

---

## **TEMA 6**

### **RATIONAL UNIFIED PROCESS (RUP)**

1. Introducció.
2. SW best practices.
3. Què es el RUP?.
4. Estructura estàtica del RUP.
5. Estructura dinàmica del RUP.

---

1

#### **6.1**

### Introducció

---

- Objectiu de l'enginyeria del software: Produir software de alta qualitat a baix preu
- És necessari un enfoc industrial per la producció
- Necessitem models de organització per aconseguir aquest objectiu

2

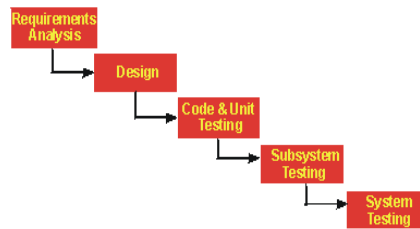
- L'èxit de la producció industrial del software implica:
  - Especialització de la producció
  - Reutilització
  - Treballs i responsabilitats organitzades en una cadena de valor
  - Adecuar les tasques a les capacitats del personal
- La crisi del software es una prova de mala organització
- Tractament = Best Practices: tècniques de desenvolupament de SW, provades comercialment, que una empresa de software ben organitzada ha de aplicar.
- Provades comercialment = Fruit de moltes experiències.

RUP desplega les 6 millors estratègies (*best practices*) del desenvolupament de software:

- Desenvolupament iteratiu del software.
- Gestió dels requeriments.
- Utilització d'arquitectures basades en components.
- Modelat visual de software.
- Verificar la qualitat del software.
- Gestió de la configuració.

## Desenvolupament iteratiu del software

- El SW ha sigut tradicionalment desenvolupat mitjançant un cicle en cascada (waterfall lifecycle), aquest tendeix a diferir l'aparició de riscos cap al final del projecte. Si aquest riscs finalment donen lloc a problemes el cost de la seva solució pot ser molt elevat, o simplement pot no haver solució.

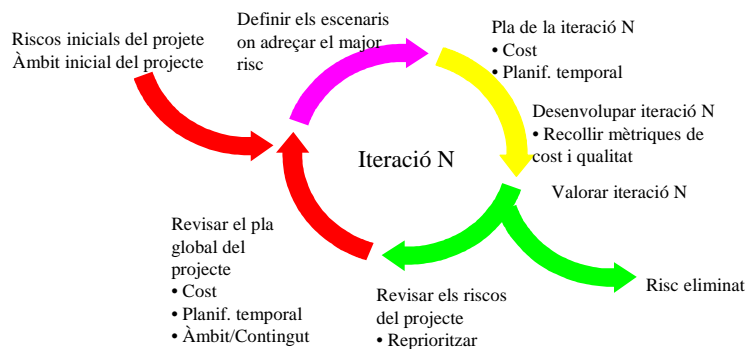


waterfall

5

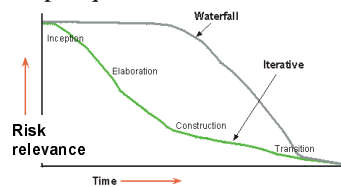
## La reducció del risc dirigeix les iteracions

- Per evitar aquest problema s'han d'utilitzar cicles de vida iteratius i incrementals, com ara el conegut model en espiral o qualsevol variant seva que conservi aquestes propietats.
- Cada iteració es basa en la construcció d'un número reduït d'escenaris que se centren primer en els riscos més importants i determinen les classes i les categories a construir en la iteració.



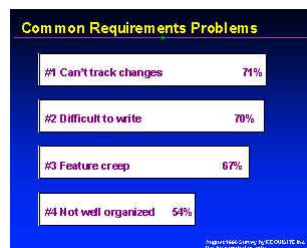
6

- Els desenvolupadors són forçats a afrontar els riscos aviat.
- Potencia el feedback constant dels usuaris
- Detecció d'inconsistències entre requeriments, disseny i codificació.
- La verificació i validació continuada dona una idea real de l'estat del projecte.
- El treball dels diferents equips (disseny, codificació, prova, etc.) es distribueix en el temps i poden utilitzar lliçons apreses.
- Admet canvis de requeriments.
- Integració progressiva d'elements.
- Admet canvis tàctics per qüestions de mercat.



7

- Requeriment: condició o capacitat que un sistema ha d'assolir.
- S'ha d'admetre que els requeriments són dinàmics, canvien durant la vida d'un projecte.
- De fet, hi ha estudis que demostren que la principal font de fracàs als projectes és la naturalesa canviant dels requisits i la inhabilitat per manejar aquesta circumstància.
- Hem de recordar que l'èxit final del projecte vindrà determinat pel nombre de requeriments assolits satisfactòriament.



(estudi fet al 1996)

8

## Gestió dels requeriments

- La gestió dels requeriments consisteix en descobrir, organitzar i documentar les necessitats reals que porta a un clients a sol·licitar el desenvolupament d'un sistema, així com mantenir un acord continu sobre els requeriments que canvien.
- Els avantatges de la gestió de requeriments són:
  - Els requeriments poden tenir una prioritat i un seguiment.
  - Les inconsistències es poden detectar més fàcilment.
  - Seguiment de l'estat del projecte.

