GRAU D'ENGINYERIA INFORMÀTICA

PROGRAMACIÓ II

Bloc 1:

Mòdul i abstracció de dades (1)

Sergio Sayago (basat en material de Laura Igual)
Departament de Matemàtiques i Informàtica
Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

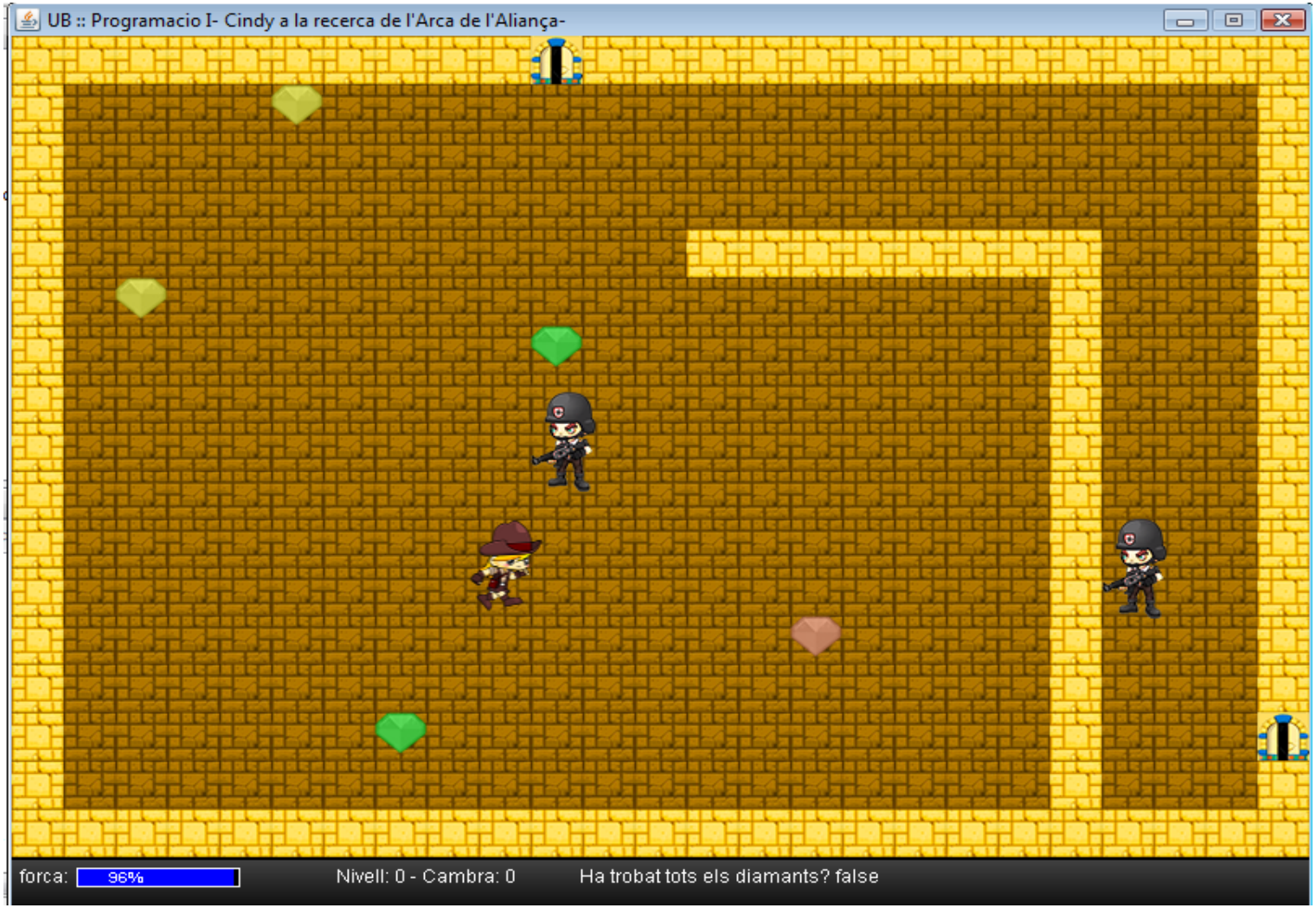
Temari Teoria

Bloc 1: Concepte de mòdul i abstracció de dades

Bloc 2: Programació orientada a objectes

Bloc 3: Programació orientada a esdeveniments

Una pregunta per començar...com es programa això?



