Gestió i Desenvolupament del Software GDS

Activitats de la Gestió de la Configuració

Curs 2022-23

Marc Talló Sendra

ÍNDEX

- **Identificació IC i CM aggregates**
- ***Promotion management***
- ***Release management***
- ***Branch management***
- ***Variant management***
- ***Change management***

ÍNDEX

- **Identificació IC i CM aggregates**
- *Promotion management*
- *Release management*
- *Branch management*
- *Variant management*
- *Change management*

IDENTIFICACIÓ IC i CM AGGREGATES

- **Equivalent a BP1**: **Identificació** i **emmagatzematge** dels artefactes seleccionats com a **items de configuració** en un dipòsit (***repository***) segur.
- **Consideracions:**
  - **Identificar IC** (items de configuració):
    - Tots aquells **fitxers que necessitem** per poder compilar el programa.
    - **No afegir** els **fitxers** entremitjos generats pel **compilador**.
    - **No afegir** els **executables o llibreries** que generem.

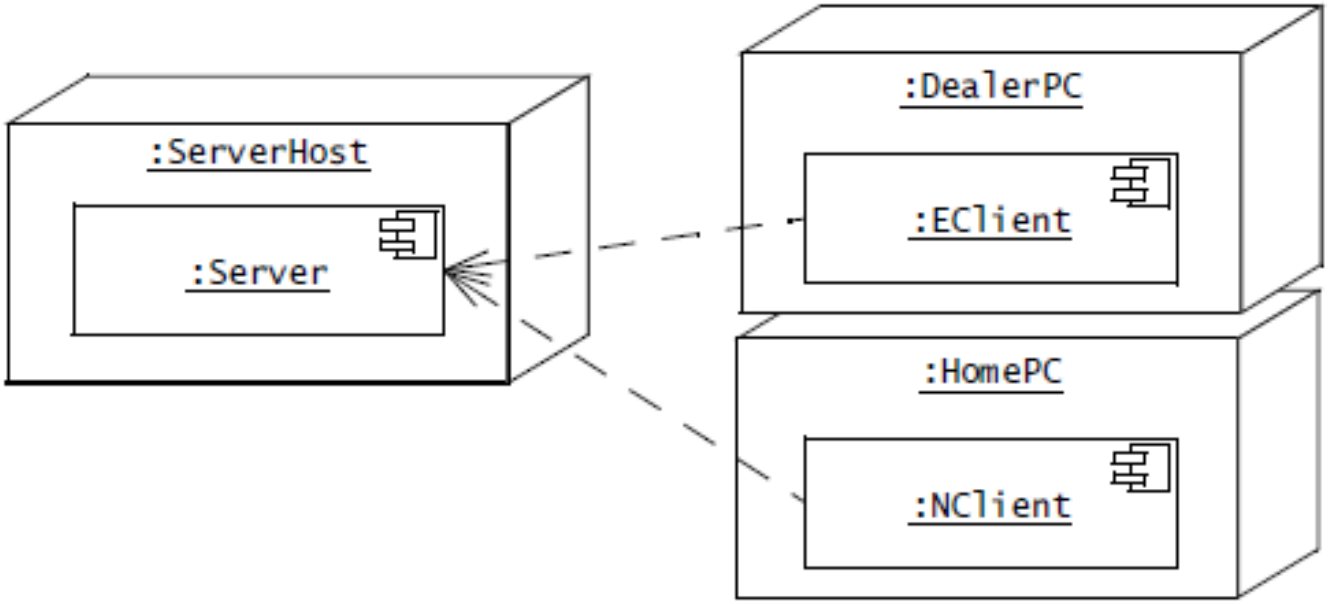
      Potser afegir llibreries que necessitem **per compilar i no venen** per defecte **amb el compilador**.

IDENTIFICACIÓ IC i CM AGGREGATES

- **Equivalent a BP1**: **Identificació** i **emmagatzematge** dels artefactes seleccionats com a **items de configuració** en un dipòsit (***repository***) segur.
- **Consideracions:**
  - **Identificar** els **CM aggregates** (Configuration Management aggregate):
    - Considerar **un únic agregat format per tots els ICs**.
      - És molt **fàcil de gestionar**.
      - Es generen **moltes versions** de tot el **software per canvis** en una part.
      - **No tenim versió pels mòduls** que formen el software.
    - **Agrupar** els **ICs** dels **mòduls** del **software** en **agregats**:
      - Hi ha una **gestió manual per saber** quines **versions de cada mòdul** son les **que formen el software final**.
      - Tenir **versió pels mòduls** del software i al tractar-los independentment en la gestió de configuració **facilitem** la seva **reutilització** (llibreries).
      - Necessitem un **repositori per cada agregat** (aconsellable).