## Utilització d'arquitectures basades en components

- Principalment, l'arquitectura del SW consisteix en el conjunt de decisions importants per seleccionar els elements estructurals que componen el sistema SW: les seves interfícies i comportament, i la col·laboració entre ells per formar els subsistemes que conjuntament donen lloc al sistema SW.
- Però, l'arquitectura del SW també fa referència a qüestions com flexibilitat davant canvis, reutilització, restriccions econòmiques i tècniques, etc.
- Definició alternativa: l'arquitectura és el que queda quan ja no es pot treure més informació si es vol entendre el sistema i saber com funciona.

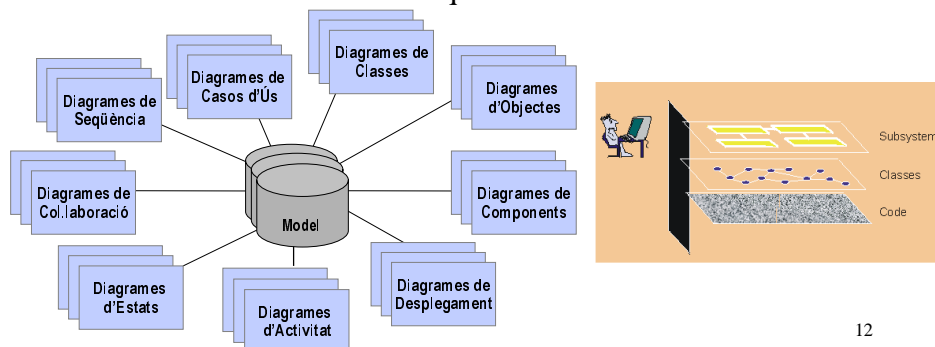
## Utilització d'arquitectures basades en components

- Per altra banda, un component SW és una peça no trivial de codi amb una interfície i funcionalitat ben definits, que es presenta encapsulat (és reemplaçable).
- En una arquitectura basada en components, els elements estructurals són els components.
- Alguns avantatges d'aquest tipus d'arquitectures són deguts a la seva modularitat que implica:
  - Facilitat per gestió de canvis.
  - Facilitat per la labor de prova del SW.
  - Suporten els cicles de vida iteratius e incrementals.
  - Reutilització de SW.

## Modelat visual de software

(més detalls veure tema anterior)

- Utilitzar llenguatges estàndard de modelatge visual com ara l'UML que permet descriure un sistema des de diferents perspectives i la comunicació no ambigua entre membres d'un equip de desenvolupament, tot això, des de un nivell d'abstracció adequat.

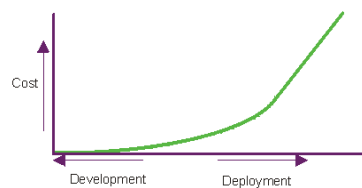


6.2

## Verificar la qualitat del software.

(més detalls al tema de Qualitat del SW, ES2)

- Concepte difícil de definir. Però relativament fàcil de detectar
- De fet, el concepte de qualitat del SW és multi-dimensional: es pot parlar de la qualitat del producte, però també del propi procés de desenvolupament, etc.
- De moment, direm que sigui com sigui que definim qualitat del SW, l'absència d'aquesta dóna lloc a problemes.
- El desenvolupament iteratiu afavoreix les proves



Els problemes del SW son de 100 a 1000 vegades mes costosos de trobat i reparar després del desplegament

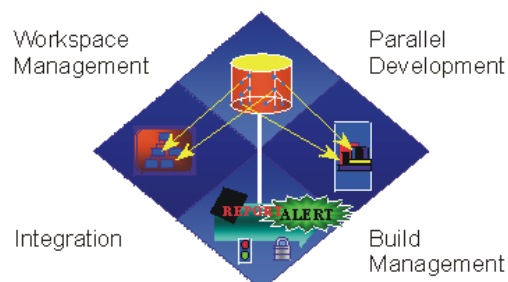
13

6.2

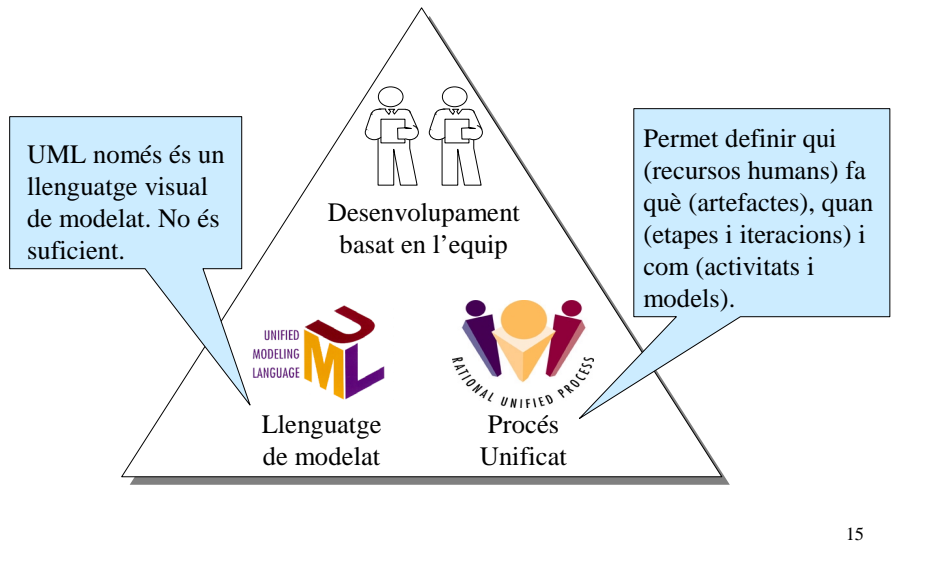
## Gestió de la configuració

(més detalls al tema de Gestió de la Configuració ES2)

- El desenvolupament de sistemes grans implica diferents equips humans treballant paral·lelament, si les seves accions no estan coordinades aviat entrem en una situació de caos.
- Els requisits canviants i el desenvolupament iteratiu e incremental també poden portar al caos si no hi ha coordinació.
- Principalment, s'han de coordinar els entorns de treball, el desenvolupament paral·lel, la integració i la gestió de les versions.



14



- Qualsevol procés de desenvolupament de SW te com a propòsits principals:
  - Establir QUI fa QUE, QUAN i COM.
  - Monitoritzar objectivament l'avanç del projecte.
- Això implica repetitivitat i predictivilitat a l'hora de fer projectes.
- Sense un procés ben definit tot es fa de manera ad hoc i, per tant, l'èxit dels projectes recau en els esforços heroics d'alguns membres de l'equip. La insostenibilitat d'això és proporcional a la mida del projecte.
- El rational unified process (RUP) és un procés pel desenvolupament de SW proposat per la companyia Rational que incorpora els coneixements anomenats best practices i està suportat per un conjunt d'eines.

## Definició d'un procés

- Un procés software ha d'especificar:
  - la seqüència d'activitats a realitzar per l'equip de desenvolupament: flux d'activitats
  - productes que han de crear-se: què i quan
  - assignació de tasques a cada membre de l'equip i a l'equip com un tot
  - proporcionar heurístiques
  - criteris per a controlar el procés

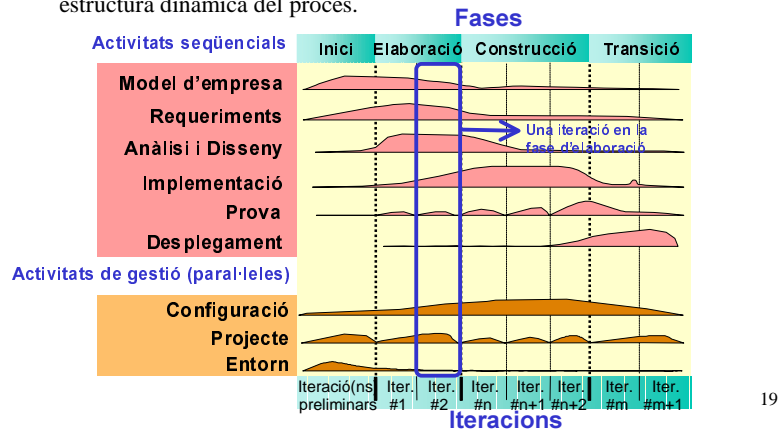
## Principis del Model Unificat

El Procés Unificat és:

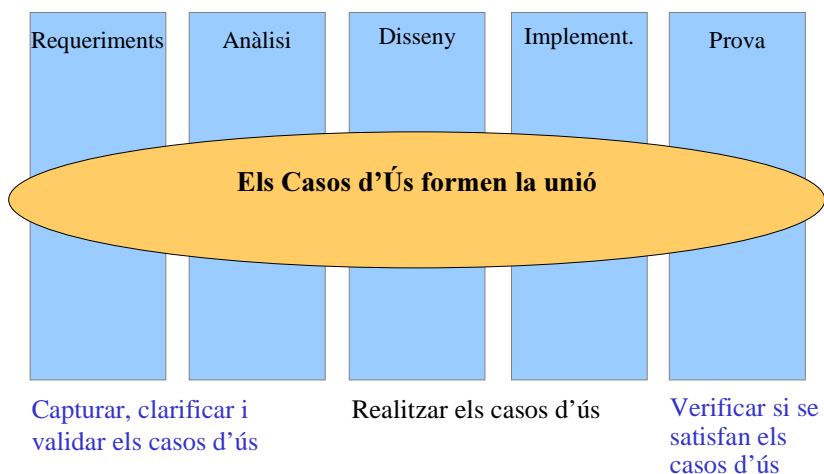
- **Iteratiu i Incremental:** el treball es divideix en iteracions petites en funció de la importància dels casos d'ús i l'estudi de riscos (un risc és una variable del projecte que el posa en perill o n'impedeix l'èxit).
- **Dirigit per casos d'ús:** des de l'especificació fins al manteniment.
- **Centrat en l'arquitectura:** reutilitzable i com a guia fins a la solució.

## Iteratiu i Incremental

- El RUP pot ser visualitzat com una estructura de dos dimensions:
  - L'eix vertical fa referència a les diferents macro-activitats involucrades en el desenvolupament del SW → estructura estàtica del procés.
  - L'eix horitzontal fa referència a l'evolució del SW al llarg del temps → estructura dinàmica del procés.



## Dirigit per casos d'ús

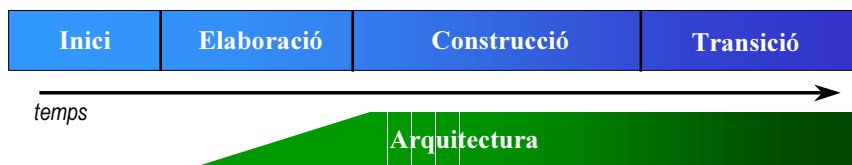


## Els casos d'ús guien les iteracions

- Guien una sèrie d'activitats de desenvolupament
  - Creació i validació de l'arquitectura del sistema
  - Definició de procediments i casos de prova
  - Planificació d'iteracions
  - Creació de la documentació de l'usuari
  - Desplegament del sistema
- Sincronitzar el contingut dels diferents models

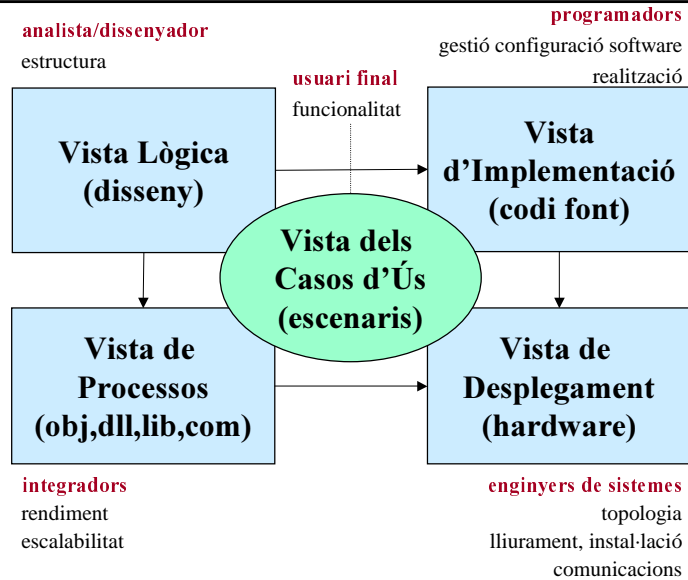
## Centrat en l'arquitectura

- Els casos d'ús especifiquen funció, l'arquitectura especifica forma (subsistemes, interfícies entre ells, distribució física, cohesió, acoblament).
- Una arquitectura ben definida des de l'inici permet que canvis al llarg del procés (nous requeriments) no siguin greus.
- Els models són vehicles per visualitzar, especificar, construir, i documentar l'arquitectura.
- El RUP guia els refinaments successius d'una arquitectura executable.
- L'arquitectura inclou una col·lecció de vistes dels models.



Es construeix fonamentalment en la fase d'elaboració.

## Una arquitectura de 4+1 vistas



## ...una arquitectura de 4+1 vieses

- **Vista de casos d'ús:** què fa el sistema per als usuaris finals.
  - Descriu el sistema com un conjunt de transaccions des del punt de vista dels actors externs.
  - Creada en la fase d'inici i guia la resta del procés.
- **Vista lògica:** domini del problema.
  - Aspectes estàtics i dinàmics d'un sistema en termes de classes i objectes.
  - Objectes, classes, col·laboracions, interaccions, paquets.
  - Creada en l'elaboració, madurada en la construcció.
- **Vista d'implementació:** organització estàtica del software.
  - Organització de les components (inclusió de classes) en l'entorn de desenvolupament.
  - Mostra el repartiment de classes en components i subsistemes.
  - Mòduls, subsistemes.
  - Creada en l'elaboració, madurada en la construcció.

### ...una arquitectura de 4+1 vistas

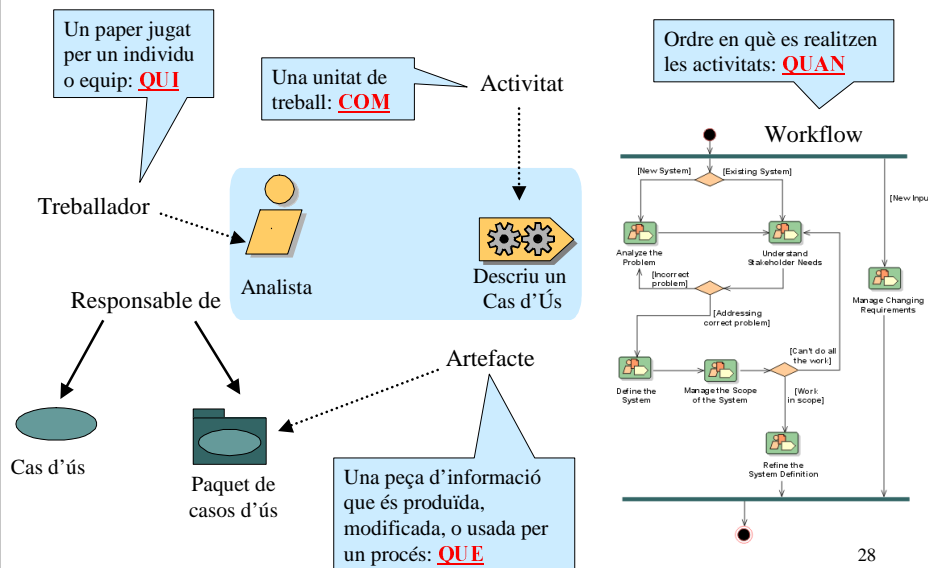
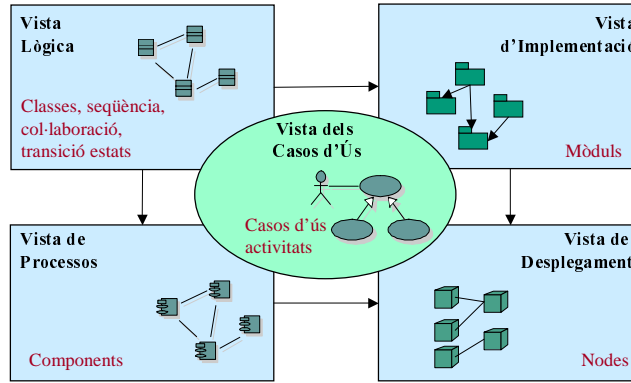
---

- **Vista de processos:**
  - Es centra en l'estructura d'implementació del sistema en temps d'execució.
  - Mostra la correspondència entre classes i llibreries en temps d'execució com applets de Java, components Active X, DLLs
- **Vista de desplegament o física: hardware.**
  - Descriu els diferents recursos de hardware i la implementació del programa en aquests recursos, considerant-ne el rendiment.
  - Nodes físics del sistema i connexions entre nodes.
  - Creada en la fase d'elaboració del procés.

### ...una arquitectura de 4+1 vistas

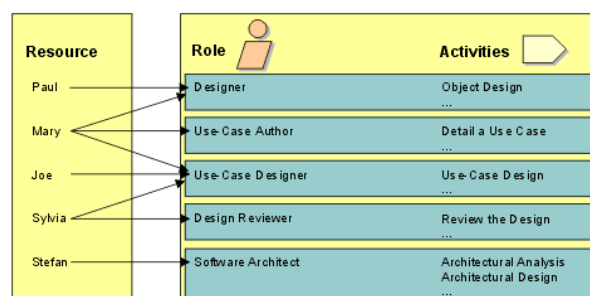
---

- **Vista d'implementació: organització estàtica del software.**
  - Organització de les components (inclusió de classes) en l'entorn de desenvolupament.
  - Mostra el repartiment de classes en components i subsistemes.
  - Mòduls, subsistemes.
  - Creada en l'elaboració, madurada en la construcció.
- **Vista de desplegament o física: hardware.**
  - Descriu els diferents recursos de hardware i la implementació del programa en aquests recursos, considerant-ne el rendiment.
  - Nodes físics del sistema i connexions entre nodes.
  - Creada en la fase d'elaboració del procés.



## Treballador (qui)

- Són rols, no individus.
- Un individu pot tenir diversos rols.
- Un rol pot ser portat a terme per diversos individus.
- Els treballadors porten a terme activitats i són responsables d'artefactes.
- Cada treballador ha de tenir associades unes habilitats especials a complir pels individus que assoleixen el rol del treballador.
- Exemples de treballadors predefinitos en RUP: change control manager, project manager, system administrator, test designer, implementer, etc.



29

## Activitats (com)

- Unitat de treball que porta a terme un individu en un rol determinat.
- Aquest treball te com a resultat la creació i manteniment dels artefactes associats al rol responsable del treball.
- Les activitats poden ser utilitzades com a unitats de planificació i control de progrés.
- Exemples: trobar els actors i casos d'ús, portar a terme proves, codificar una classe, etc.
- Noteu que una activitat pot estar composta per d'altres.
- Normalment les activitats tenen 3 parts:
  - reflexió (comprendre què s'ha de fer)
  - execució (modelar, implementar)
  - revisió (verificar, avaluar)

30

**Activity: Develop Business-Modeling Guidelines**

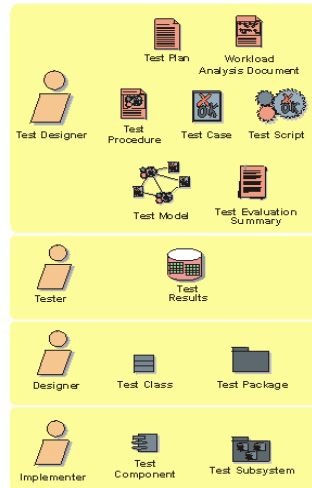
|   |  |
|---|--|
| <b>Purpose</b>  |  |
| <ul style="list-style-type: none"> <li>■ To develop business-modeling guidelines.</li> </ul>  |  |
| <b>Steps</b>  |  |
| <ul style="list-style-type: none"> <li>■ <a href="#">Tailor the business-modeling guidelines</a></li> <li>■ <a href="#">Capture decisions</a></li> </ul>  |  |
| <b>Input Artifacts:</b>   | <b>Resulting Artifacts:</b>  |
| <ul style="list-style-type: none"> <li>■ <a href="#">Business-Modeling Guidelines</a></li> <li>■ <a href="#">Development Case</a></li> <li>■ <a href="#">Tools</a></li> </ul>   | <ul style="list-style-type: none"> <li>■ <a href="#">Business-Modeling Guidelines</a></li> </ul> |
| <b>Frequency:</b> Before business modeling starts.  |  |
| <b>Role:</b> <a href="#">Business-Process Analyst</a>   |  |
| <b>Workflow Details:</b>  |  |
| <ul style="list-style-type: none"> <li>■ <a href="#">Core Workflow: Business Modeling</a> <ul style="list-style-type: none"> <li>■ <a href="#">Assess Business Status</a></li> </ul> </li> <li>■ <a href="#">Core Workflow: Environment</a> <ul style="list-style-type: none"> <li>■ <a href="#">Prepare Guidelines for an Iteration</a></li> </ul> </li> </ul> |  |

31

- Son elements d'informació produïts, modificats o usats pel desenvolupament del SW.
- Son l'entrada i la sortida de les activitats i la responsabilitat dels treballadors.
- Exemples: model de casos d'ús (models), un cas d'ús (elements d'un model), documents, codi font, executables, etc.
- Noteu que un artefacte pot estar compost per d'altres.
- Els artefactes es sotmeten a la gestió de la configuració i el control de canvis.
- En RUP els artefactes s'han dividit en 5 conjunts:
  - De gestió.
  - De requeriments.
  - De disseny.
  - D'implementació.
  - De desplegament.

32

## Test Artifact Set



33

- Seqüències d'activitats per produir artefactes.
- Es veuen les col·laboracions entre treballadors.
- En RUP venen expressats com diagrames d'activitat de l'UML.
- Els fluxos de treball principals definits en el RUP especifiquen les macro-activitats del seu eix vertical:

## Activitats seqüencials

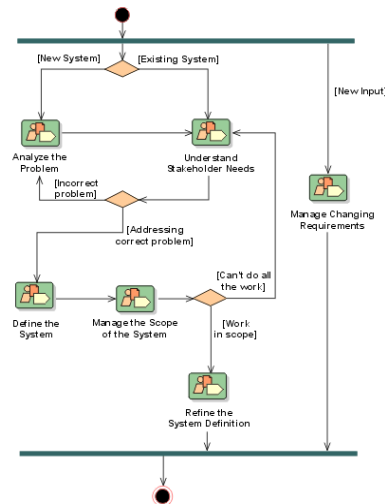
**Model d'empresa**  
**Requeriments**  
**Anàlisi i Disseny**  
**Implementació**  
**Prova**  
**Desplegament**

## Activitats de gestió (paral·leles)

**configuració**  
**projecte**  
**Entorn**

34

## Requirements: Overview



35

- Objectiu:
  - Entendre el conjunt de processos de negoci que esdevenen dins de l'empresa com a pas previ la recopilació de requeriments del sistema a desenvolupar.
  - Ens hem de centrar en: Com aconseguix l'empresa els seus objectius. ⇒ Re-enginyeria .
- Una empresa es caracteritza per:
  - **Informacions** que produeix i manipula
  - **Tasques** que produeixen o manipulen la informació
  - **Treballadors** que participen en les tasques
  - **Fluxos de treball** que defineixen com els treballadors col.laboren per complir una tasca



Definir un *casos d'ús del negoci* para mostrar el context i els límits de l'organització a estudi.

36

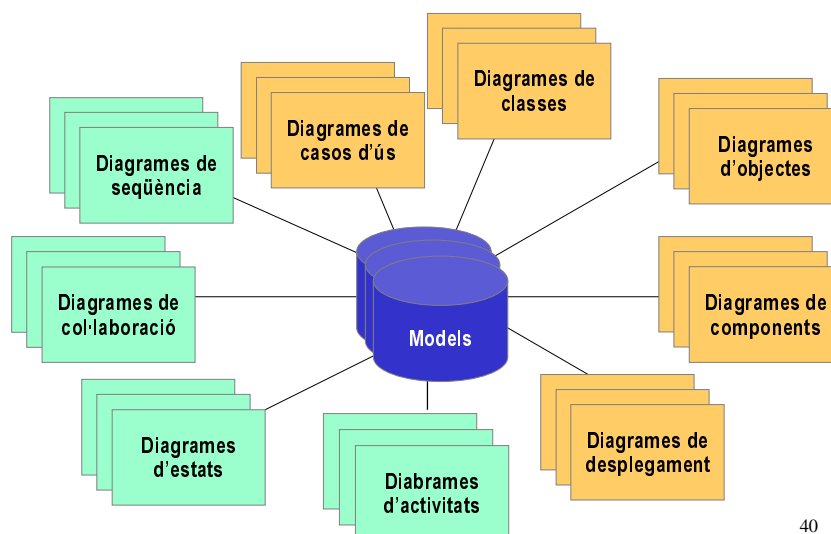
## Model de requeriments

- **Objectiu:**
  - S'han d'establir els requeriments funcionals i no funcionals del sistema.
- A partir del model de negoci es construeix el model de casos d'ús.
- **Per obtenir:**
  - Els casos d'ús s'extreuen de les accions.
  - Els actors s'extreuen dels rols que executen les accions.
  - Les classes s'extreuen de les informacions (entitats).

