Curs 20-21

# UD 7. Anàlisi i disseny orientat a objectes

7.1. Introducció a UML

Entorns de desenvolupament

IES Jaume II El Just
Tavernes de la Valldigna

# Continguts

# Introducció

UML (Llenguatge Unificat de Modelat) és un llenguatge universal (conjunt d'eines, majoritàriament gràfiques i no lligat a cap llenguatge de programació concret) que ens ajuda en el disseny de sistemes que plasmen idees d'una forma convencional i fàcil d'entendre per comunicar a altres persones.

La versió actual del llenguatge és la 2.5.1, publicada la seua última revisió en desembre de 2017 (https://www.omg.org/spec/UML).

Com va vam veure a les primeres unitats, un dels principals problemes a l'hora d'abordar el desenvolupament de programari és captar amb precisió allò que desitja l'usuari. Per tal d'abordar aquest problema, i d'aportar un flux de treball coherent a l'hora de desenvolupar els sistemes, apareixen diferents metodologíes de desenvolupament: Cascada, Espiral, amb Prototipat, metodologíes àgils, etc. En aquestes metodologíes, establíem diverses fases que abordaven la conceptualització del sistema (fase d'anàlisi), el seu disseny, la implementació i les proves i manteniment.

El llenguatge **UML** no és una metodología en sí, sinò un **conjunt d'eines de suport per a aquestes fases de desenvolupament**. Per altra banda, es tracta d'un llenguatge que ha estat àmpliament utilitzat amb models de desenvolupament clàssics, com el model en cascada, on es conceptualitza tot el sistema d'una vegada. Amb les metodologíes àgils aplicades al desenvolupament de programari, com Scrum, XP, etc. UML ha perdut popularitat, ja que les aplicacions es desenvolupen en base a anar incorporant poc a poc nous requeriments i desenvolupant per complet xicotetes funcionalitats. No obstant això, UML segueix sent una excel·lent eina si l'apliquem correctament per a la docuentació dels nostres projectes i per a la captació de nous requeriments.

Durant el que queda de curs, veurem UML com a suport per al desenvolupament de programari basant-nos en aquests models àgils, que van incorporant poc a poc noves funcionalitats.

# 2. Organització d'UML

UML es compon de diferents diagrames per tal de presentar diferents perspectives d'un sistema (models).

UML 2.5 estableix dos tipus de diagrames principals: *diagrames estructurals* i *diagrames de comportament*. Jeràrquicament, podriem representar tots aquests diagrames de la següent forma (font: Wikipedia):