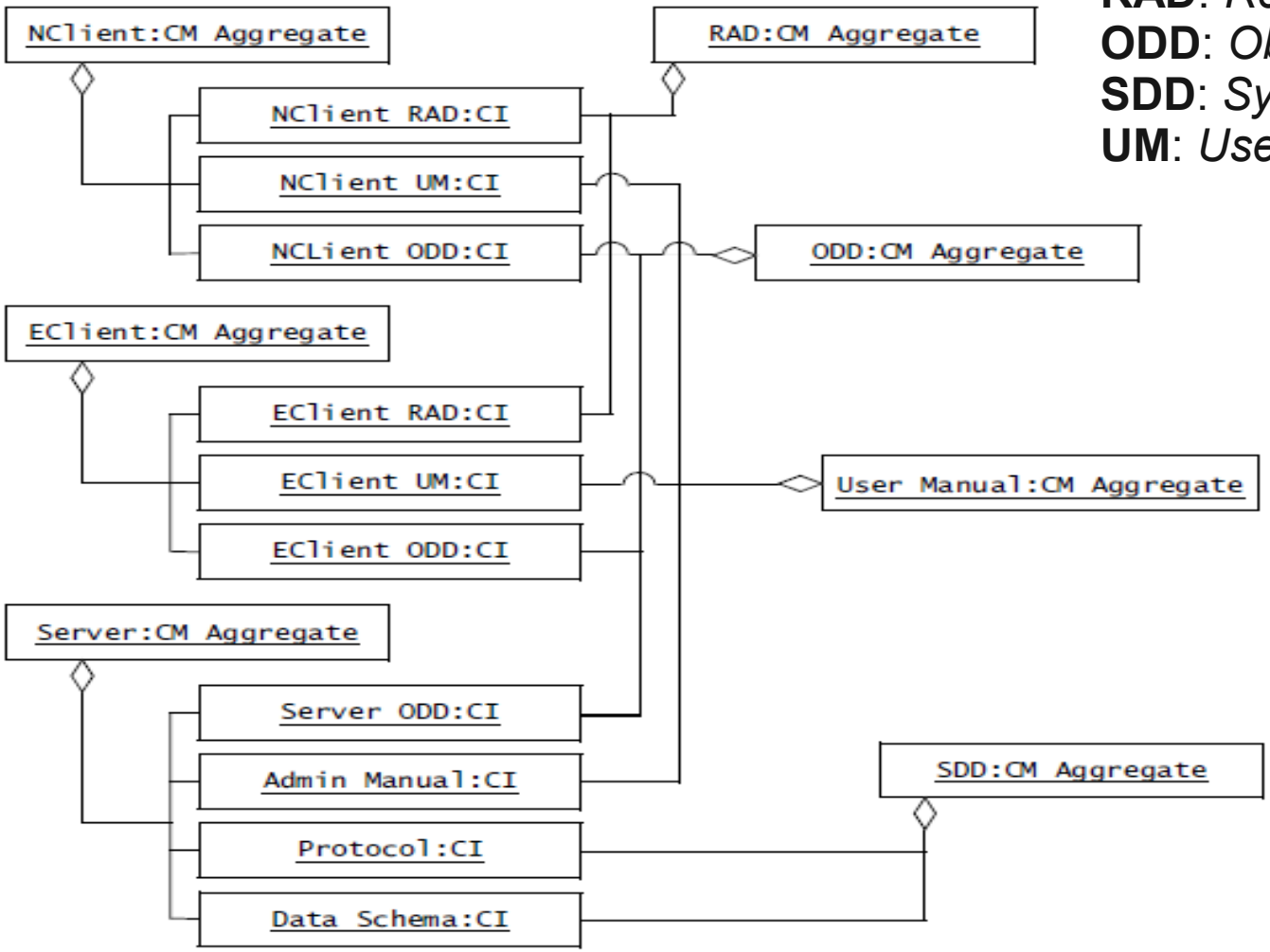
IDENTIFICACIÓ IC i CM AGGREGATES

- **Exemple:**
  - Un **servidor** *(:Server)* que **mostra** un **catàleg** de les **peces** utilitzades en la **fabricació** d'un **cotxe**.
  - **Dos tipus d'usuari:**
    - **Expert** *(:EClient): venedors, reparadors …*
    - **Novell** *(:NCLient): usuaris cotxe*



