Assignació de Responsabilitats a Capes

- Problemàtica
- Transició de l'especificació al disseny. Etapes
- Punt de partida
- Casos d'Ús concrets
- Assignació de responsabilitats a capes
- Patró Boundary-Control-Entity
- Especificació de cada capa
- Què queda per fer?
- Realització de Casos d'Ús
- Bibliografia

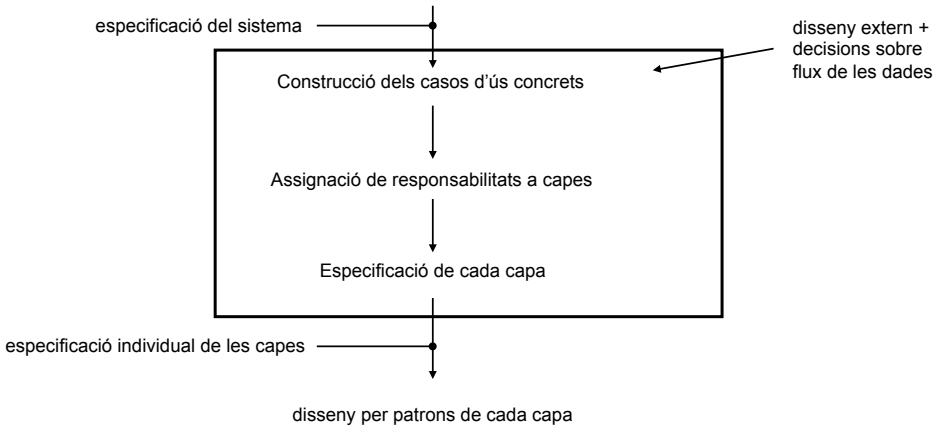
Problemàtica

- Els models de l'especificació:
  - defineixen els conceptes del món real (domini del problema)
  - veuen el sistema software com una caixa negra
    - ✓ existeix un únic component que modelitza el sistema software sencer
  - no es consideren els requisits no funcionals
    - ✓ en particular, de comunicació amb l'usuari ni els tecnològics
  - són independents de la tecnologia
    - ✓ exploten al màxim la capacitat expressiva del llenguatge
- Els models del disseny:
  - defineixen els conceptes que es desenvoluparan per proporcionar una solució a les necessitats del món real (domini de la solució)
  - veuen el sistema software com una caixa blanca
    - ✓ distingeixen els components del patrò arquitectònic escollit
    - ✓ identifiquen els elements de cada component i la seva interacció
  - consideren els requisits no funcionals
    - ✓ en particular, de comunicació amb l'usuari
  - són depenents de la tecnologia
    - ✓ han d'adaptar-se a les restriccions pròpies de la tecnologia escollida

Transició de l'especificació al disseny

- En conseqüència, necessitem:
  - passar del món del problema al món de la solució
    - ✓ identifiquem i apliquem patrons arquitectònics i de disseny
    - ✓ considerem necessitats de disseny (e.g., control dels errors)
  - assignar les responsabilitats dels casos d'ús d'especificació als elements de disseny
    - ✓ assignació de responsabilitats a capes o components
    - ✓ les operacions ofertes per cada capa han de realitzar les responsabilitats
  - les propietats de l'arquitectura han de satisfer els requisits no funcionals
    - ✓ hem elegit canviabilitat, eficiència, portabilitat i reusabilitat
    - ✓ els patrons escollits ho han de soportar
    - ✓ es consideren els requisits de comunicació amb l'usuari
  - apropar els models d'especificació a la tecnologia
    - ✓ disseny de la comunicació amb actors externs (interfície de l'usuari, serveis)
    - ✓ adaptació al marc de la tecnologia escollida:
      - ✓normalització (orientació a objectes)
      - ✓disseny d'esquemes de bases de dades relacionals

