

---

## TEMA 3

# DISSENY DEL SOFTWARE

### 1. Introducció.

- Procés de disseny.
- Disseny de dades, disseny arquitectònic, disseny de la interfície, disseny procedimental.
- Principis (objectius) del disseny.

### 2. Conceptes del disseny.

- Abstracció.
- Modularitat.
- Refinament.

### 3. Disseny modular efectiu.

- Independència funcional.
- Cohesió
- Acoblament.
- Heurístiques per a un disseny modular efectiu.
- Arquitectures de software.

### 4. Disseny d'Interfícies d'Usuari

- **Definició.** El procés d'aplicar diferents tècniques i principis amb el propòsit de definir un dispositiu, un procés o un sistema amb suficients nivells de detall com per permetre la seva realització física.

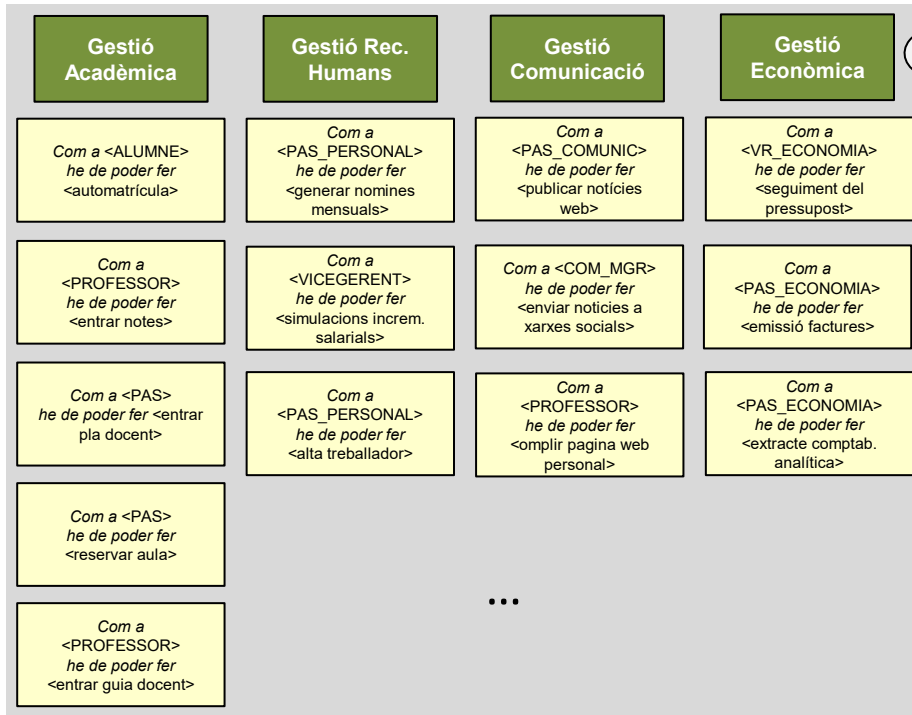
Està situat en el nucli tècnic del procés d'E.S. I s'aplica independentment del paradigma de desenvolupament utilitzat.

*Domini del Problema => Domini de la Solució*

- **Objectiu.** Produir un model o representació del sistema que pugui ser utilitzat, en una fase posterior, a fi d'implementar-lo.

# Dels requisits a l'arquitectura

## DOMINI DEL PROBLEMA (AN. REQUISITS: PRODUCT BACKLOG)

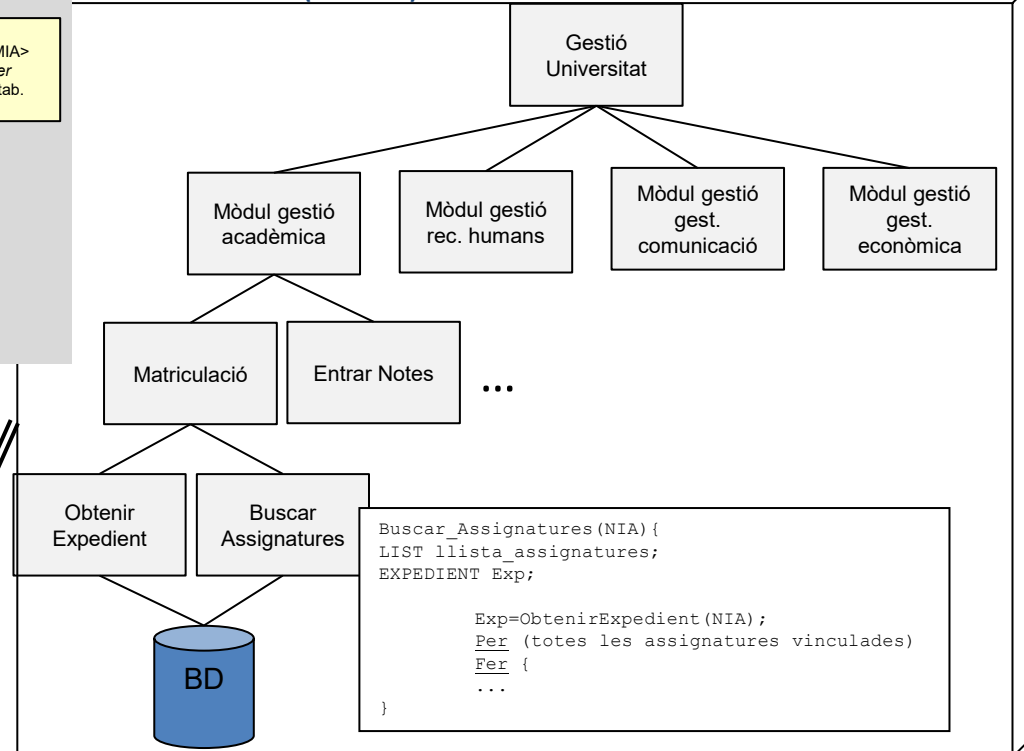


**QUÈ**  
farà el software

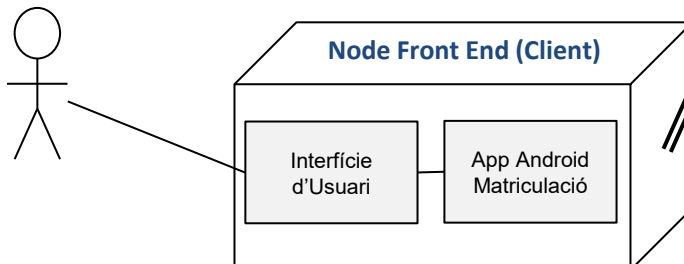
**COM**  
ho resoldrem  
tècnicament

## DOMINI DE LA SOLUCIÓ (DISSENY)

### Node Back End (Servidor)




### Node Front End (Client)



- **Disseny preliminar:** centrat en la transformació dels requisits de les dades i en l'arquitectura del software (estructura modular).
- **Disseny detallat:** conceptes més específics, refinament de l'estructura de dades i representació algorísmica dels mòduls.