IDENTIFICACIÓ IC i CM AGGREGATES

- **Exemple de CM *aggregates*:**



RAD: *Requirements Analysis Document*
ODD: *Object Design Document*
SDD: *System Design Document*
UM: *User Manual*

ÍNDEX

- Identificació IC i CM aggregates
- ***Promotion management***
- *Release management*
- *Branch management*
- *Variant management*
- *Change management*

Promotion management

- **Exemple proposta** de **canvi**: L'**usuari novell** ha de poder **buscar** la **peça a partir d'un mapa gràfic del cotxe** (navegador gràfic).
  - **Alguns CM *aggregates* es veuen afectats, altres no.**
  - **P.Ex. Modifiquem el RAD:**
    - **Altres desenvolupadors** poden seguir **treballant en les seves branques.**
    - **Al acabar**, fem una **nova promoció** del **RAD:CM *aggregate*.**
    - La **resta de desenvolupadors poden**, o no, **fer un rebase.**
- Els **usuaris interessats** en el **IC** promocionat, fan un ***rebase*.**

ÍNDEX

- Identificació IC i CM aggregates
- *Promotion management*
- ***Release management***
- *Branch management*
- *Variant management*
- *Change management*

Release management

- **Nou release:**
  - **Decisió** basada en **marketing i *quality control***.
  - **Sembla una promotion**, però és molt **més complex i costós**.
    - Si hi ha un **error** greu en una ***promotion***, sempre podem **tornar a** una **promoció anterior**. En el cas d'un ***release***, **no** podem.
  - ***Quality control team*:**
    - Comprova qualitat dels components. **Interferència mínima** amb el treball dels **desenvolupadors**.
    - Realitza **testing** (no els usuaris!!!)
    - Pot **demanar a desenvolupadors** que **corregeixin errors**.
    - Un cop **s'assegura que tot funciona**, crea el ***release***.
  - Si el **nou release** és **per a software developers**:
    - Es pot **demanar** que l'**usuari** faci: test, debug, correcció d'errors ...
      - S'anomena ***bazaar model*** (Ex: open source, ...).