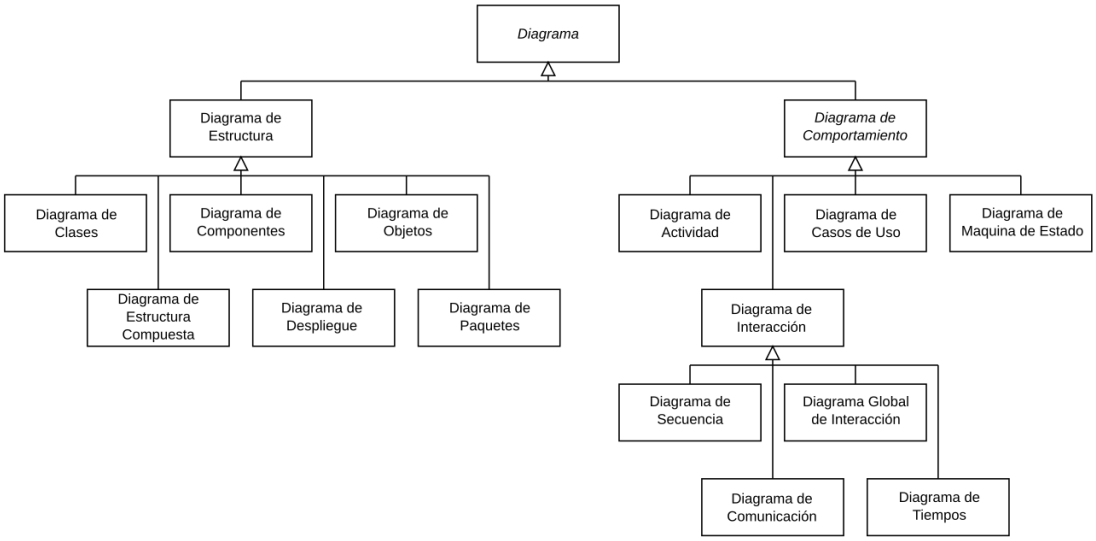


**Figura 1:** Conjunt de diagrames d'UML

Veiem una descripció breu d'aquests diagrames:

- ***Diagrames Estructurals***: Representen l'**estructura estàtica** dels objectes en un sistema. Distingim els següents diagrames:
  - **Diagrama de classes**: Es tracta del tipus de diagrama més utilitzat, i mostra les clases d'un sistema, junt amb els seus atributs i operacions, així com les relacions entre les diferents classes.
  - **Diagrama d'objectes o d'instància**:Semblants als diagrames de classes, però fent ús d'exemples concrets del món real. La seua utilitat és saber com es vorà un sistema en un moment donat.
  - **Diagrama de paquets**: Mostra les dependències entre els diferents paquets (*packages*) del nostre sistema.
  - **Diagrama de components**: Mostra la relació estructural dels diferents components d'un sistema de programari. S'utilitza principalment quan treballem amb sistemes complexos, formats per diferents subsistemes que es comuniquen entre sí mitjançant APIs. Un exemple podria ser una aplicació client-servidor, on tindriem el component client i el component servidor.
  - **Diagrama de desplegament**: Mostra el maquinari i programari en sistemes que requereixen de diversos equips, amb diferents configuracions.
  - **Diagrama de perfils**: Permeten extendre l'UML per tal d'adaptar-se a una plataforma concreta de programació (ja que no tots els llenguatges orientats a objectes són

exactament iguals). Així, tindrem diagrames de perfils per a Java, C#, etc.
  - **Diagrama d'estructura composta**: Mostren l'estrucutra interna d'una classe. És semblant al diagrama de classes, però permet entrar més en detall en l'estructura interna d'alguns components.

- ***Diagrames de comportament***: Mostren el comportament dinàmic dels objectes en el sistema:
  - **Diagrama de casos d'ús**: Es tracta d'un dels diagrames més coneguts d'UML, i ofereixen una visió general dels actors (ja siguen usuaris o altres sistemes) involucrats en el sistema, i mostren les funcions que necessiten realitzar aquests en el sistema en desenvolupament i com interactúen entre elles. És un diagrama de gran utilitat quan comencem a desenvolupar un sistema, ja que ens permet identificar fàcilment els actors i els principals processos del sistema.
  - **Diagrama d'activitats**: Mostren gràficament els fluxos de treball, tant a nivell empressarial com operatiu de qualsevol dels components del sistema.
  - **Diagrama de màquina d'estats**: Descriuen el comportament dels diferents objectes del sistema, en funció del seu estat.
  - **Diagrames d'interacció**: Es tracta d'un subconjunt de diagrames de comportament que comprén:
    * *Diagrames de seqüència*: Mostren com interactúen entre ells els diferents objectes del sistema en un escenari en particular.
    * *Diagrama de comunicació (anteriorment anomentat diagrama de col·laboració)*: Semblant als diagrames de seqüència, però centrant-se en els missatges que es passen els diferents objectes.
    * *Diagrama de temps*: Representen el comportament dels objectes en un marc de temps donat.
    * *Diagrama global d'interaccions*: Molt semblants ald diagrames d'activitat, però mostrant una seqüència de diagrames d'interacció. Podem dir que es tracta d'una col·lecció de diagrames d'interacció i l'ordre en què aquestes ocorren.