	D. Preliminar	D. Detallat
D. Dades	X	X
D. Arquitectònic	X	
D. Procedimental		X
D. Interfície	X	

 Moltes vegades es fa prèviament

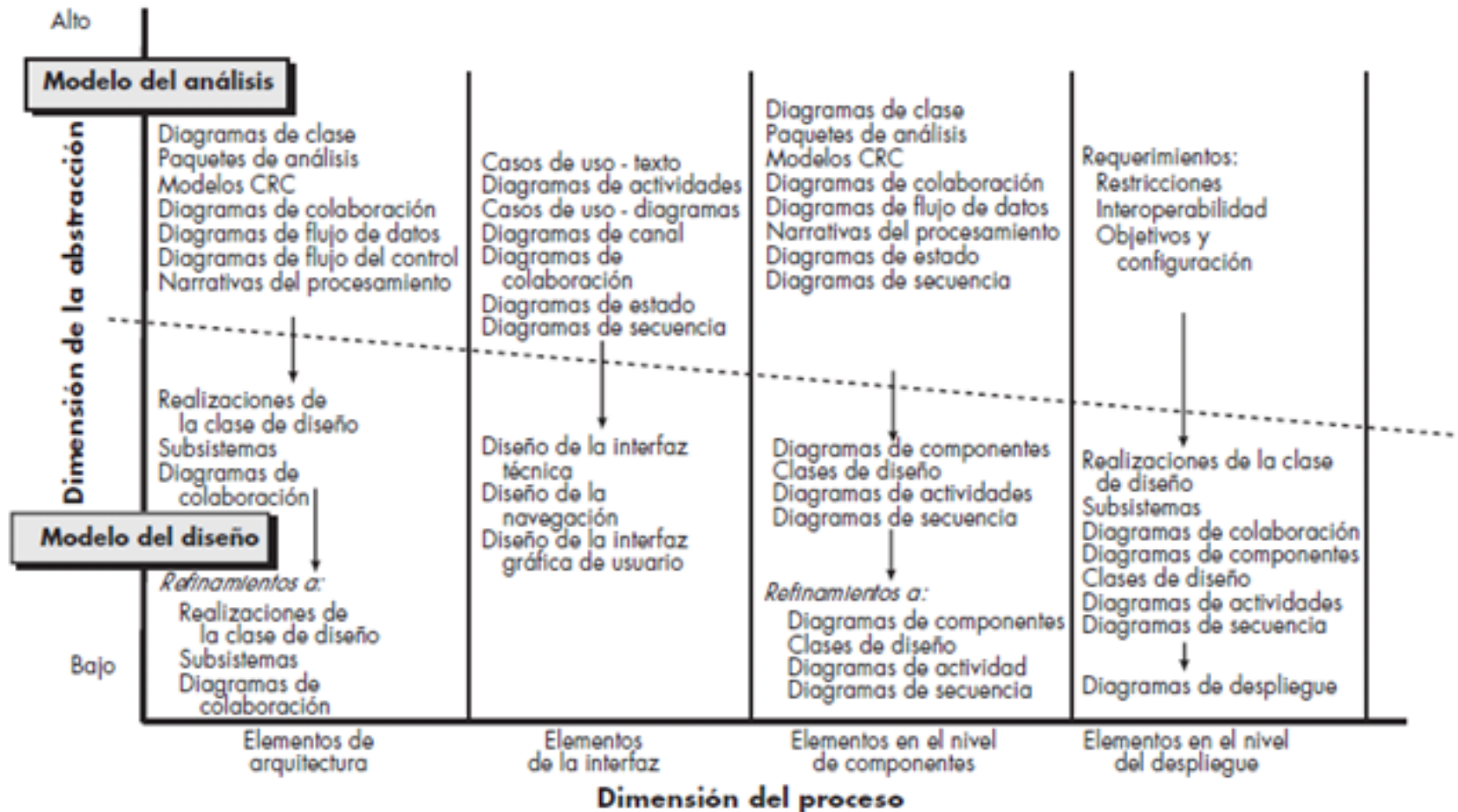
## D. Dades, D. Arquit., D. Procedural, D. Interf.

---

- **D. Dades.** Transforma el model de dades de l'anàlisi (diccionari, en l'estructurat o els atributs, en l'OO), en les estructures de dades que després s'implementaran.
- **D. Arquitectònic.** Defineix la relació entre els principals elements estructurals del programa. Es desenvolupa una estructura modular i jeràrquica del software.
- **D. Procedimental.** Transforma els elements estructurals de l'arquitectura del programa en una descripció procedimental dels elements del software. Descripció de les funcions a un nivell més proper al LP que el nivell de les EP. de l'anàlisi.
- **D. Interfície.** Descriu com es comunica el software amb ell mateix, amb els sistemes que operen amb ell i amb els operadors que l'utilitzen.

# Dimensions del model de disseny

Reproduït de [Pressman2010]



## Principis (objectius) del disseny

---

- **Verificabilitat.** Poder avaluar la correctesa del disseny.
- **Completesa.** Totes les components (estructures de dades, mòduls, interfícies externes, etc.) han de ser especificades.
- **Consistència.** Que no hi hagi inconsistències inherents.
- **Eficiència.** Utilització adequada dels recursos del sistema. Els recursos no són infinits.
- **Seguible.** Tots els elements del disseny han de poder-se “mapejar” cap a l’anàlisi de requisits.
- **Simplicitat/Comprensible.** Cal tenir en compte que el document de disseny, igual com era el d’especificació de requisits, serà utilitzat com a base per a les etapes posteriors.

Permet definir components de manera abstracta a diferents nivells, sense preocupar-nos dels seus detalls d'implementació. Una abstracció descriu el *comportament extern* de la component, sense haver de parar atenció als detalls interns que produeixen el comportament.

L'abstracció és un ajut a la modularització ja que permet veure el comportament extern dels mòduls a fi de connectar-los.

estructurat

- **Abstracció Funcional.** Abstracció d'una component des del punt de vista de la funció que realitza. Una seqüència d'instruccions que té una funció específica i limitada.

*Exemple:* porta => obrir, tancar.

O.O.

- **Abstracció de dades.** Col·lecció de dades que descriuen un objecte de dades.

*Exemple:* porta => tipus, pes, dimensions, direcció d'obertura, etc.

temps real

- **Abstracció de control.** Implica un mecanisme de control del programa sense especificar detalls interns.

*Exemple:* porta => si es prem el botó d'obrir llavors...