Bloc 1: **Mòdul i abstracció de dades**

1. **Introducció**
2. Complexitat intrínseca de les aplicacions
3. Descomposició de problemes complexos
4. Factors de qualitat
5. Modularitat
    1. Descomposició funcional
    2. Descomposició basada en objectes

Introducció

- Anys 50,
  - Apareixen els primers llenguatges de programació
    - FORTRAN 1957 (FORmula TRANslation) -> alternativa al llenguatge assemblador
    - LISP (LISt Processor) 1958 -> es basa en llistes (enllaçades)
    - ...
  - Programes petits utilitzant assaig i error

...introducció

- Finals dels anys 60:
  - Hardware més potent
  - Augmenta la dificultat dels problemes abordats
  - Sistemes grans sense estructuració
  - Impossibles de mantenir
  - Molt costosos i escassament fiables

- Necessitat d'ordenar i detallar el camí de resolució del problema abans de començar a programar.

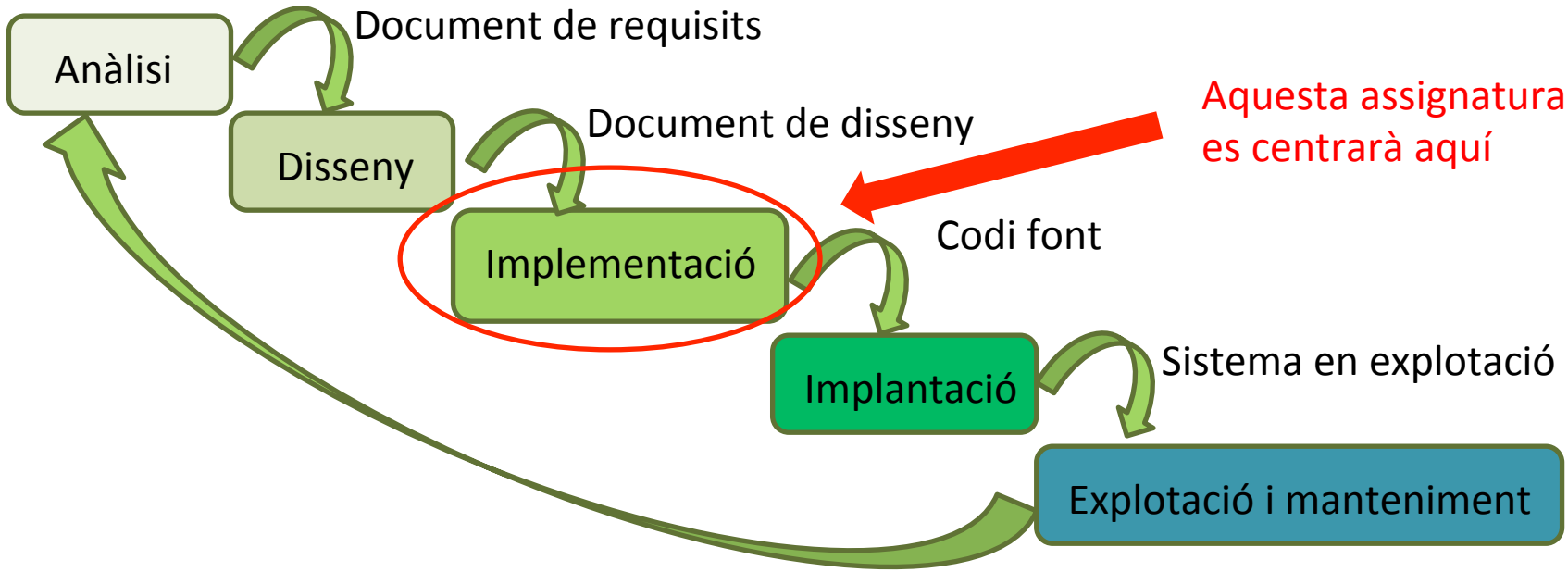
- Crisis del Software...apareix **l'Enginyeria del Software**
  ... evoluciona fins avui en dia

Introducció

- L'objectiu general de la **Enginyeria del Software** és produir software de qualitat
- Per qualitat s'entén l'adequació del software als requisits exigits
- El camí per a obtenir software de qualitat és mitjançant un plantejament rigorós del problema

Introducció

- Etapes del cicle de vida del software:



El **Cicle de vida del software o procés de desenvolupament de software** és aquell en el que les necessitats de l'usuari són traduïdes en requisits de software, aquestos transformats en disseny i el disseny implementat en codi. Després ve la implantació i explotació que porta de nou a l'anàlisi.

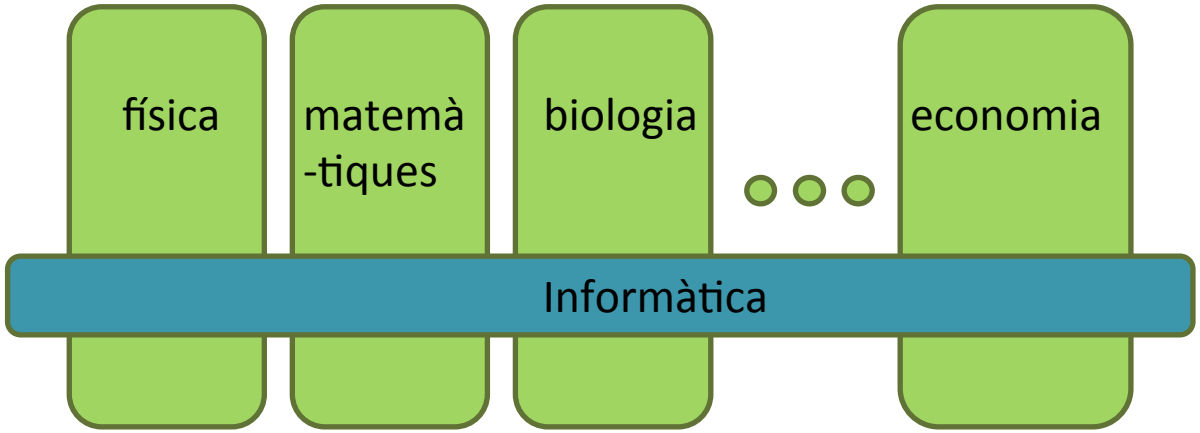
Bloc 1: **Mòdul i abstracció de dades**

1. Introducció
2. **Complexitat intrínseca de les aplicacions**
3. Descomposició de problemes complexos
4. Factors de qualitat
5. Modularitat
    1. Descomposició funcional
    2. Descomposició basada en objectes

Complexitat intrínseca de les aplicacions informàtiques

1. El domini del problema.

- La informàtica és una disciplina transversal
- Tracta problemes de caràcter interdisciplinari



A més,

- Dificultat en transmetre necessitats i expectatives dels usuaris
- Canvis en els requisits durant el desenvolupament
  - Per exemple: requisits d'un sistema de navegació aèria

Complexitat intrínseca de les aplicacions informàtiques

2. Sistemes grans.
   - Pot haver desenes de desenvolupadors implicats que, a més, poden estar separats geogràficament
   - Problemes de comunicació i coordinació
3. Parts comunes entre aplicacions són, molt sovint, difícilment reutilitzables.
4. Dificultat per caracteritzar el comportament de sistemes discrets.
   - Nombre molt gran d'estats, per això les proves no són completes.