Release management

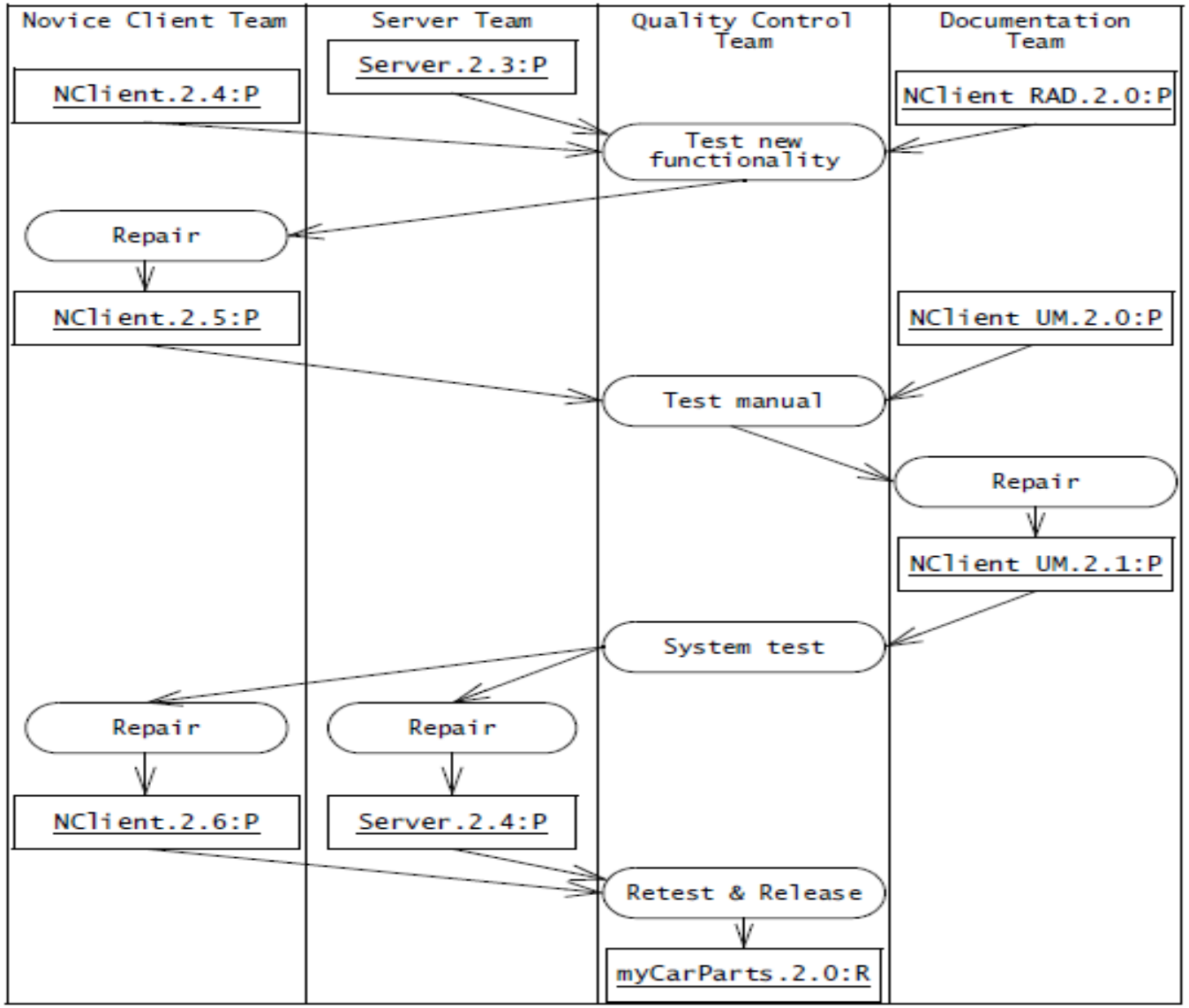


Diagrama d'activitats d'un exemple de *release management.*

P: Promotion
R: Release

ÍNDEX

- Identificació IC i CM aggregates
- *Promotion management*
- *Release management*
- ***Branch management***
- *Variant management*
- *Change management*

Branch management

- Permet **desenvolupament concurrent** (fins ara ens havíem centrat en aspectes seqüencials).
- **Exemple:** Suposem 3 equips treballant en:
  - Navegació gràfica:
    - Treballa amb versió més nova de `Nclient` i `Server`.
  - Extensió `Eclient` per guardar històric de cerques:
    - Treballa amb versió anterior (i més estable) de Server.
  - Millora temps resposta servidor.