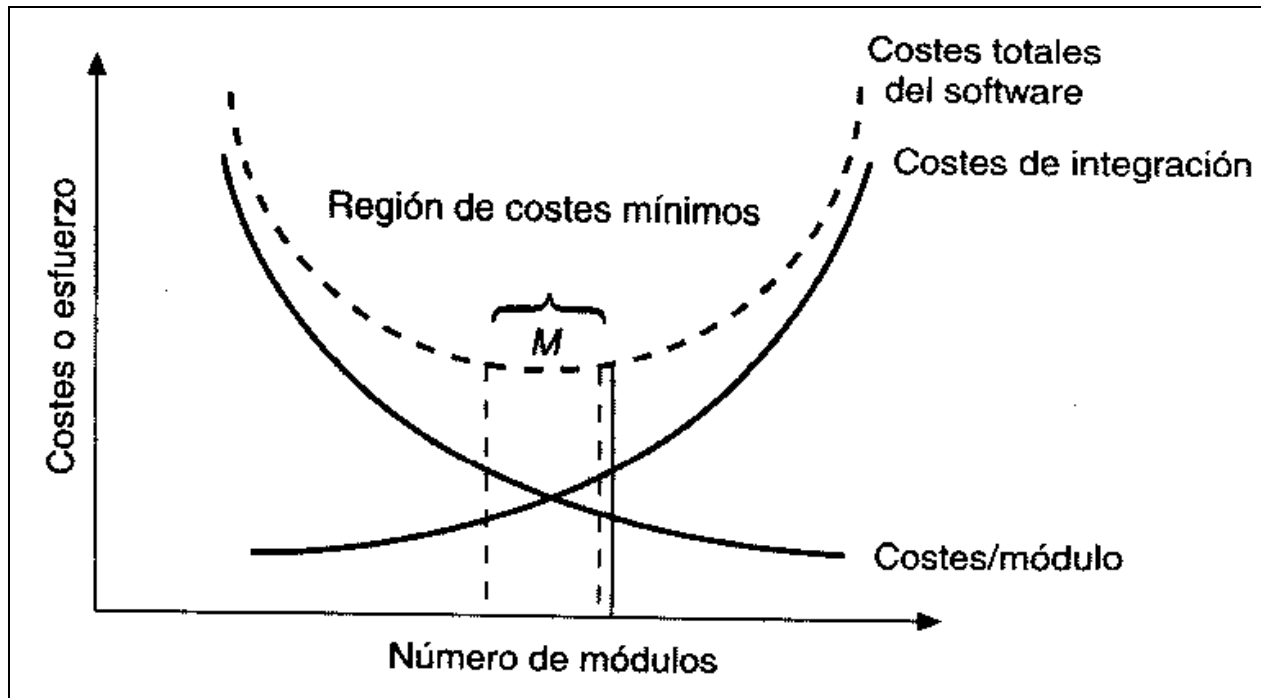
- Aplicar el principi de *divide&conquer* al disseny de software.
- **Mòdul.** És la unitat bàsica d'un programa, és finita i identificable respecte a la compilació i la lectura. És lògicament separable. Pot ser una macro, una funció, un conjunt de dades i funcions associades, un tipus abstracte de dades, etc.
- Mida recomanable d'un mòdul:
  - $C(p)$  grau de complexitat d'un problema  $p$ .
  - $E(p)$  esforç per programar una solució.
  - $C(p_1) > C(p_2) \Rightarrow E(p_1) > E(p_2)$
  - $C(p_1+p_2) > C(p_1) + C(p_2) \Rightarrow E(p_1+p_2) > E(p_1) + E(p_2)$

## Modularitat. Mida dels mòduls

Però no es pot dividir infinitament.

Dividir implica augmentar la interdependència entre mòduls:

- Augmenta el cost associat a crear les interfícies.
- Costa més detectar l'origen dels errors i corregir-los.



Solució (criteri) per arribar a  $M \Rightarrow \uparrow$  Cohesió,  $\downarrow$  Acoblament

- Dues estratègies per dissenyar una jerarquia de mòduls (arquitectura del software):
- **Top-down.** Refinaments successius. Es comença per un nivell més abstracte i a cada pas obtenim noves components que ofereixen una visió més concreta.
- **Bottom-up.** Es comença des de les components de més baix nivell de la jerarquia i de manera progressiva establim les components de nivell més alt.
- **Estratègia mixta.** Es comença des dels dos extrems de la jerarquia fins que es troben.

## Disseny modular efectiu

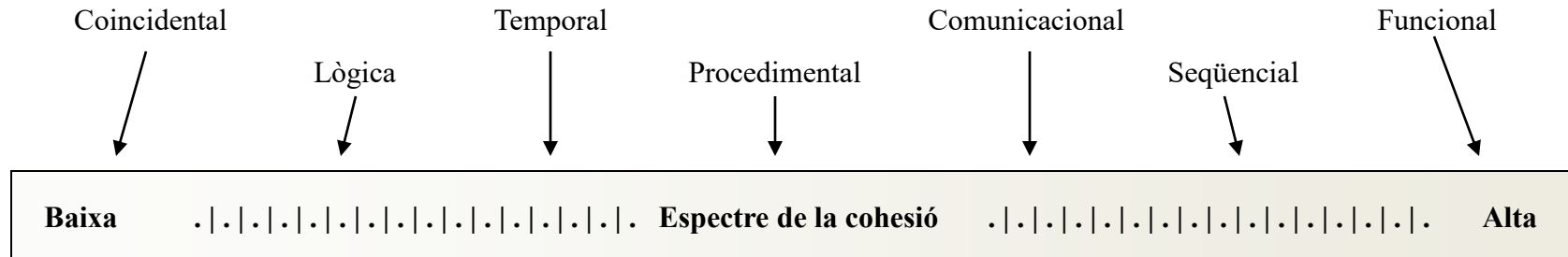
- **Independència funcional.** Un sistema es considera modular si està format per una sèrie de components de manera que cadascuna proporcioni una bona definició d'abstracció, i que un canvi introduït sobre una component tingui un impacte mínim sobre la resta.

La modularitat efectiva s'aconsegueix amb independència funcional, que vol dir mòduls amb una funció única i poca interacció entre ells.

- **Cohesió.** Cada mòdul té una funció única. És l'extensió del principi de la ocultació.
- **Acoblament.** És una mesura de la interdependència entre mòduls.

**Objectiu:** màxima cohesió i mínim acoblament.

$\uparrow \text{Cohesió} \Rightarrow \downarrow \text{Acoblament}$



- **Coincidental.** Tasques poc relacionades. Per exemple, partir un programa arbitràriament en mòduls cada x línies.
- **Lògica.** Tasques relacionades lògicament. Per exemple, un mòdul amb totes les funcions d'entrada/sortida.
- **Temporal.** Tasques que s'han d'executar durant el mateix període de temps. Per exemple, alliberar memòria, tancar fitxers, etc.
- **Procedimental.** Partició a partir del diagrama de flux, és a dir, elements relacionats que s'executen en un ordre específic.
- **Comunicacional.** Tasques que operen sobre les mateixes dades.
- **Seqüencial.** Les dades resultat de cada element de processament del mòdul són l'entrada del següent element.
- **Funcional.** Tots els elements del mòdul es relacionen per realitzar una única funció.