Etapes inicials de la fase del disseny

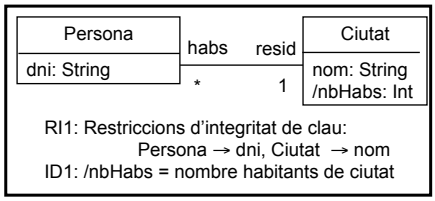


Punt de partida: especificació

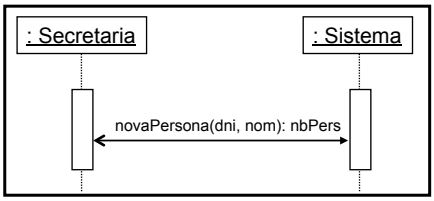
cas d'ús Nova Persona
actors Persona, Secretaria
curs principal
1. La *Persona* comunica el seu dni i ciutat de residència a la *Secretaria*
2. La *Secretaria* introdueix el dni i la ciutat en el *Sistema*
3. El *Sistema* enregistra les dades
4. El *Sistema* mostra el nombre de persones existents

extensions
dni-existeix: Ja existeix alguna persona amb el dni donat (2)
1. El *Sistema* avisa a la *Secretaria* que ja existeix la persona
2. Acaba el cas d'ús

ciutat-no-existeix: No existeix cap ciutat amb el nom donat (2)
1. El *Sistema* avisa a la *Secretaria* que no existeix la ciutat
2. Acaba el cas d'ús



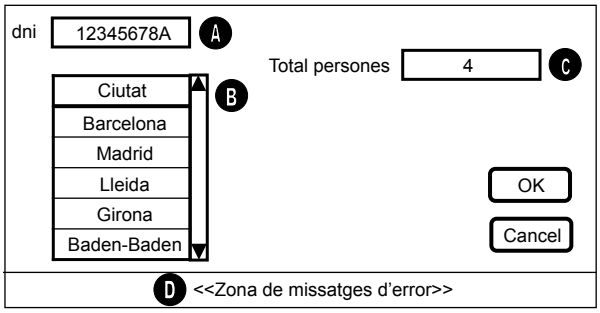
RI1: Restriccions d'integritat de clau:
Persona → dni, Ciutat → nom
ID1: /nbHabs = nombre habitants de ciutat



context novaPersona(dni: String, nom: String): Enter
-- dóna d'alta la persona amb el *dni* i la ciutat *nom*
pre: 1.1 existeix la ciutat *nom*
post: 2.1 dóna d'alta la instància de persona amb *dni*
post: 2.2 associa la persona amb la ciutat *nom*
post: 2.3 result = nombre de persones del sistema

Punt de partida: Disseny extern de la interfície

- S'identifiquen els elements d'interacció per la comunicació actor – sistema
  - camps, desplegables, zones de missatge, etc.
  - els bategem per referir-nos-hi des dels casos d'ús concrets
- Identificar els elements interns de control de la interacció (1 per cas d'ús)
- No ens preocupem de temes de disseny gràfic
  - colors, formes, etc.
  - zooms, scrolling, minimitzar, etc.



Construcció dels casos d'ús concrets
Canvis al model del comportament

- Cal veure com canvien els models creats a l'especificació al considerar el disseny extern
  - operacions que apareixen, altres que són substituïdes, etc.
  - cada esdeveniment de presentació es correspon a una operació en el diagrama de seqüència
  - els paràmetres de les operacions indiquen el flux d'informació actors – sistema
- Com a resultat, en el disseny poden aparéixer noves versions de:
  - els diagrames de seqüència
  - els contractes