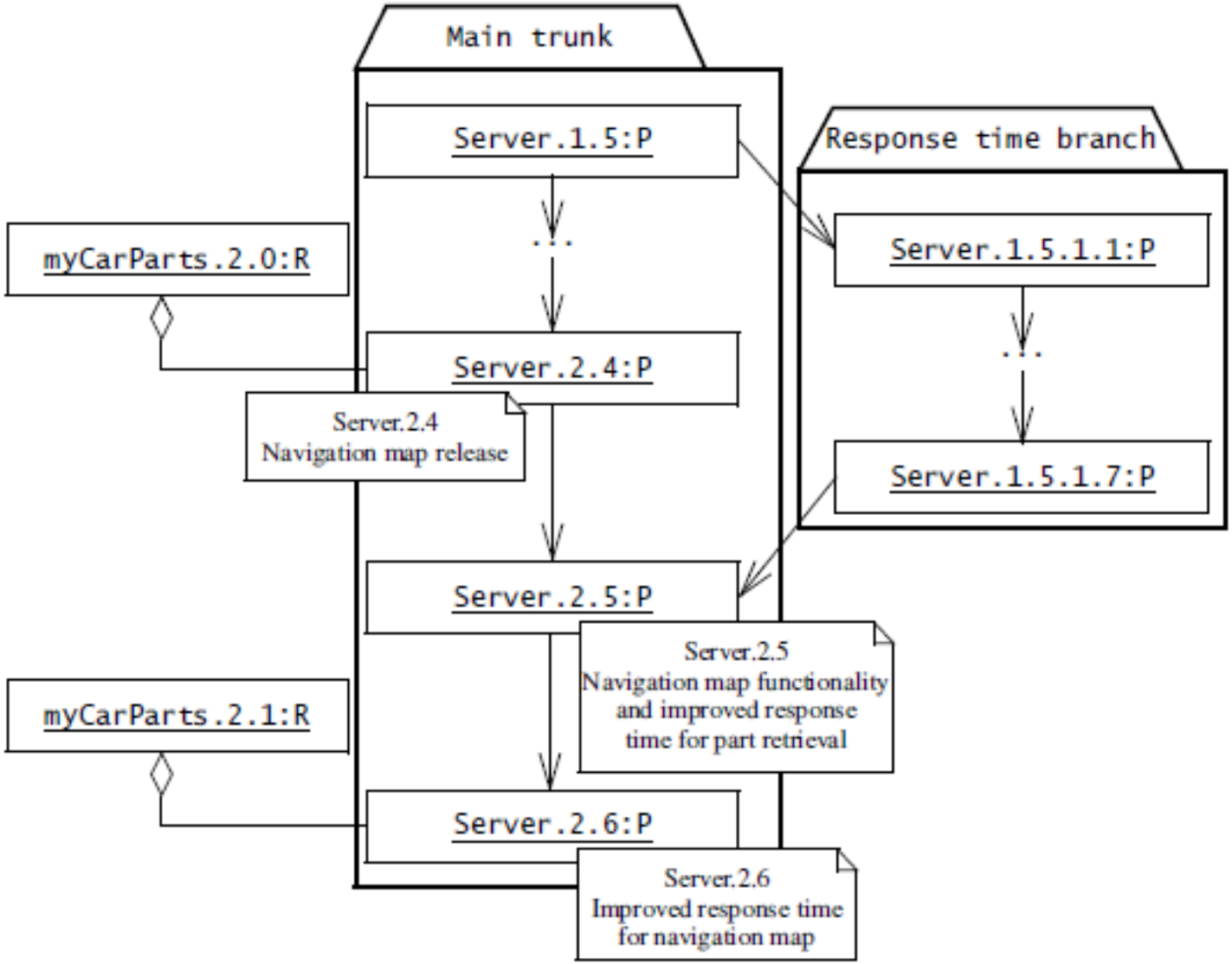
Canvis en components *non-overlapping* → OK

Canvis en el mateix component → *branches* *(i merging)*

Branch management

- ***Branches*** ens permeten **més flexibilitat en l'entrega** *(promotion o release)*:
  - Si una branca no s'ha completat, es pot entregar més tard.
- **Què (desenvolupar) i on?**
  - **Millores funcionals → Branca *main.***
    - *Canvis que només afecten els interfaces.*
  - ***Millores de desenvolupament → Branch.***
    - *Desenvolupament de Ics.*
  - **Al final** cal fer un ***merge.***
    - *Eines de GC (ho heu vist a pràctiques).*
    - *Cal fer testing !!!*

Branch management



Exemple de *branch management*

Branch management

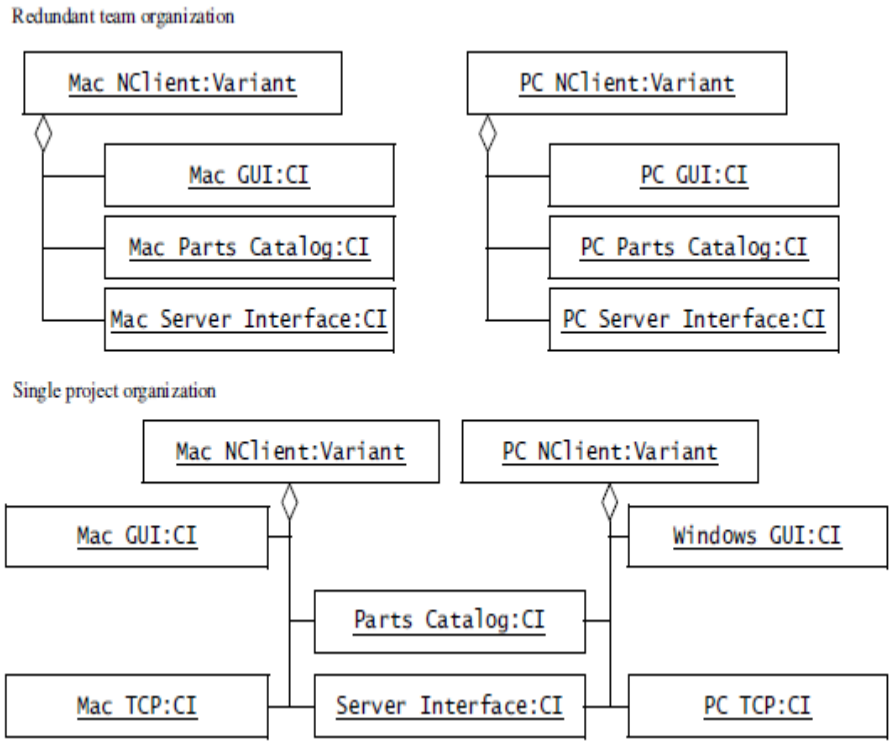
- **Receptes per *branch management*:**
  - **Minimitzar** els **overlaps**.
  - **Si no** es poden **minimitzar**, llavors:
    - **Identificar** els **overlaps** més probables:
      - Abans de dissenyar i implementar, identificar els possibles overlaps (P.Ex. No modificar els interfaces de les overlapped classes).
    - ***Rebase* frequentment**:
      - Afecta a les *branch* i no al *main.*
      - Ajuda a trobar els overlaps.
    - **Comunicar conflictes** probables:
      - Comunicar **entre equips** els **probables overlaps**.
      - **Millora** el **disseny** (té en compte els *constrains* de diferents equips).
    - **Minimitzar canvis** en el ***main.***
      - **Al main, només corregir bugs!!!** Fer tots els altres **canvis a les *branch*.**
    - **Minimitzar** el nombre de **branques**:
      - **No abusar** de les branques!!!
      - **Moltes branques** → **Més** possibilitat de **overlaps**.
      - Crear **branch** només quan es **necessita desenvolupament concurrent i** quan els **conflictes** es poden **solucionar**.
  - **Requereix de *management approval*** i s'ha de planejar curosament.