# Cohesió

## COINCIDENTAL

```
for (i=0; i<maxi; i++)
    printf(" %d ", x[i]);
if (k!=l) k++;
...
for (j=0; j<maxj; j++)
    printf(" %d ", y[j]);
if (k!=l) k++;
...
int X(int p, int maxp, int r, int *s, int *q)
{
```

```
    for (p=0; p<maxp; p++)
```

```
        printf(" %d ", q[p]);
```

```
    if (*s!=r) (*s)++;
```

```
}
```

## TEMPORAL

### Mòdul\_Inserció\_CD

```
Buscar_info_id()
```

```
Llegir_pistes()
```

```
Buscar_última_posició()
```

## LÒGICA

### Mòdul\_Vigilància

```
Detecció_moviment()
```

```
Gravar()
```

```
Trucar_policia()
```

```
Disparar_alarma()
```

```
Programar_alarma()
```

## COMUNICACIONAL

### Mòdul\_Compra

```
Consulta_dades(id_prod)
```

```
Afegir_carro(id_prod)
```

```
Actualitzar_stock(id_prod)
```

```
Suggerir_relac(id_prod)
```

## PROCEDIMENTAL

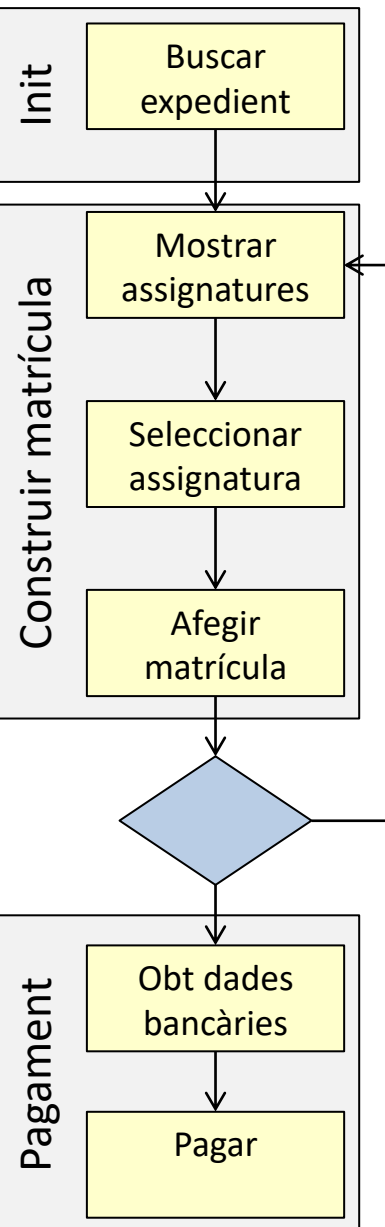
## SEQÜENCIAL

### Mòdul\_Navegació

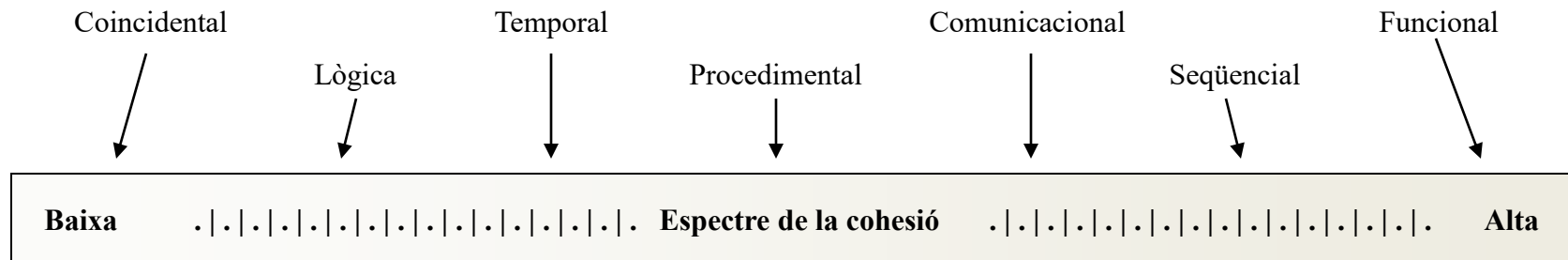
```
r=Buscar_recorr(or,dest)
```

```
t=Calc_temps(r)
```

```
h=Calc_hora_arrib(t)
```

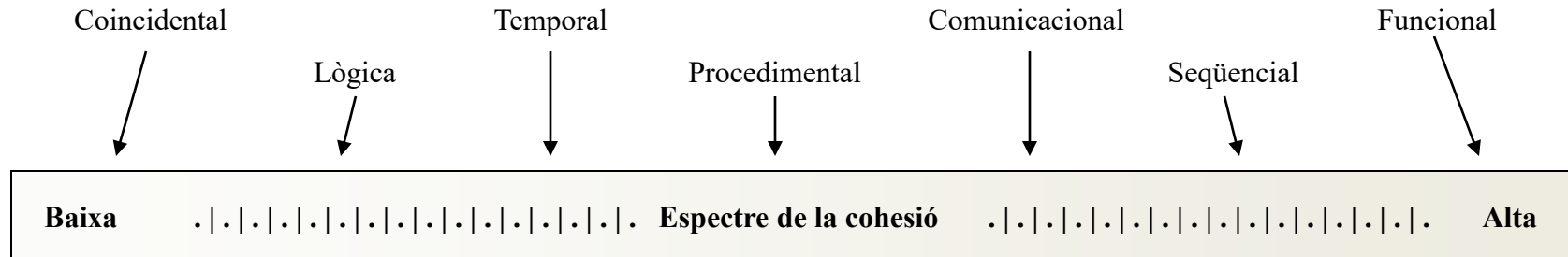


## Quin nivell de cohesió té un mòdul que ... ?



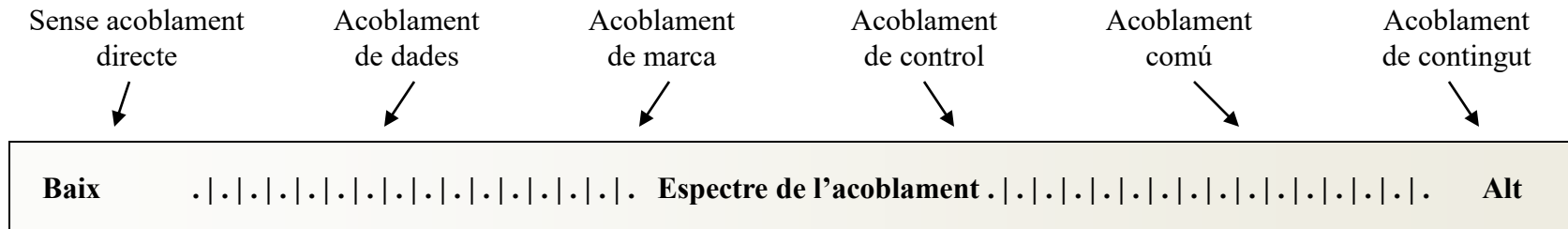
- Quin nivell de cohesió té un mòdul que agrupa les següents funcions referents a tasques relacionades amb gestió comercial: *veure\_catàleg\_productes()*, *pagar\_bonus\_a\_comercial()*, *acumular\_descomptes\_compra()*, *alta\_proveïdor()*, *rebre\_curriculums\_candidats\_comercials()*, *consultar\_projectes\_obra\_social()*.
- Quin nivell de cohesió té un mòdul que agrupa les següents funcions encadenades: *e=buscar\_expedient\_alumne(DNI)*, *e2=afegir\_notes(e)*, *gravar\_expedient(e2)*.
- Quin nivell de cohesió té un mòdul que agrupa les següents funcions de finalització enquesta: *bloquejar\_formulari\_votació()*, *mostrar\_estadística\_participació()*, *fer\_recompte()*.
- Quin nivell de cohesió té un mòdul que agrupa les següents funcions referents a tasques relacionades amb gestions professor: *entrar\_notes()*, *reservar\_aules()*, *consultar\_full\_nòmina()*, *demanar\_avaluació\_recerca()*, *validar\_targeta\_pàrking()*.
- Quin nivell de cohesió té un mòdul que agrupa les següents funcions referents a tasques relacionades amb gestió de música: *reproduir\_mp3()*, *fer\_llista\_de\_reproducció()*, *recomanar\_cançons\_semblants()*, *Subscriure\_Spotify()*, *Imprimir\_partitura()*, *Compartir\_audio\_xarxes\_socials()*.
- Quin nivell de cohesió té un mòdul que agrupa les següents funcions que operen sobre la dada d'entrada *c* (correu en edició): *enviar\_correu(c)*, *gravar\_correu\_a\_enviats(c)*, *refrescar\_finestra\_edicio\_correu(c)*.
- Quin nivell de cohesió té un mòdul que agrupa aquestes funcions després d'una detecció d'intrusió en un sistema de vigilància: *activar\_so\_alarma()*, *bloquejar\_porta()*, *encendre\_llum()*.

## Quin nivell de cohesió té un mòdul que ... ?



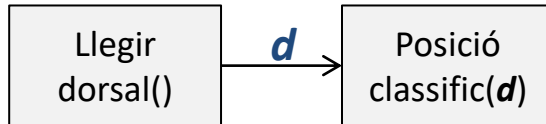
- Lògica** • Quin nivell de cohesió té un mòdul que agrupa les següents funcions referents a tasques relacionades amb gestió comercial: *veure\_catàleg\_productes()*, *pagar\_bonus\_a\_comercial()*, *acumular\_descomptes\_compra()*, *alta\_proveïdor()*, *rebre\_curriculums\_candidats\_comercials()*, *consultar\_projectes\_obra\_social()*.
- Seqüencial** • Quin nivell de cohesió té un mòdul que agrupa les següents funcions encadenades: *e=buscar\_expedient\_alumne(DNI)*, *e2=afegir\_notes(e)*, *gravar\_expedient(e2)*.
- Temporal** • Quin nivell de cohesió té un mòdul que agrupa les següents funcions de finalització enquesta: *bloquejar\_formulari\_votació()*, *mostrar\_estadística\_participació()*, *fer\_recompte()*.
- Lògica** • Quin nivell de cohesió té un mòdul que agrupa les següents funcions referents a tasques relacionades amb gestions professor: *entrar\_notes()*, *reservar\_aules()*, *consultar\_full\_nòmina()*, *demanar\_avaluació\_recerca()*, *validar\_targeta\_pàrking()*.
- Lògica** • Quin nivell de cohesió té un mòdul que agrupa les següents funcions referents a tasques relacionades amb gestió de música: *reproduir\_mp3()*, *fer\_llista\_de\_reproducció()*, *recomanar\_cançons\_semlants()*, *Subscriure\_Spotify()*, *Imprimir\_partitura()*, *Compartir\_audio\_xarxes\_socials()*.
- Comunicacional** • Quin nivell de cohesió té un mòdul que agrupa les següents funcions que operen sobre la dada d'entrada *c* (correu en edició): *enviar\_correu(c)*, *gravar\_correu\_a\_enviats(c)*, *refrescar\_finestra\_edicio\_correu(c)*.
- Temporal** • Quin nivell de cohesió té un mòdul que agrupa aquestes funcions després d'una detecció d'intrusió en un sistema de vigilància: *activar\_so\_alarma()*, *bloquejar\_porta()*, *encendre\_llum()*.



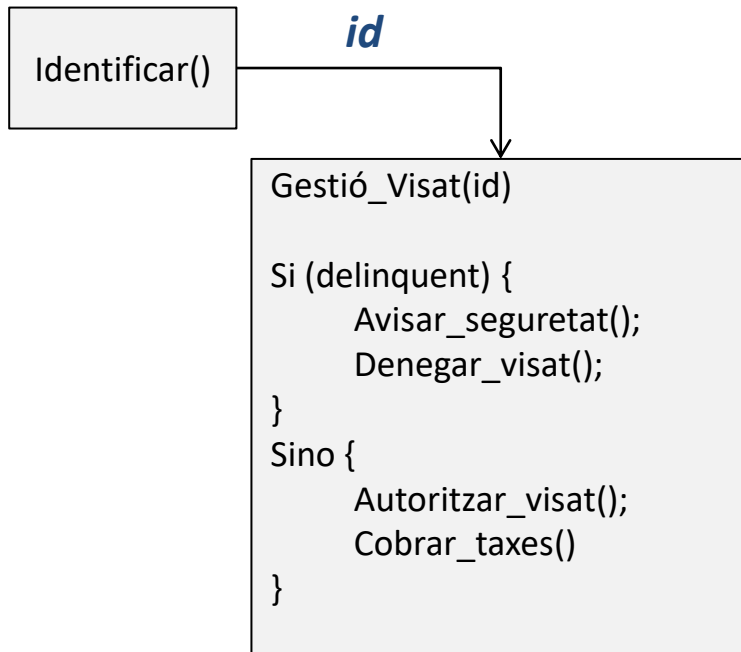


- **Acoblament de dades.** Dos mòduls es comuniquen amb paràmetres on cadascun és una dada simple o bé un conjunt de dades homogènies.
- **Acoblament de marca (per registre).** Un mòdul passa a un altre un registre o estructura de dades amb no homogeneïtat (amb diferents camps). Cal una comprovació de les estructures de dades. Per alta banda, cal evitar passar registres amb molts camps dels quals només se n'utilitza algun.
- **Acoblament de control.** Un mòdul passa un paràmetre a un altre amb intenció de controlar el seu comportament. És més acceptable que el mòdul inferior doni un control al superior (per exemple, notificació d'error).
- **Acoblament comú.** Diversos mòduls accedeixen a la mateixa àrea global de dades.
  - Qualsevol error en un mòdul pot afectar a un altre via dades globals.
  - Difícil seguir el programa pel que fa a qui modifica les dades globals.
  - Manteniment difícil si es canvia l'estructura de dades global (molts mòduls hi accedeixen).
  - Si hi ha moltes dades globals, és difícil establir quines dades utilitza cada mòdul.
- **Acoblament de contingut.** Un mòdul accedeix directament a una part de l'altre mòdul (sentències tipus *go to*).