Conéixer tots aquests diagrames i saber desenvolupar-los és feina dels enginyers de programari. Al nostre cas, anem a familiaritzar-nos amb els principals diagrames, per tal de poder entendre'ls i implementar-los, així com realitzar modificacions per tal d'ampliar la funcionalitat d'un sistema. Conéixer-los ens servirà també per aprofundir en el nostre coneixement sobre la programació orientada a objecte i l'organització del programari.

## 3. Eines CASE: UmbrelloUML

Sote el terme d'**eines CASE** (*Computer Aided Software Engineering*, o *Enginyeria del programari Assistida per Ordinador*) s'engloba tot un conjunt d'aplicacions informàtiques de suport en les diferents etapes del cicle de vida.

Hi ha eines CASE més completes i costoses que altres, i ens ajudaran en major o menor mesura en es nostres tasques. Dos de les principals funcionalitats que trobarem a aquestes eines seran:

- Ens ajudaran a *dibuixar* els diferents diagrames, per tal de documentar els nostres projectes i comunicar-nos millor amb l'usuari, i
- Ens generaran automàticament part del codi de les nostres aplicacions, al menys en la seua estructura bàsica.

Algunes de les eines CASE que podem trobar, i les principals característiques són:

- **Dia**: Senzill, simple i versàtil. Permet la generació de codi (Java, Python, C/C++) a partir dels diagrames de classes. No suporta UML2. Multiplataforma. Programari lliure.
- **Argo UML**: Permet la generació de codi Java a partir dels diagrames de classes. No suporta UML2. Multiplataforma. Té un gran inconvenient: no té l'opció de desfer (undo).
- **Microsoft Visio**: Permet fer varis tipus de diagrames UML.
- **Visual Paradigm**: Molt complet i intuïtiu. S'integra amb Eclipse i Netbeans, però les característiques avançades com la generació de codi i l'enginyeria inversa (crear el diagrama a partir del codi) necessiten una llicència de pagament.
- **Papyrus**: Complement d'Eclipse, gratuït però molt complex.
- **Object Aid**: Plugin d'Eclipse. Genera diagrames de classe per mitjà d'enginyeria inversa i, amb la versió de pagament, també diagrames de seqüència.
- **EasyUML**: Plugin de NetBeans. Només permet la creació de diagrames de classe.

Una altra eina, que és la que utilitzarem és ***Umbrello UML Modeller***, disponible directament des dels repositoris d'Ubuntu, i que forma part del conjunt d'eines de KDE.

Des de la seua pàgina web https://umbrello.kde.org/ podeu descarregar les versions per a Windows, Mac i Snap per a Ubuntu, tot i que podem instal·lar-lo des dels repositoris d'ubuntu amb:

```
1 sudo apt install umbrello
```

Com comenta a la web, *Umbrello UML Modeller* és una eina de programari lliure per a la realització de diagrames UML, que permet crear diagrames de programari i altres sistemes per tal de dissenyar la documentació i l'estructura de les nostres aplicacions.

Una vegada instal·lat, quan obrim *Umbrello UML Modeller* trobarem una interfície semblant a la següent:

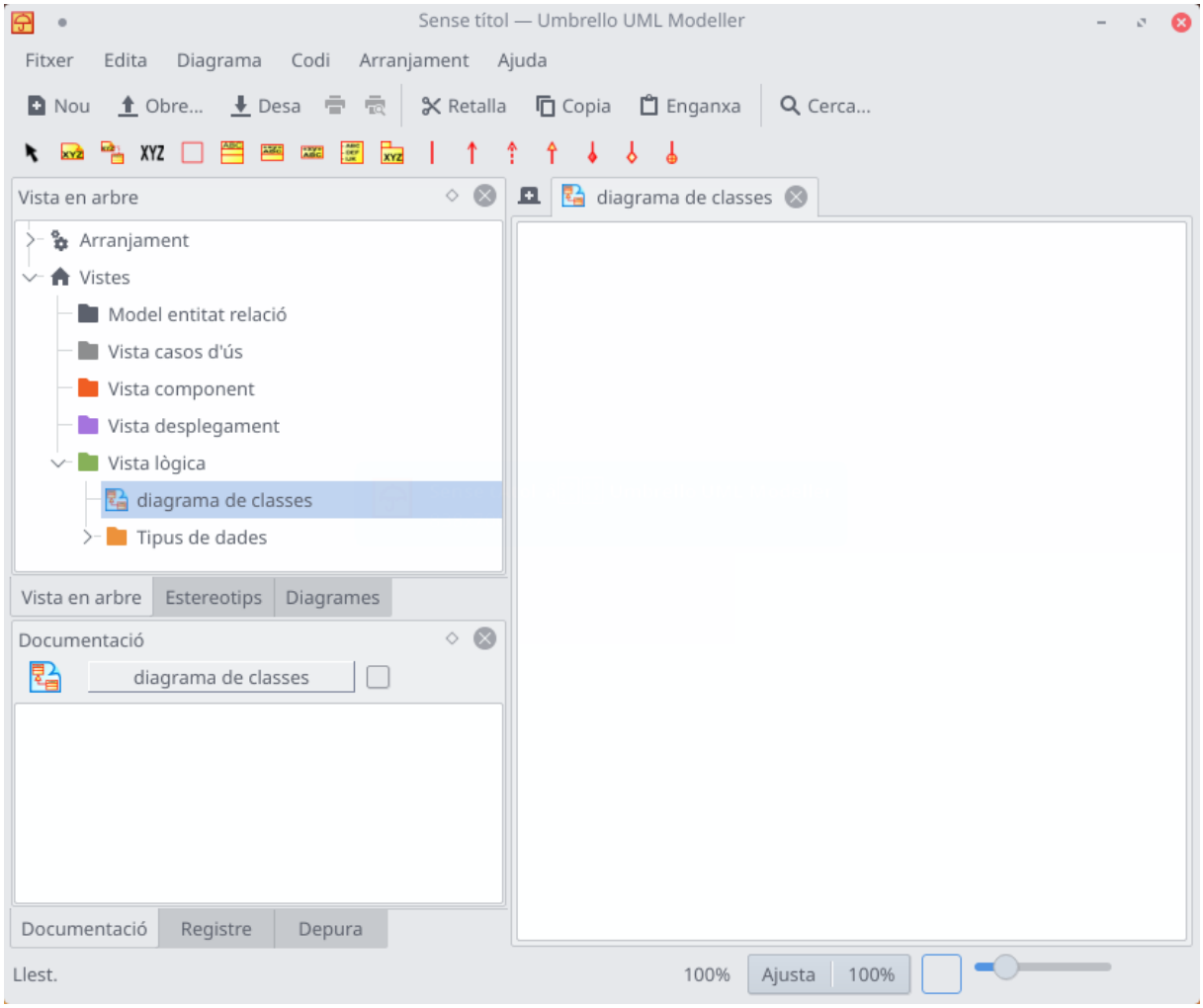


**Figura 2:** Interfície d'Umbrello

Com podem veure, a la part de l'esquerra se'ns mostren diferents carpetes per a les diferents *vistes* de la nostra aplicació. Cada *vista* contindrà un o diversos tipus de diagrames. Els diagrames que utilitzarem en aquesta unitat serà el diagrama de casos d'ús (*Vista casos d'ús*), i els diagrames de classes (*Vista lògica*).

Es tracta d'una aplicació bastant senzilla, que ens permet anar creant els diagrames incorporant els diferents elements i establint les relacions entre ells. A més, també ens permetrà generar el codi de base per a les nostres aplicacions.

Podeu consultar tota la informació sobre com generar diagrames UML des del seu complet manual: https://docs.kde.org/trunk5/es/kdesdk/umbrello/.

## 4. Referències

Podeu trobar més informació d'utilitat a:

- https://www.uml.org/
- https://diagramasuml.com/
- https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado