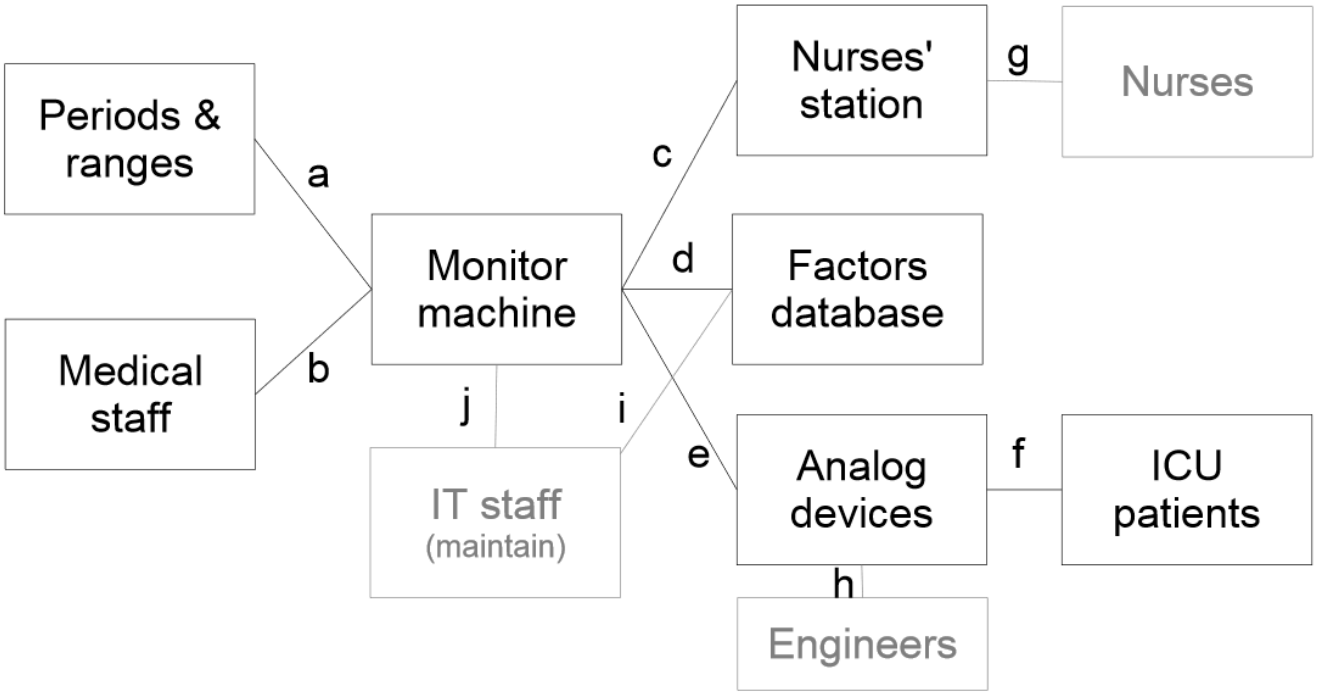
Tipi di fenomeni

- Individuali
  - Eventi: un'istanza di qualcosa che accade
  - Entità : qualcosa che persiste nel tempo
  - Valori: un elemento di una data sorta (algebrica)
- Relazioni
  - Stati: i rapporti tra le entità e valori
  - Verità : predicati tra gli individui (statici)
  - Ruoli : i rapporti tra gli eventi e gli individui

Nel nostro esempio



Altre interfacce?

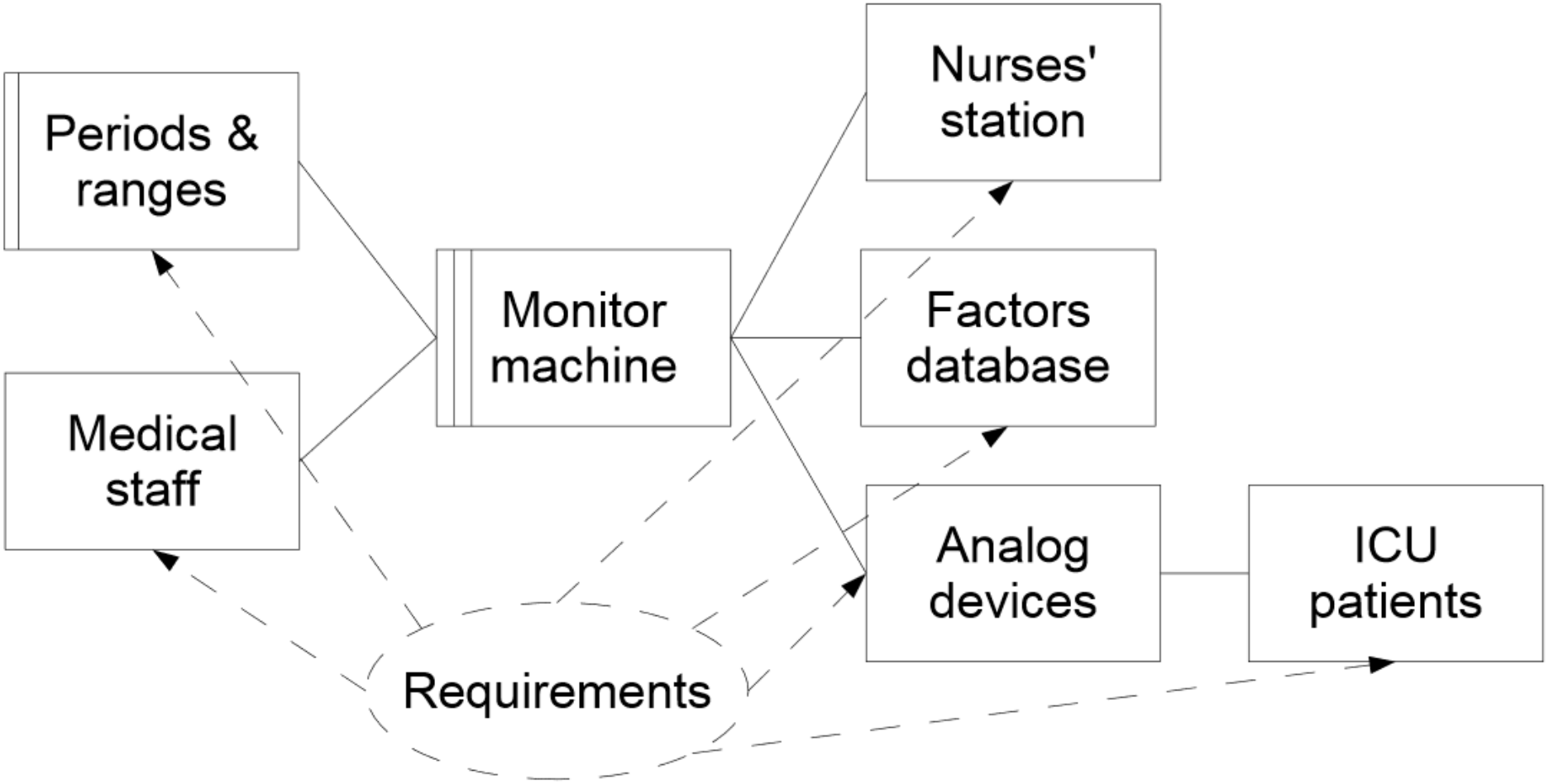


- Ragionevoli ma esterne al nostro problema
- Ci devono essere? Le dobbiamo assumere?
  - Materia per colloqui con il cliente

Torniamo ai requisiti

- L'aggiunta di una bolla "Requisiti " collegata con tutti i domini, non aiuta molto
- Però, dal momento che abbiamo inserito nel context diagram tutti i domini di rilevanza per i requisiti, per definizione avremo frecce dai requisiti a tutti loro
- Non molto utile, aggiunge solo complessità

Esempio


Periods & ranges
Medical staff
Monitor machine
Nurses' station
Factors database
Analog devices
ICU patients
Requirements

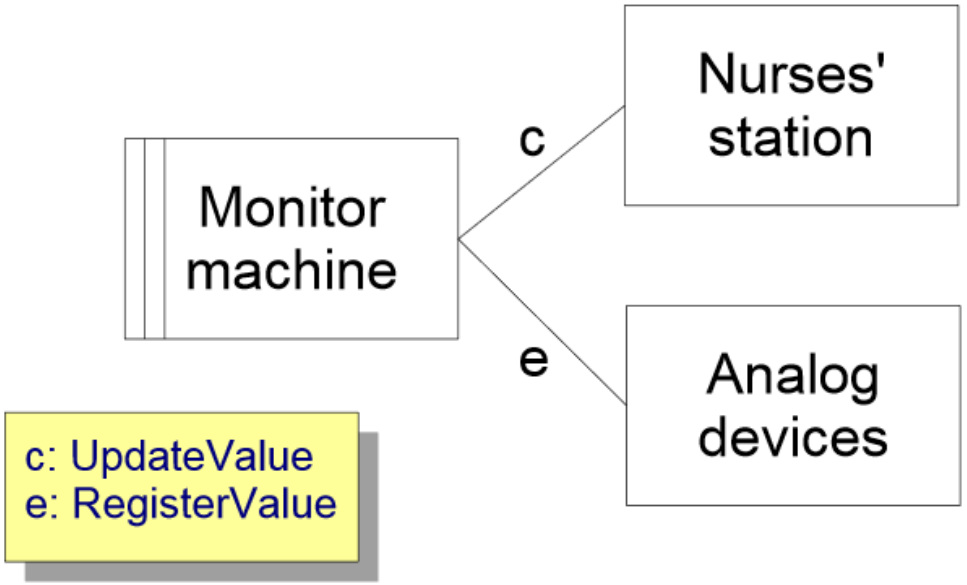
Decomporre problemi complessi

- Il modo classico per gestire la complessità è la decomposizione in sottoproblemi

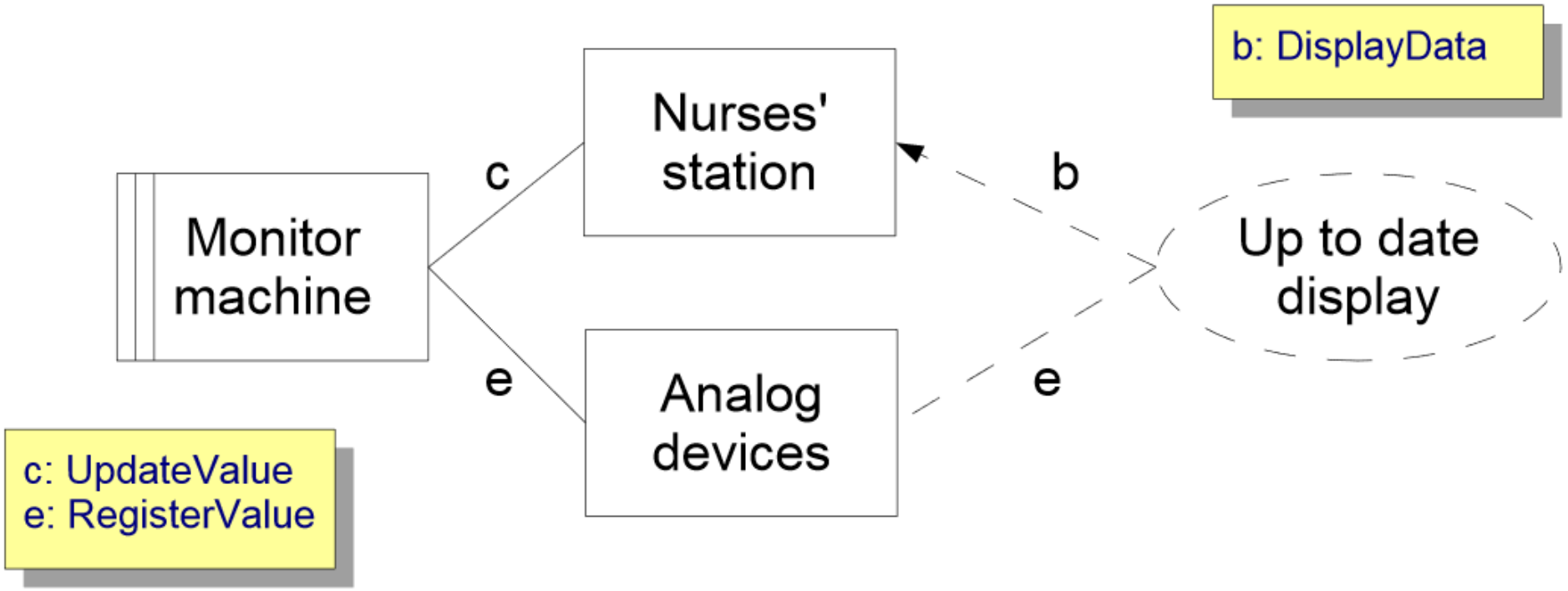
- per analogia : se un compito è complesso , dividerlo in passi più semplici
- in contrasto : i passi sono sequenziali e distinti, i sottoproblemi spesso non lo sono

Semplifichiamo

- Consideriamo un semplice problema collegato: mostrare i valori grezzi dei sensori analogici
  sulla p



Aggiungiamo i requisiti



Scrivere i requisiti

- Ma come sono scritti i requisiti?
  - Obiettivo principale: le relazioni tra i fenomeni di interfaccia e i domini devono essere chiari
- Linguaggio formale?
  - Certo, se avete bisogno di sicurezza
  - Logica, automi, diagrammi di stato, equazioni, ...
- Informali?
  - Certo, purché sia sufficientemente rigorosi per permettere l'implementazione della specifica
  - Linguaggio naturale, disegni, ...