## Model d'anàlisi

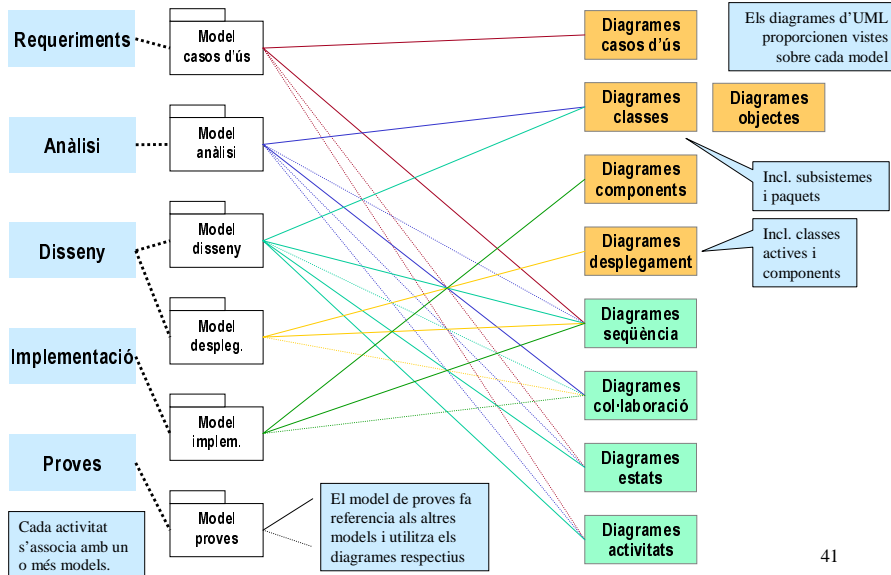
- **Objectiu:**
  - A partir dels casos d'ús obtenir el disseny preliminar del sistema que haurà de ser refinat en el model del disseny.
  - Nivell d'abstracció més alt que en el disseny.
  - Es defineix una arquitectura del sistema.
  - Ens permet arribar al sistema final de forma incremental, mitjançant evolució dels prototips.
- **Per cada cas d'ús es defineix la seva realització. Una realització consta de dues parts:**
  - Part estàtica: diagrama de classes d'anàlisi (comunicació, entitat i control).
  - Part dinàmica: diagrames d'interacció (seqüència i col.laboració).

- Objectiu:
  - Refinar el disseny del sistema del model d'anàlisi introduint els requeriments no funcionals i restriccions de l'entorn d'implementació.
- De manera iterativa es refina el diagrama de classes d'anàlisi fins obtenir un disseny del sistema adient per a passar a l'implementació:
  - noves classes, atributs, operacions
  - s'eliminen algunes classes
  - poden combinar-se algunes classes (modularitat efectiva)



6.4

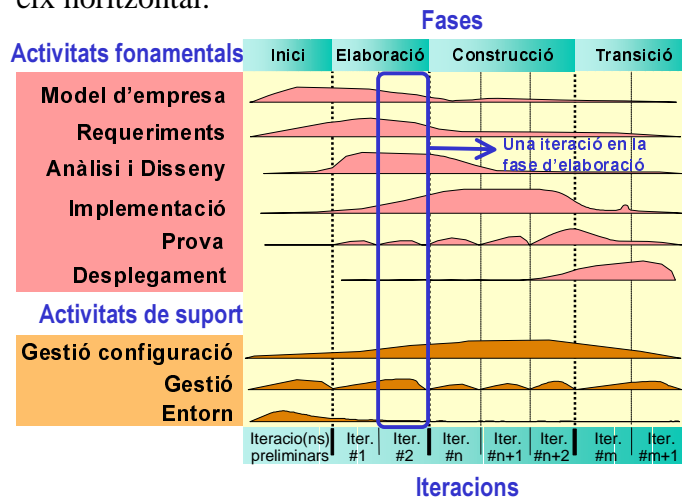
## Activitats, models i diagrames



6.5

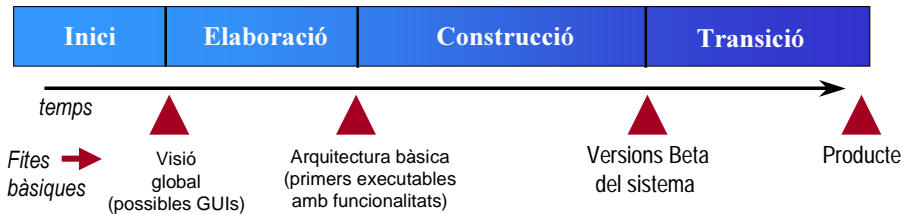
## Estructura dinàmica del procés

- Les fases definides en RUP especifiquen activitats del seu eix horitzontal:



6.5

## Fases del cicle de vida

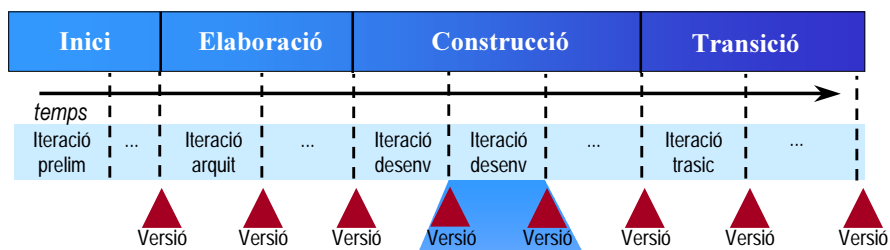


- **Inici.** Desenvolupar l'àmbit del projecte i definir el model de negoci (entorn). S'especifiquen actors i casos d'ús.
- **Elaboració.** Planificar el projecte, especificar el domini del problema, establir les bases de l'arquitectura. En acabar es té un esquelet del sistema.
- **Construcció.** Refinar l'arquitectura, integració contínua, construir el producte en iteracions successives.
- **Transició.** Transferència del producte als usuaris. Es comença amb una versió beta que es va perfeccionant en el decurs de la fase.