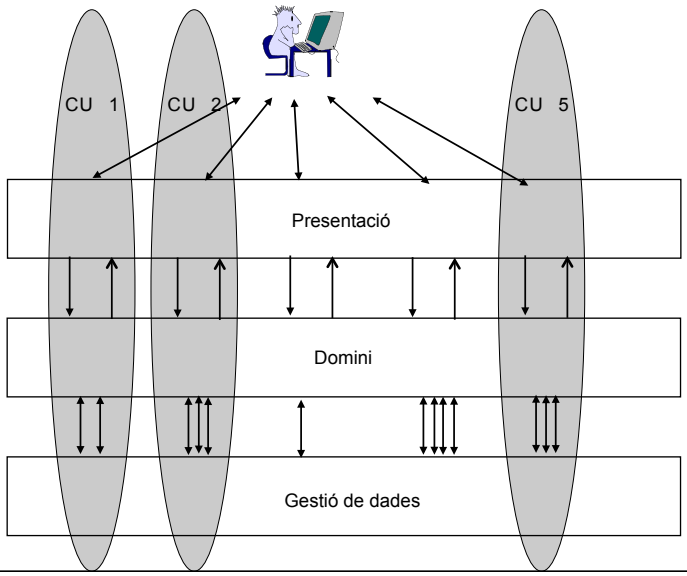
Assignació de responsabilitats a capes

- Responsabilitats: (pre i post) condicions dels contractes (o accions dels casos d'ús) + informació del model de dades (o conceptual)
  - Cal assignar cada responsabilitat a una o més capes
- Típicament:
  - Capa de presentació: gestió dels elements del disseny extern
  - Capa de domini: transformacions, càlculs, ...
  - Capa de gestió de dades: actualitzacions, consultes i obtenció d'elements
- La repartició de responsabilitats entre les capes de domini i de gestió de dades depèn del *patró predominant* a la capa de domini
  - *Domain Model / Transaction Script*
- Algunes responsabilitats no es tractaran directament en el mètode que implementa l'operació, poden estar delegades a altres components
  - Principalment, alguns invariants del sistema (restriccions d'integritat)
  - Exemples típics: tractament de claus en el SGBD, selecció de valors correctes a la interfície d'usuari

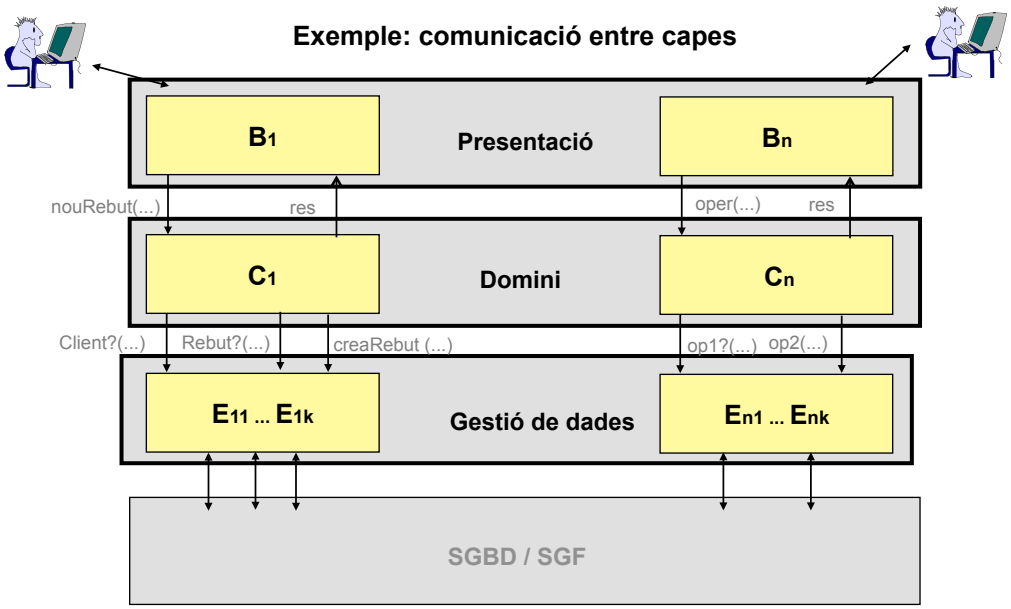
Especificació de cada capa

- Cal identificar les operacions i atributs (si és el cas) de cada capa.
  - Equival a considerar cada capa com una caixa negra
  - Especifiquem els serveis o operacions oferts entre capes
  - Especifiquem l'estat del cas d'ús com a atributs de la capa
  - Ajuda a reduir l'acoblament entre capes
- L'assignació de responsabilitats entre les capas es tradueix amb les operacions i atributs definits en cada capa.
- L'assignació de responsabilitats ve determinada pels casos d'ús (i contractes) del sistema:
  - Distribució de responsabilitats de cada cas d'ús a totes les capes
  - Ens cal dins de cada capa, algun element que reculli les responsabilitats de cada cas d'ús dins de cada capa

Exemple: comunicació entre capes n CASOS d'ÚS



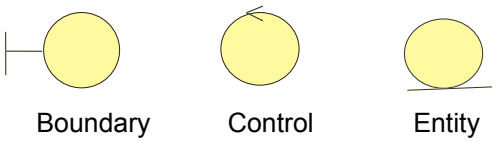
Exemple: comunicació entre capes



Patró Entity-Control-Boundary

- En la definició del comportament d'un sistema, d'un escenari de cas d'ús, etc, els diferents elements que el composen poden estar alineats a una d'aquestes perspectives o rols:

– **Boundary**
– **Control**
– **Entity**



- El comportament d'un sistema o d'un cas d'ús es pot descriure amb aquestes tres perspectives o rols
- És una descripció inicial dels elements del sistema o cas d'ús i del seu comportament
- Cal, de totes formes, refinar aquesta descripció amb la realització d'un disseny detallat d'aquests elements i del seu comportament, tenint en compte patrons, criteris de qualitat, recursos tecnològics disponibles, paradigmes de desenvolupament, etc

Patró Entity-Control-Boundary: Boundary

- L'element (o classe) Boundary és un element del sistema que s'encarrega de la gestió de la frontera del sistema o subsistema amb els actors externs.
- La classe Boundary assumeix les responsabilitats de:
  - Front- End: s'encarrega de recollir i processar els inputs i peticions que es reben de components externs.
  - Back-End: s'encarrega de gestionar la comunicació amb els components de suport externs als que fem algun tipus de petició.
- Exemples:
  - Front-End:
    - Formulari entrada dades
    - Interfície d'usuari, impresora, dispositiu extern
    - Interfície de serveis oferts a l'exterior
  - Back-End:
    - Sistema d'autorització pagament amb tarja crèdit
    - Interfície d'un sistema extern

Patró Entity-Control-Boundary: Control

- L'element (o classe) Control és un element del sistema que s'encarrega de la gestió del flux de la interacció interna del sistema o subsistema (cas d'ús).
- La classe Control assumeix les responsabilitats de:
  - donar resposta a les peticions d'elements Boundary (Front-End)
  - colaborar amb altres elements Entity, Control i Boundary (Back-End) per donar aquesta resposta
- Exemples:
  - Controladors de la Capa de Domini
  - Classe que gestiona la interacció complexa en una Capa de Presentació
  - Classe que gestiona l'accés a la Base de Dades

Patró Entity-Control-Boundary: Entity

- L'element (o classe) Entity és un element generalment persistent i passiu del sistema que gestiona conjunts d'informació significativa del sistema o subsistema (cas d'ús).

- La classe Entity assumeix les responsabilitats de:
  - enmagatzemar les dades relatives a un concepte o informació,
  - proveïr funcions per a gestionar el comportament/evolució d'aquestes dades
  - validar aquestes dades
  - proveïr funcions per a realitzar petits càlculs a partir d'aquestes dades

- Exemples:
  - Classe del model conceptual d'especificació que esdevindrà persistent amb els mètodes de tractament, accés i càlcul necessaris.

Patró Boundary-Control-Entity: Comportament

- El comportament es descriu amb diagrames d'interacció (diag. Sequència)
- Un escenari d'execució del Cas d'Ús s'inicia fora del sistema.
- Les Boundaries (Front-End) detecten aquest inici.
- Les responsabilitats de l'execució del Cas d'Ús (o contracte) es distribueixen entre els diferents elements del sistema segons els rols anteriors.
- Els elements del sistema (Boundary, Control, Entities) interaccionen adequadament entre ells per a satisfer aquestes responsabilitats.
- Les interaccions entre elements venen delimitades per la següents regles:
  - Elements amb elements del mateix tipus (Excepte Boundary)
  - Control pot interactuar amb Entities i Boundaries
  - Entities i Boundaries no poden interactuar entre elles directament

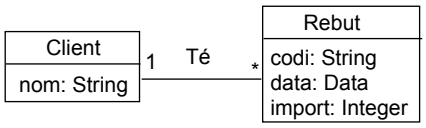
	Boundary	Control	Entity
Boundary		X	
Control	X	X	X
Entity		X	X

Patró Boundary-Control-Entity: VOPC

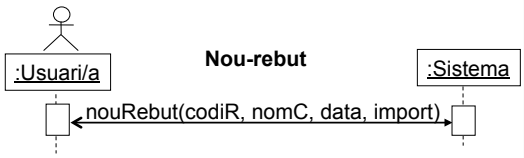
- VOPC (View of Participant Classes)
- Es un diagrama de classes específic per a cada Cas d'Ús.
- S'acostuma a realitzar previ a la definició del comportament amb la interpretació de l'especificació del Cas d'Ús (o operacions de sistema). Es completa un cop fet l'anàlisi de comportament (associacions, atributs, operacions, ...)
- S'hi representen el conjunt de classes que participen en la realització del cas d'ús i la relació entre elles.
  - Les classes representades són les que apareixen en el diagrama d'interacció (diag. seqüència) amb atributs i operacions
  - Les relacions entre elles poden tenir la multiplicitat i la navegabilitat necessària per la realització del cas d'ús.

- El conjunt de VOPCs de tot el sistema (els de tots els Casos d'Ús) donarà lloc al primer esbós de diagrama de classes de disseny:
  - Diagrama de classes de Capa Presentació (Boundaries)
  - Diagrama de classes de Capa Domini (Controladors)
  - Diagrama de Classes de Capa de Gestió de Dades (Entities)

Exemple: models de partida

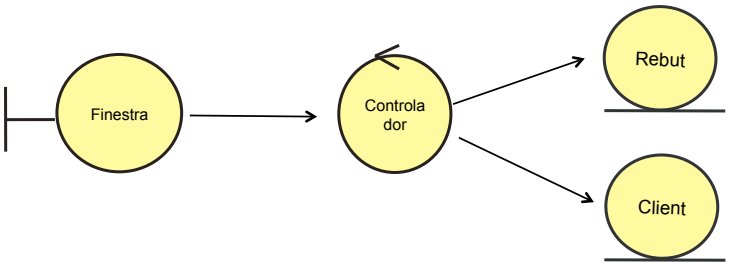


Claus de les classes: (Client, nom); (Rebut, codi)

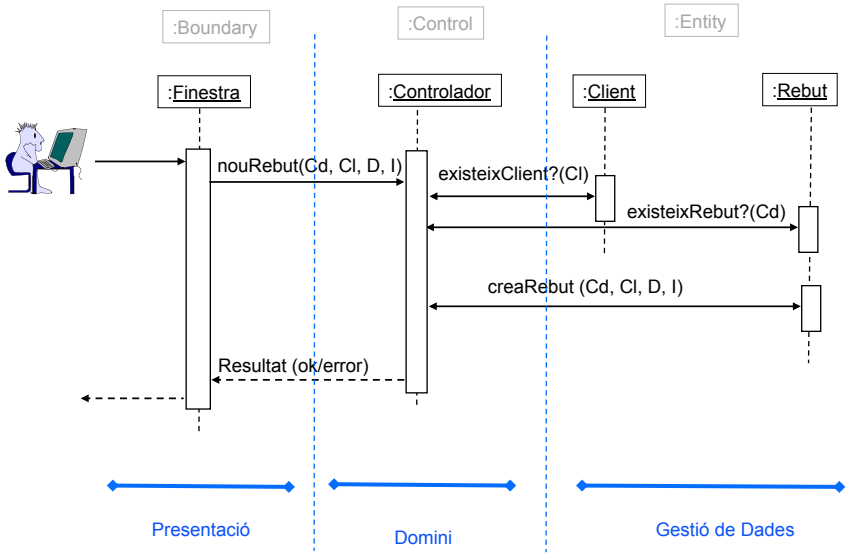


context nouRebut (codiR: String, nomC: String, data: Data, import: Integer)
-- enregistrar un nou rebut per un client
pre: 1.1 existeix el client amb nomC
post: 2.1 alta d'una instància de rebut R amb R.codi=codiR, R.data=data i R.import=import
2.2 alta d'una instància de l'associació 'Té' que associa el rebut R i el client amb nom=nomC.