Annotazioni

- Le interfacce sono annotate, con sintassi

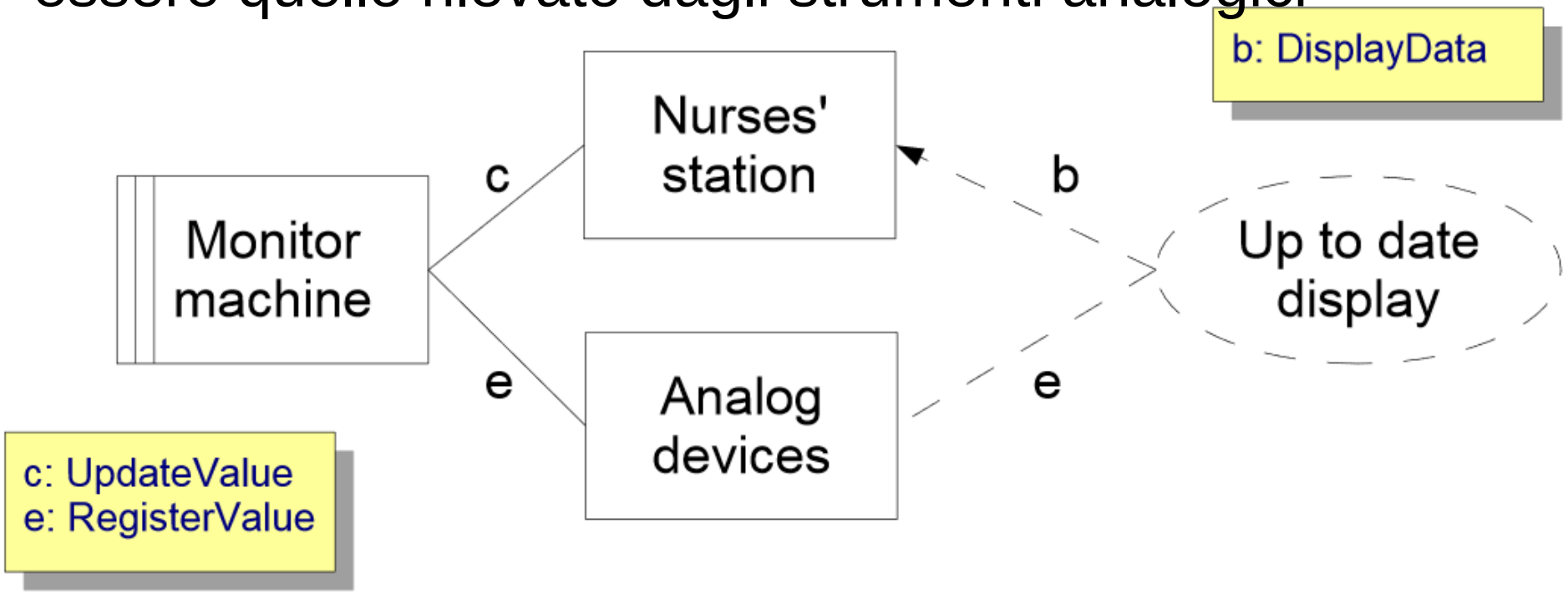
  dominio! {eventi}

- Quando *dominio*! controlla gli *eventi* condivisi

Scriviamo i requisiti

$\forall AD!RegisterValue(factor,v).NS!DisplayData(factor)=v$

Il valore mostrato sulla postazione degli infermieri deve essere quello rilevato dagli strumenti analogici



Come risolvere il problema

Ricordiamo:

$$M, W \models R$$

- Dati i requisiti, dobbiamo:
  - Descrivere le proprietà del dominio
  - Costruire la specifica macchina

Scriviamo $\mathcal{M}$ e $\mathcal{W}$ tali che $\mathcal{M}, \mathcal{W} \models \mathcal{R}$

- R - Requisito: (display aggiornato)
  ∀AD!RegisterValue(factor,v).NS! DisplayData(factor)=v

- W - Dominio: (la postazione degli infermieri va aggiornata)
  MM!UpdateValue(factor,v)⇒NS! DisplayData(Factor)=v

- M – Specifica della macchina: (programma da scrivere)

Bibliografia

- Fuggetta cap 4.
- Testo di approfondimento
  - M. Jackson, "Software Requirements & Specifications", Addison-Wesley, 1995.