Bloc 1: **Mòdul i abstracció de dades**

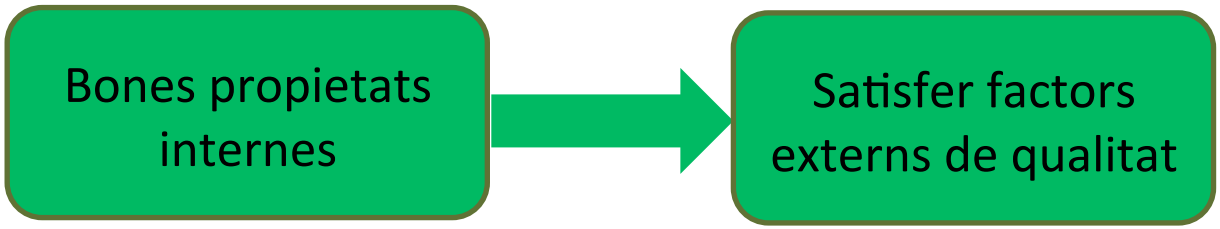
1. Introducció
2. Complexitat intrínseca de les aplicacions
3. **Descomposició de problemes complexos**
4. Factors de qualitat
5. Modularitat
    1. Descomposició funcional
    2. Descomposició basada en objectes

Descomposició de problemes complexos

- No podem fer que la dificultat desaparegui, però, podem desenvolupar tècniques i eines que ens permetin **tractar-la i gestionar-la.**
  - Amb aquesta finalitat, definim **factors de qualitat del programari externs i interns**.

Descomposició de problemes complexos

- **Factors externs:**
  - Poden ser detectats pels usuaris
  - Qualitat externa es la que realment preocupa
- **Factors interns:**
  - Només són percebuts pels dissenyadors i programadors
  - Mitjà per aconseguir la qualitat externa
- Objectiu:



Bloc 1: **Mòdul i abstracció de dades**

1. Introducció
2. Complexitat intrínseca de les aplicacions
3. Descomposició de problemes complexos
4. **Factors de qualitat**
5. Modularitat
    1. Descomposició funcional
    2. Descomposició basada en objectes

(alguns) factors de qualitat externs

- **Correcció**
- Robustesa
- Extensibilitat
- Reutilització
- Compatibilitat
- Eficiència
- Portabilitat
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Correcció:

  Capacitat dels productes de software per a realitzar amb exactitud les seves tasques tal i com es defineixen en les especificacions.

  "El REQ FUNC 5 no està implementat"

Factors de qualitat

- Correcció
- **Robustesa**
- Extensibilitat
- Reutilització
- Compatibilitat
- Eficiència
- Portabilitat
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Robustesa:

  Capacitat dels sistemes de reaccionar apropiadament davant condicions excepcionals.

  Es va la llum...però el programa fa "autosave" (penseu en Google Docs)

Factors de qualitat

- Correcció
- Robustesa
- **Extensibilitat**
- Reutilització
- Compatibilitat
- Eficiència
- Portabilitat
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Extensibilitat:

  Facilitat d'adaptar els productes de software als canvis o ampliacions d'especificació.

  Els (bons) programes informàtics evolucionen al llarg del temps

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- **Reutilització**
- Compatibilitat
- Eficiència
- Portabilitat
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Reutilització:

  Capacitat dels elements de software de servir per a la construcció de diferents aplicacions.

  Penseu en la finestra (concretament, diàleg) de Guardar Como

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- Reutilització
- **Compatibilitat**
- Eficiència
- Portabilitat
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Compatibilitat:

  Facilitat de combinar uns elements de software amb altres.

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- Reutilització
- Compatibilitat
- **Eficiència**
- Portabilitat
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Eficiència

  Capacitat d'un sistema software per exigir la menor quantitat possible de recursos hardware, tals com temps del processador, espai ocupat de memòria interna i externa o ample de banda utilitzat en els dispositius de comunicació

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- Reutilització
- Compatibilitat
- Eficiència
- **Portabilitat**
- Facilitat d'us
- Funcionalitat
- Oportunitat

Factors de qualitat

- Portabilitat

  Facilitat de transferir els productes software a diferents entorns hardware i software

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- Reutilització
- Compatibilitat
- Eficiència
- Portabilitat
- **Facilitat d'us**
- Funcionalitat
- Oportunitat

Factors de qualitat

- Facilitat d’us

  Facilitat amb la qual persones amb diferents formacions i aptituds poden aprendre a utilitzar els productes software i aplicar-los a la resolució de problemes.

  També cobreix la facilitat d'instal·lació, d’operació i de supervisió.

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- Reutilització
- Compatibilitat
- Eficiència
- Portabilitat
- Facilitat d'us
- **Funcionalitat**
- Oportunitat

Factors de qualitat

- Funcionalitat

  Conjunt de possibilitats que proporciona un sistema.