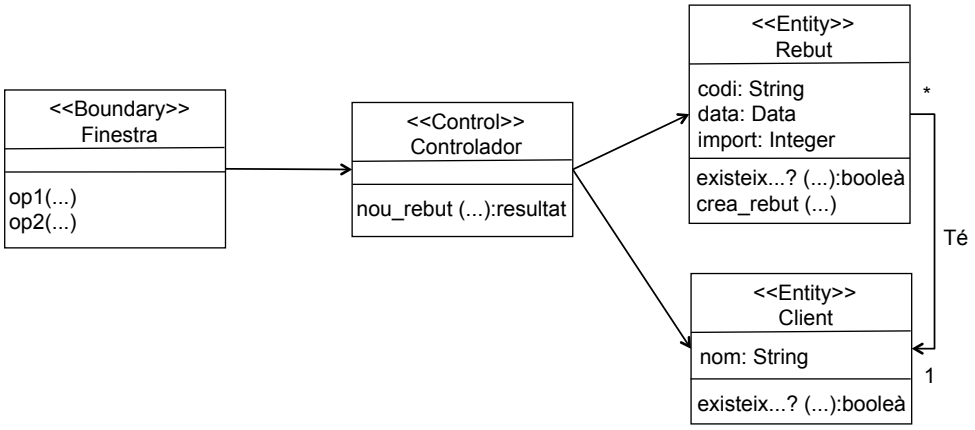
Exemple: VOPC inicial



Exemple: Diagrama Seqüència



Exemple: VOPC detallat



Especificació de cada capa
Capes individuals

- Capa de presentació:
  - ✓ s'identifiquen els esdeveniments de presentació
  - ✓ les operacions venen determinades pel tipus d'element de disseny extern implicat
  - ✓ la majoria de responsabilitats impliquen crides a la capa de domini
  - ✓ l'operació de creació de l'objecte Boundary pot exigir alguna operació addicional no prevista a la capa de domini (p.e. obtenir valors vàlids per un desplegable).
- Capa de domini:
  - ✓ es conserva tota operació que té responsabilitats a la capa de domini o dades
  - ✓ s'adapta la signatura a aquestes responsabilitats
  - ✓ les responsabilitats a fer es satisfan a la capa de domini o es deleguen a la capa de gestió de dades.
- Capa de dades: segons el patró predominant a la capa de domini, oferirà operacions d'una o més de les categories següents
  - ✓ operacions de consulta:
    - ❖ obtenció d'elements a partir de la seva clau
    - ❖ obtenció de totes les instàncies d'una classe
    - ❖ altres consultes
  - ✓ operacions d'actualització

Què queda per fer?

- Una vegada es disposa de l'especificació de cada capa, cal dissenyar cada capa independentment:
  - si podem, d'adalt a baix, per si detectem errors o mancances
  - apliquem patrons de disseny a cada capa
  - cada capa té les seves particularitats
    - ✓ presentació: elements del disseny extern → quins i on?
    - ✓ domini: patrons a aplicar (Domain Model / Transaction Script)
    - ✓ dades: pas del diagrama de classes a un esquema de base de dades relacional
- Realització de cada cas dús:
  - Fer el disseny de tot el sistema per unitats més petites: Cas d'Ús
  - Per cada cas d'ús fer el disseny de dalt a baix
  - Mantenir en consistència disseny entre casos d'ús

Realització de Casos d'Ús

- La realització de casos d'ús separa les responsabilitats definides pels especificadors (representat pel model de casos d'ús i els requisits del sistema) de les responsabilitats dels dissenyadors del sistema.
- La realització de casos d'ús proporciona una construcció en el model de disseny que organitza artefactes relacionats amb el cas d'ús però que pertanyen al model de disseny.
- Aquests artefactes son:
  - Diagrama de classes software
  - Diagrames d'interacció (de col·laboració o de seqüència) que descriuen el comportament del cas d'ús mitjançant la col.laboració o interacció entre objectes.

Bibliografia

- *Applying UML and Patterns*
  C. Larman
  Prentice Hall, 2005 (Tercera edició), caps. 13, 17 i 18

- Patró Entity-Control-Boundary
http://epf.eclipse.org/wikis/openup/core.tech.common.extend_supp/guidances/guidelines/entity_control_boundary_pattern_C4047897.html

- Use case realization
http://www.ibm.com/developerworks/rational/library/content/RationalEdge/aug04/5670.html

http://epf.eclipse.org/wikis/openup/practice.tech.use_case_driven_dev.base/guidances/guidelines/uc_realizations_448DDA77.html