ÍNDEX

- Identificació IC i CM aggregates
- *Promotion management*
- *Release management*
- *Branch management*
- ***Variant management***
- *Change management*

Variant management

- **Variants**: **Versions que co-existeixen**.
  - **Exemple:** Diferents SO, plataformes, graus de funcionalitat (versions per novells, per experts, ...).
  - Dues **aproximacions fonamentals**:
    - ***Redundant teams***:
      - Un **equip per** cada **variant**.
      - Tots tenen, aproximadament, els mateixos requeriments.
      - Només un pocs IC són compartits pels equips.
    - ***Single project***:
      - Es **maximitza** la **quantitat** de **codi compartit**.



Variant management

- Pregunta: Quin **model** creieu que és el **més utilitzat?**
  - Sorprenentment, a la industria és el ***Redundant teams***.
  - **Redueix** la **complexitat** en la **organització**.
  - **Problemes** introduïts pel **code sharing** *(single project):*
    - ***Single supplier / multiple consumers:***
      - **IC comuns** són utilitzats per equips amb **finalitats** (i potser requeriments) **diferents**. Aquest IC han de **complir tots aquests requeriments**.
    - Peticions de **canvi** de **llarga durada**:
      - Si es demana un **canvi a** un **IC** comú el temps **pot ser molt llarg** (ha de tenir en compte els requeriments de tots els equips).
    - **Inconsistències entre variants**:
      - Un **IC** comú pot estar **dissenyat amb** un ***thread-control* en ment**. Mentre que una **variant** pot necessitar una **GUI amb arquitectura amb control de fluxe** *event-driven*.

Variant management

- Maneres de **reduir** aquests **problemes**:
  - ***Single supplier / multiple consumers:***
    - Si un **canvi** pertany a una **variant**, **no** fer-la en el **codi comú**.
    - **Si** un **canvi beneficia** a **TOTES les variants**, llavors es fa en el **codi comú**.
  - **Peticions** de **canvi** de **llarga durada**:
    - **S'involucra** a l'**equip** que demana el **canvi**.
    - El canvi es realitza en una **nova *promotion* del codi comú** (es pot fer en una *branch*).
    - La **resta** d'**equips treballen** en la ***promotion* anterior** del **codi comú**.
    - S'envia a l'**equip que va demanar el canvi i ho valora**.
    - **Un cop és valida**, la **resta** d'**equips** fan un ***rebase***.

ÍNDEX

- Identificació IC i CM aggregates
- *Promotion management*
- *Release management*
- *Branch management*
- *Variant management*
- ***Change management***

Change management

- **Equivalent** a **BP2** i **BP3:**
  - Registre i **seguiment** de **peticions de canvi, control i auditoria**.

Gestió i Desenvolupament del Software GDS

FI Activitats de la Gestió de la Configuració

Curs 2022-23

Marc Talló Sendra