43

6.5

## Fases i iteracions



Una **iteració** és una seqüència d'activitats amb un pla establert i un criteri d'avaluació, donant com a resultat una versió executable.



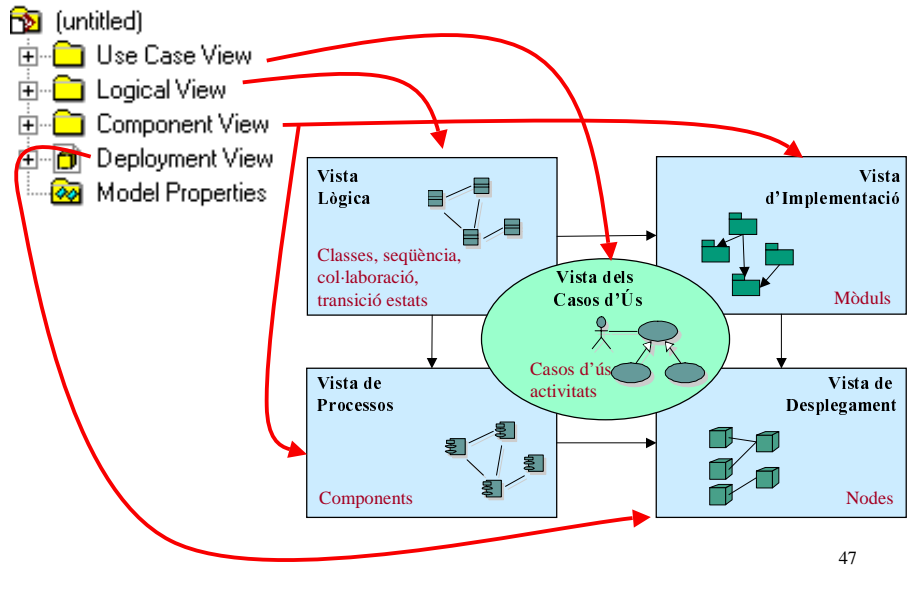
44

- Determinar el context i els requeriments més importants per derivar criteris d'acceptació del producte final.
- Determinar: riscos, recursos, cost.
- Artefactes: visió de negoci, model de casos d'ús, glossari, llista de riscos, pla de projecte.

- Aprofundiment en els requeriments de més risc (casos d'ús) i en general en la compressió del domini del problema i la seva solució.
- Configuració de l'entorn de desenvolupament.
- Primera versió de l'arquitectura.
- Artefactes: model de casos d'ús més elaborat, requeriments sense cas d'ús i requeriments no funcionals, llista de riscos revisada, prototipus executable.
- El RUP s'articula al voltant de 5 vistes del SW, aquestes vistes s'anomenen 4+1 vistes d'arquitectura i ens donen la possibilitat de centrar-nos en diferents aspectes del SW.

6.5

## Correspondència entre 4+1 vista i Rose



6.5

## Construcció

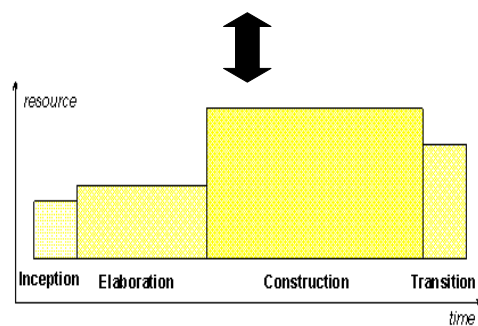
- Desenvolupament, integració i verificació de components.
- Lliurament d'una versió alfa o beta per a la seva validació.
- Artefactes: versió alfa o beta instal·lades, manuals d'usuari, etc.

48

- Validació alfa o beta.
- Funcionament paral·lel amb sistemes que s'han de reemplaçar.
- Entrenament dels usuaris.
- Correcció d'errors.
- Artefactes: producte final, manuals.

### Repartiment de recursos en un projecte típic

|          | <u>Inception</u> | <u>Elaboration</u> | <u>Construction</u> | <u>Transition</u> |
|----------|------------------|--------------------|---------------------|-------------------|
| Effort   | ~5%              | 20%                | 65%                 | 10%               |
| Schedule | 10%              | 30%                | 50%                 | 10%               |



6.5

## Nombre mitjà d'iteracions

|             |         |
|-------------|---------|
| Inici       | 0, 1    |
| Elaboració  | 1, 2, 3 |
| Construcció | 1, 2, 3 |
| Transició   | 1, 2    |



| Fase<br># itera. | Inici | Elaboració | Construcció | Transició | Total |
|------------------|-------|------------|-------------|-----------|-------|
| Poques           | 0     | 1          | 1           | 1         | 3     |
| Típic            | 1     | 2          | 2           | 1         | 6     |
| Alt              | 1     | 3          | 3           | 2         | 9     |

Regla heurística pel nombre d'iteracions:  $6 \pm 3$

51