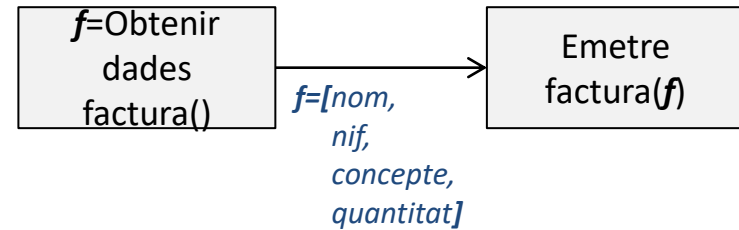
## DADES



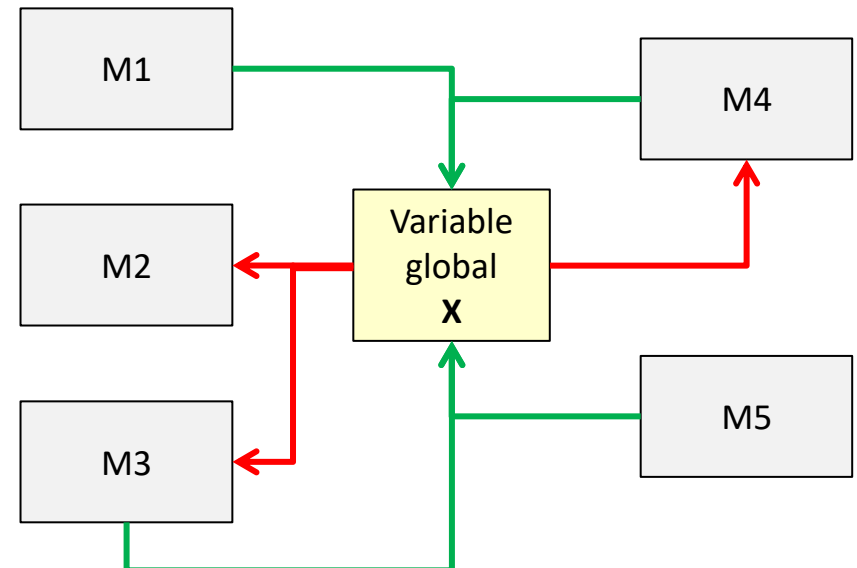
## CONTROL



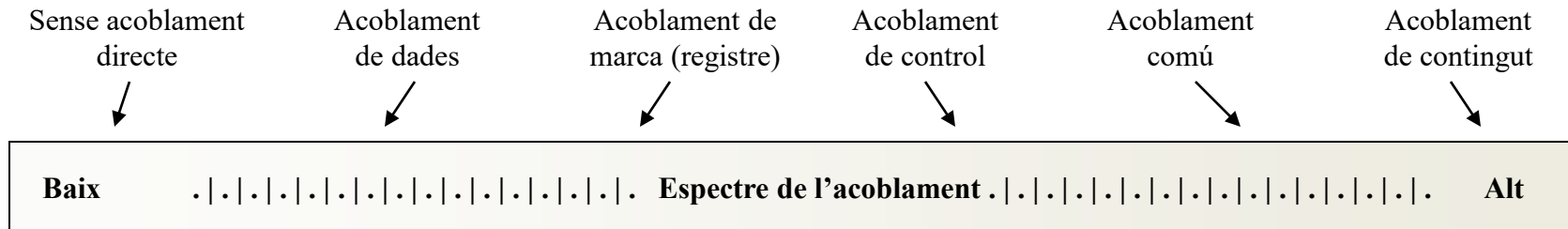
## REGISTRE



## COMÜ

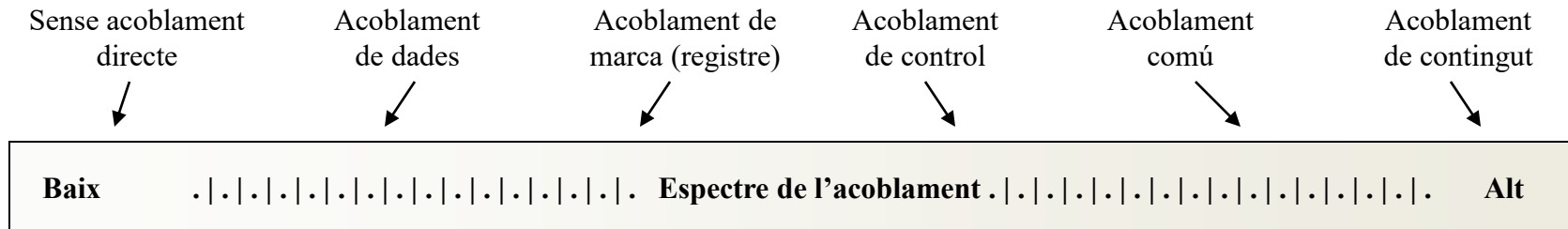


## Quin nivell d'acoblament tenen uns mòduls que ... ?



- Quin nivell d'acoblament tenen dos mòduls *Alta\_usuari* i *Compra\_producte* que es passen una variable  $r=\{\text{nom, telefon, email, num\_compte, data\_caducitat}\}$ ?
- Quin nivell d'acoblament tenen dos mòduls *Estimar\_nivell\_de\_risc\_fons\_inversió* i *Decidir\_inversions* que es passen com a paràmetre una variable *risc* que és un valor entre 0 i 100?
- Quin nivell d'acoblament tenen dos mòduls *trobat=Buscar\_Expedient* i *Mostrar\_Assignatures\_Possibles\_Matricula* on el segon mostra assignatures de 1er si *trobat*=FALS, o les assignatures que li toca a l'alumne si *trobat*=CERT?
- Quin nivell d'acoblament tenen dos mòduls *Gestió\_matrícules\_estudiants* i *Gestió\_proveïdors\_restauració*?
- Quin nivell d'acoblament tenen dos mòduls *VAL=Processar\_matrícula(VAL)* i *Gestió\_pagament\_matrícula(VAL)* on *VAL* és una variable  $\{\text{cert, fals}\}$  que en cas que la matriculació sigui correcta és *cert* perquè s'efectuï el pagament?
- Quin nivell d'acoblament tenen dos mòduls *Demandar\_taxi* i *Cercar\_taxi\_proper* que es passen una variable  $r=\{\text{id\_usuari, punt\_origen, punt\_desti, hora, tipus\_servei}\}$ ?
- Quin nivell d'acoblament tenen dos mòduls *Recopilar\_mesures\_temperatura* i *Construir\_estadistiques\_temperatura* que es passen com a paràmetre una variable *t* que és un valor de temperatura?

## Quin nivell d'acoblament tenen uns mòduls que ... ?



- Quin nivell d'acoblament tenen dos mòduls *Alta\_usuari* i *Compra\_producte* que es passen una variable  $r=\{\text{nom, telefon, email, num\_compte, data\_caducitat}\}$ ?  
**Registre**
- Quin nivell d'acoblament tenen dos mòduls *Estimar\_nivell\_de\_risc\_fons\_inversió* i *Decidir\_inversions* que es passen com a paràmetre una variable *risc* que és un valor entre 0 i 100?  
**Control**
- Quin nivell d'acoblament tenen dos mòduls *trobat=Buscar\_Expedient* i *Mostrar\_Assignatures\_Possibles\_Matricula* on el segon mostra assignatures de 1er si *trobat*=FALS, o les assignatures que li toca a l'alumne si *trobat*=CERT?  
**Control**
- Quin nivell d'acoblament tenen dos mòduls *Gestió\_matrícules\_estudiants* i *Gestió\_proveïdors\_restauració*?  
**Sense acoblament**
- Quin nivell d'acoblament tenen dos mòduls *VAL=Processar\_matrícula(VAL)* i *Gestió\_pagament\_matrícula(VAL)* on *VAL* és una variable  $\{\text{cert, fals}\}$  que en cas que la matriculació sigui correcta és *cert* perquè s'efectuï el pagament?  
**Control**
- Quin nivell d'acoblament tenen dos mòduls *Demandar\_taxi* i *Cercar\_taxi\_proper* que es passen una variable  $r=\{\text{id\_usuari, punt\_origen, punt\_desti, hora, tipus\_servei}\}$ ?  
**Registre**
- Quin nivell d'acoblament tenen dos mòduls *Recopilar\_mesures\_temperatura* i *Construir\_estadistiques\_temperatura* que es passen com a paràmetre una variable *t* que és un valor de temperatura?  
**Dades**

- Un **patró de disseny** és una estructura de classes que és reutilitzable per solucionar diferents problemes de diferents àmbits. Per exemple, un patró *Iterador* que té les funcions per moure's a través d'una llista d'objectes.
- Un **patró arquitectònic** és una solució general sobre l'estructura de les components del software i reutilitzable per problemes comuns dins de contextos donats. Els patrons arquitectònics són similars als patrons de disseny però tenen un abast més ampli.

## Exemples de patrons arquitectònics

---

- **En capes.** Per a aplicacions que es poden estructurar a diferents nivells d'abstracció. Cada capa dona servei a la següent. Capes més freqüents:
  - Capa de presentació/interfície.
  - Capa d'aplicació o de servei.
  - Capa de la lògica del negoci o de domini.
  - Capa d'accés a les dades o persistent.
- **Client-Servidor.** La component del servidor dona servei a diverses components client. Per exemple, una aplicació de correu.
- **Pipeline** (arquitectura en canonada). Components enllaçades seqüencialment, quan el software processa dades que es transformen linealment. Per exemple, un compilador que enllaça l'anàlisi lèxica, sintàctica i la generació de codi.
- **Model-vista-controlador.** Separa les interfícies de la gestió de la funcionalitat bàsica. Afavoreix la reutilització.
  - Model. Conté la funcionalitat i les dades bàsiques.
  - Vista. Interacció amb l'usuari (hi pot haver més d'una vista).
  - Controlador. Gestiona l'entrada de l'usuari.

## Disseny d'interfícies d'usuari (GUI – Graphical User Interface)

- Medi de comunicació usuari  $\leftrightarrow$  màquina
- La interfície ha d'estar ben feta perquè modela la percepció que l'usuari té del software  $\rightarrow$  satisfacció de l'usuari
  - Ha de millorar l'eficiència
  - Evitar frustració de l'usuari (cometre errors)
  - El software s'ha d'adaptar al humà, no a l'inrevès
- Aspectes principals:
  - Regles d'aurades
  - Mecanismes d'interacció (per implantar les regles)

## Regles daurades (principis fonamentals)

---

### 1. Deixar el control a l'usuari

- No obligar a l'usuari a fer accions no desitjades: donar feedback, tolerar errors (permetre desfer accions – *undo*)
- Permetre interrupcions, ocultar tecnicismes

### 2. Reduir la necessitat de memoritzar

- Reduir la memoria a curt termini, definir dreceres (*shortcuts*)
- Mostrar informació de forma progressiva (general → concret)
- Jerarquia visual (usuari s'ha de focalitzar en allò important a cada moment)

### 3. Fer una interfície senzilla i consistent

- Tot el SW ha de seguir les mateixes regles de disseny, clares i concises
- No fer canvis entre models interactius si no és necessari
- Reutilitzar patrons ja coneguts (e.g. Facebook, Twitter, ...)



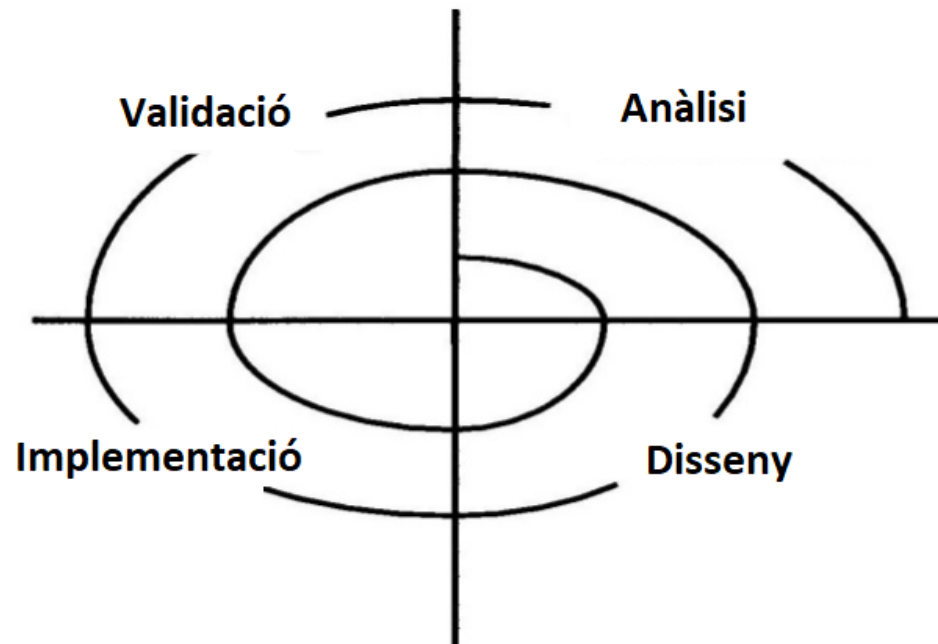
Bjarne Stroustrup (creador del llenguatge C++)

"Sempre he desitjat una computadora que sigui tan fàcil d'utilitzar com el meu telèfon. El meu desig es va fer realitat. Ja no sé com utilitzar el meu telèfon".

- Els usuaris volen aplicacions fàcils d'emprar
  - El SW ha d'ajudar a fer la feina, que no confongui i provoqui errors
- Usabilitat:

Mesura com el SW facilita l'aprenentatge, ajuda a l'usuari a recordar, redueix la possibilitat de cometre errors, augmenta l'eficiència
- Per saber si el SW és “usable” s'ha d'avaluar amb usuaris:
  - Cal aprenentatge? Evita i preveu errors?
  - Interfície està estructurada de forma lògica i consistent?
  - S'entenen els botons i icones?
  - L'usuari sap en tot moment on és? (estat del sistema)

- Model en Espiral (iteratiu)



- Anàlisi de la interfície
  - Quins perfils d'usuaris (entrevistes, anàlisi de mercat...)
  - Quines tasques a fer (casos d'ús)
  - Anàlisi del flux de treball
  - Anàlisi del contingut de la pantalla i format (estètica)
  - Ambient de treball
- Disseny
  - Definir objectes i accions
  - Distribució de la pantalla (disseny gràfic, col.locació d'icones, descripció del text)
  - Escenari d'ús

- Etapes
  - Definir objectes i accions (operacions)
  - Definir quines accions de l'usuari (esdeveniments) canvien l'estat de la interfície
  - Il·lustrar cada estat de la interfície (com es veuria)
    - Distribució de la pantalla (disseny gràfic, col·locació d'icones, descripció del text)
- Patrons de disseny
  - Abstracció que descriu un problema de disseny ben delimitat, ja solucionat anteriorment i que es pot reutilitzar (ex. Calendari giratori)
  - S'evita reinventar la roda

- Aspectes del disseny
  - Temps de resposta (massa alt en eines molt interactives)
  - Eines d'ajuda (què, quan i com)
  - Gestionar els errors (información útil i constructiva)
  - Llegendes del menú i comandes (ex. Ctrl+C)
  - Accessibilitat de l'aplicació
  - Internacionalització (idioma, codificació text)

- Principis
  - Previsió, Comunicació, Consistència, Eficiència, Flexibilitat, Aprenentatge, Reducció de la latència, ús de metàfores...
- Propietats desitjables
  - Disseny *responsive*
    - Redimensionar i col·locar els elements de la web de forma que s'adaptin a la resolució (i amplada) de cada dispositiu.
    - S'aconsegueix una correcta visualització i millor experiència d'usuari
    - El disseny responsive permet reduir el temps de desenvolupament i evita continguts duplicats
  - Adaptable a mòbils



- Recomanacions
  - Evitar gran quantitat de text (es sol llegir més lent en pantalla que en paper)
  - Evitar vincles a pàgines “en construcció”
  - Informació important al davant (usuaris prefereixen no emprar scroll)
  - Menus de navegació i capçaleres consistents
  - L'estètica no ha d'obstaculitzar la navegabilitat (millor icona senzilla que imatge 3D)
  - Opcions de navegació òbvies per a l'usuari (software intuitiu)

## Avaluació del disseny de la interfície

---

- Avaluació del prototipus → Feedback de l'usuari
- S'empren tècniques formals d'avaluació (ex. questionaris) per a avaluar la complexitat i usabilitat de la interfície
  - Aspectes qualitatius (resposta si/no, escala numérica)
  - Aspectes quantitatius:
    - Nombre de tasques realitzades i temps emprat per a cada tasca
    - Temps que ha estat mirant l'ajuda
    - Nombre d'errors comesos
- Es fa una altra volta a l'espiral (anàlisi, disseny...)
- El cicle continua fins que la interfície és satisfactòria



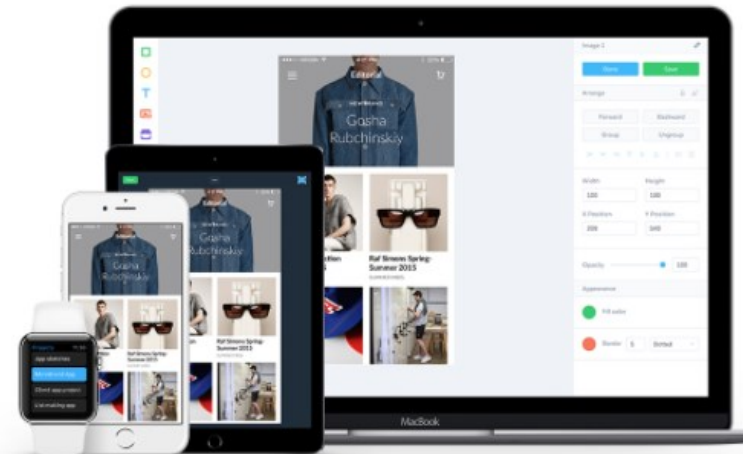
Faciliten la creació de prototipus:

- Rocket LegaSuite
- Altia
- Marvel
- ...



Simple design,  
prototyping and  
collaboration

Get started, it's free!



# Interacció Humà-Màquina (HCI): més enllà de les GUI (Graphical User Interface)

## Interfícies gestuals



## Interfícies afectives (social robots)



© Club Robòtica UAB

© Jibo



## Interfícies auditives



© Google



© Amazon



© Apple

[https://www.youtube.com/watch?v=2V4dN\\_7Kzpg](https://www.youtube.com/watch?v=2V4dN_7Kzpg)

# Interacció Humà-Màquina (HCI): més enllà de les GUI (Graphical User Interface)

---

Interfícies sensorials (ex. eye tracking)

## The Little prince

Antoine de Saint Exupéry



### Chapter 1

Once when I was six years old I saw a magnificent picture in a book, called True Stories from Nature, about the primeval forest. It was a picture of a boa constrictor in the act of swallowing an animal. Here is a copy of the drawing.

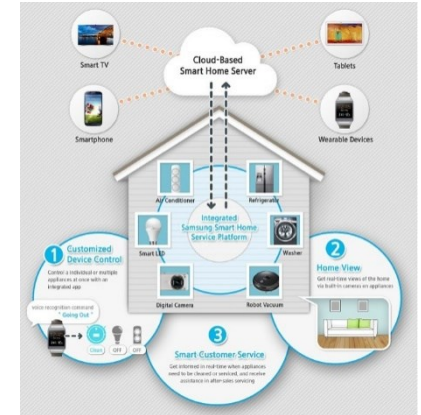
# Interacció Humà-Màquina (HCI): més enllà de les GUI (Graphical User Interface)

## Interfícies tangibles



© Disney Research

## Interfícies ubiqües (IoT)



© Samsung Smart Home

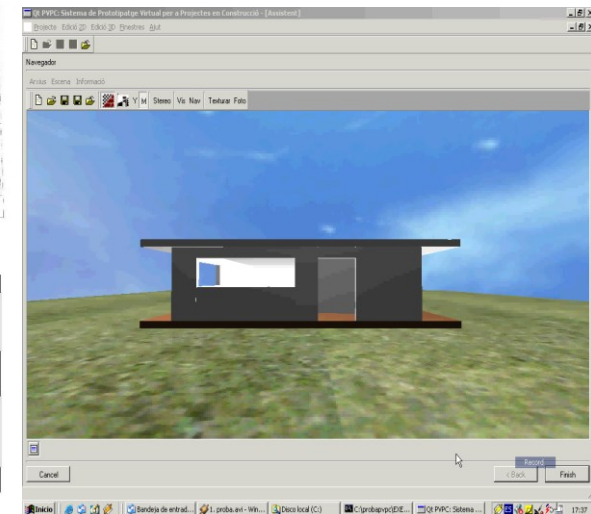
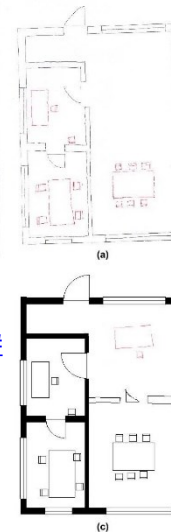
## Interfícies intel·ligents basades en llenguatge natural



© Apple



<https://quickdraw.withgoogle.com/#>



## Sketching interfaces