Factors de qualitat

- Correcció
- Robustesa
- Extensibilitat
- Reutilització
- Compatibilitat
- Eficiència
- Portabilitat
- Facilitat d'us
- Funcionalitat
- **Oportunitat**

Factors de qualitat

- Oportunitat

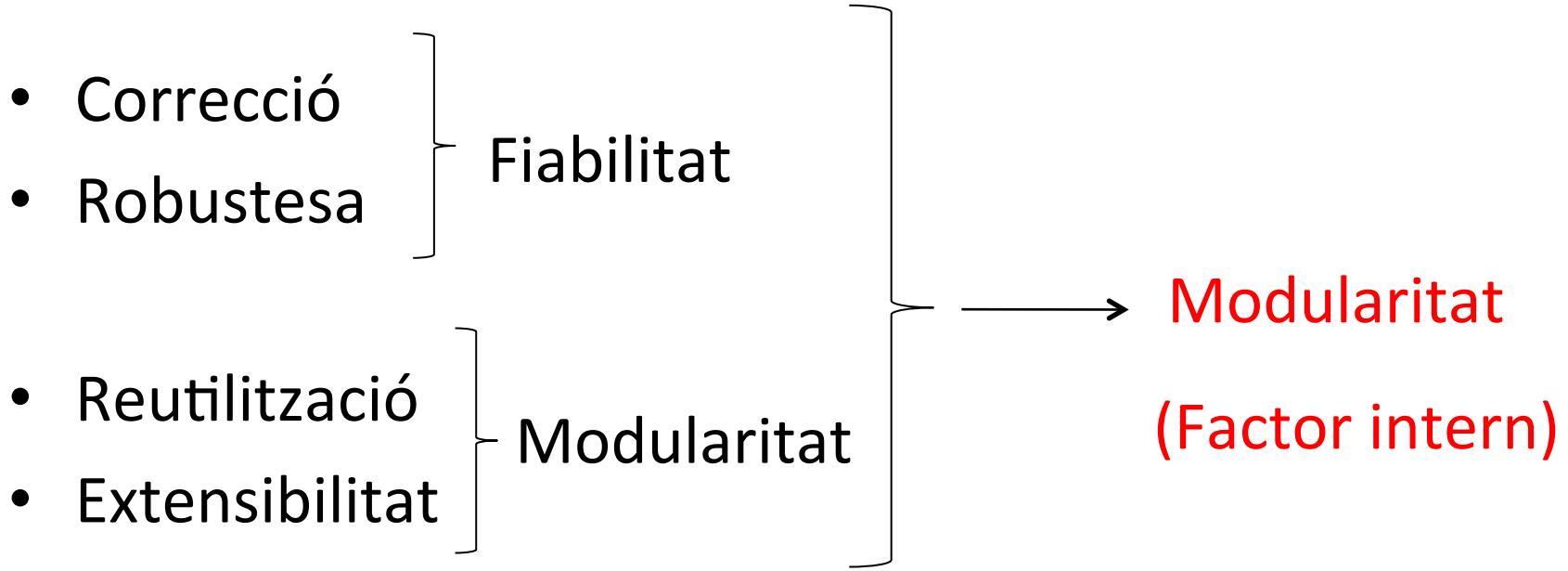
  Capacitat d'un sistema de software de ser llançat quan els usuaris ho desitgen o abans.

Factors de qualitat: reutilització MOLT important

- Millorar la **reutilització** implica millorar quasi tota la resta de qualitats
- Beneficis esperats:
  - Oportunitat: *podem desenvolupar més ràpidament*
  - Fiabilitat: *parts ja provades -> menys errors*
  - Eficiència: *parts ja provades -> més eficients*
  - Inversió: *"justificació" econòmica*
  - Consistència: *prendre com a model un codi / estil*

Factors de qualitat

Qualitats claus:



Bloc 1: **Mòdul i abstracció de dades**

1. Introducció
2. Complexitat intrínseca de les aplicacions
3. Descomposició de problemes complexos
4. Factors de qualitat
5. **Modularitat**
    1. Descomposició funcional
    2. Descomposició basada en objectes

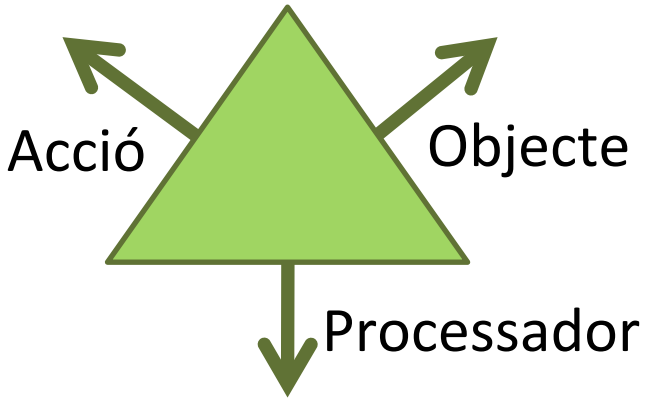
Introducció a la Modularitat

- Objectius principals a l'hora de fer disseny del software:
  1. Fiabilitat
  2. Modularitat

- Requereixen mètodes sistemàtics de descomposició dels sistemes en **mòduls**

Modularitat

- Quins criteris s'han d'utilitzar per trobar els mòduls del nostre software?
  - Les 3 forces de la computació:



En resum: "executar un sistema de software és utilitzar certs processadors per aplicar certes operacions a certs objectes."

Acció → què fa un sistema
Objecte → a qui li ho fa

- Mòduls com:
  - Unitats de descomposició funcional
  - Unitats basades en els principals tipus d'objectes

Modularitat

- Diferència:

  – **Els enfocaments tradicionals (paradigma de programació procedimental)** construeixen cada mòdul sobre alguna **unitat de descomposició funcional**, un cert aspecte de l'acció.

  – **L'enfocament (paradigma) orientat a objectes** construeix cada mòdul al voltant **d'algun tipus d'objecte.**

Modularitat

- Element clau:

  Continuïtat modular

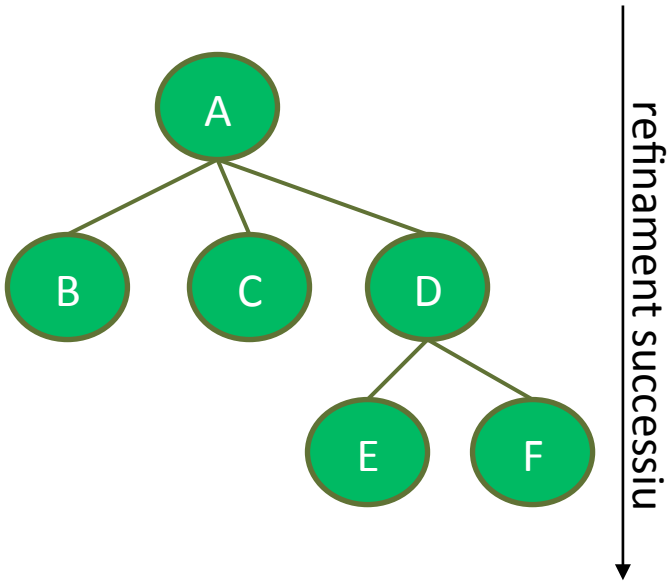
  – Un mètode de disseny satisfà aquest criteri si ens proporciona **arquitectures estables**.

    • Mantenen la quantitat de canvi en el disseny proporcional a la mida dels canvis en l'especificació (que s'ha definit prèviament).

  – Important, si es considera l'evolució del sistema a llarg termini.

Descomposició funcional

- Disseny descendent:
  - Construeix un sistema per **refinament successiu**.
  - Pot veure's com el desenvolupament en forma d'un arbre
- Procés:
  - Comença expressant un enunciat de la funció al nivell més alt d'**abstracció**.
  - Continua amb una seqüència de passos de refinament, reduint el nivell d'abstracció
  - Descompon cada operació en una combinació d'una o més operacions més senzilles

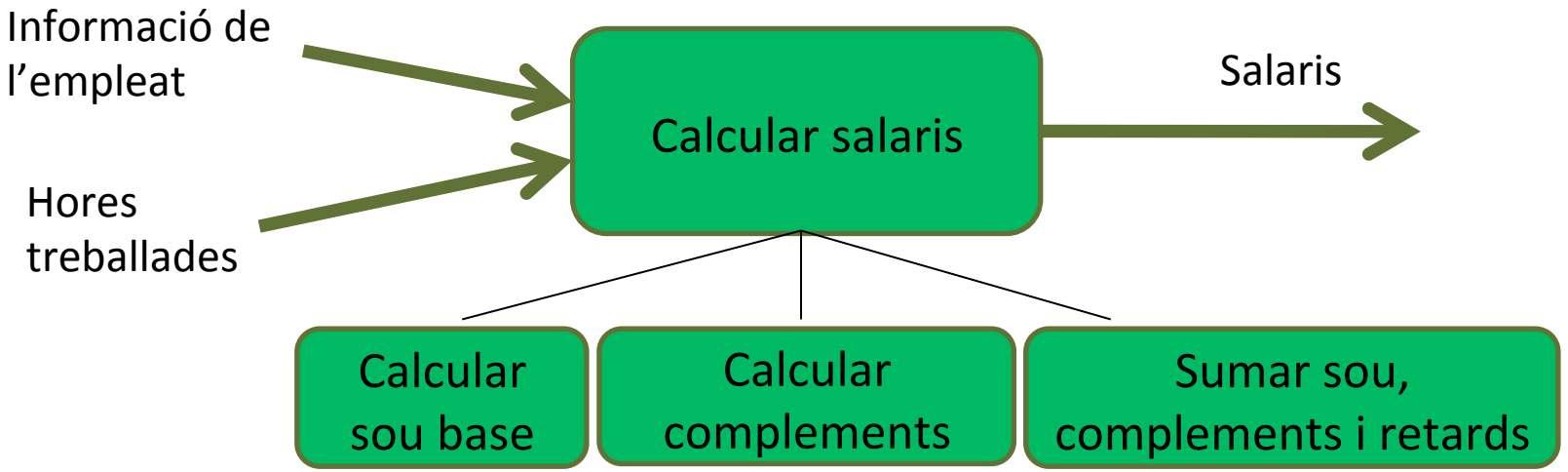


Exemples

- Penseu en els exercisis que heu fet a Programació I
- Frase monovocàlica...

Més exemples

- Sistema de nòmines



Si canviem l'objectiu i en lloc de només calcular el salari volem:

- Extreure algunes estadístiques addicionals.
- Alguns empleats es pagaran mensualment però altres trimestralment!
- El personal vol poder accedir a l'aplicació interactivament!

Pros i Contres de la descomposició funcional

- Avantatges:
  - Disciplina de pensament lògic i ben organitzada
  - Es pot ensenyar amb eficàcia
  - Promou el desenvolupament ordenat de sistemes
  - Ajuda al dissenyador a trobar un camí a través de la complexitat que sovint presenten els sistemes en les seves etapes inicials de disseny.
  - Pot ser útil per a petits programes i algorismes individuals

Pros i Contres de la descomposició funcional

- Limitacions quan tractem un problema gran:
  - Assumir que el sistema té només una funció principal (el "cim")
  - Base de la descomposició modular: propietats subjectes a canvi
  - Èmfasis prematur en restriccions d'ordre

Descomposició basada en objectes

- Podem trobar una caracterització més estable d'un sistema?
  - Exemple: sistema de nòmines

    → Tipus d'objectes manipulats pel sistema.

    L'esquema orientat a objectes definiria:
    - Tasques de l'empresa
    - Persona
    - Contracte
    - Aplicació

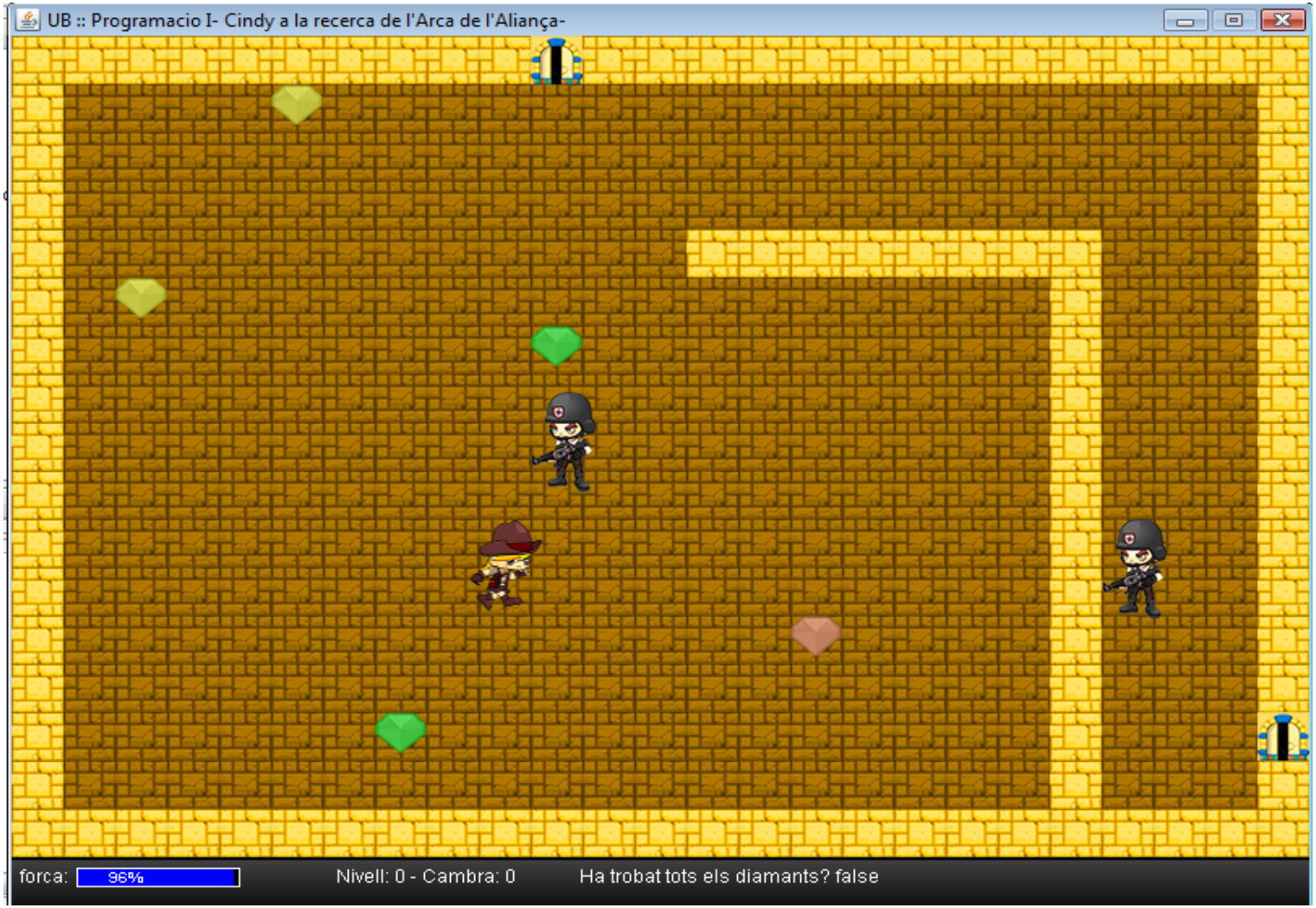
Construcció del software orientat a objectes

- La construcció del software orientat a objectes és el mètode de desenvolupament de software que basa l'arquitectura de qualsevol sistema de software en mòduls deduïts dels tipus abstractes d'objectes que manipula.

No pregunti primer **què** fa el sistema:
pregunti a **qui** ho fa!

Procés ascendent

Un altre exemple



Lectures recomanades

- Capítol 1 del llibre de Bertrand Meyer, "**Construcción de software orientado a objetos", Prentice Hall, 1998.**

- **"No Silver Bullet Essence and Accidents of Software Engineering"** Computer Magazine by Frederick P. Brooks. (April 1987)