Descripció de les classes del projecte

Equip 43.3

Tercera entrega

Professor: Jose Miguel Urquiza

Membres del grup:

Lluc Santamaria Riba
lluc.santamaria

Efraín Tito Cortés
efrain.tito

Alejandro Lorenzo Navarro
alejandro.lorenzo

Eulàlia Peiret Santacana
eulalia.peiret

Índex

1. Introducció

En aquest projecte hem desenvolupat un sistema innovador per a supermercats que permet als usuaris gestionar un catàleg de productes i organitzar-los de manera òptima en prestatgeries. L'objectiu principal és maximitzar els beneficis derivats de les similituds entre productes mitjançant la implementació de tres algorismes diferents que ofereixen distribucions òptimes.

En aquest document es troba la descripció d'atributs i mètodes de cada classe de la capa de domini, de les classes de persistència i de les classes de presentació exceptuant les vistes.

1.1. Funcionalitats extres

A més a més de les funcionalitats bàsiques que demana l'enunciat de la pràctica, també hem implementat les següents extres:

1. El sistema permet definir restriccions entre productes, assegurant que la distribució compleix amb requisits específics. L'usuari pot decidir parells de productes que no vol que apareguin junts en la distribució del supermercat.
2. Hem implementat tres algorismes diferents per resoldre el problema de distribució. Un primer algorisme utilitza una estratègia de backtracking, el segon utilitza una estratègia voraç i el tercer és el 2-Aproximació especificat en la informació adicional proporcionada pel professorat de l'assignatura.
3. L'algorisme de 2-Aproximació implementa totes les possibles millores que en el document d'informació adicional posaven al nostre abast. Aquestes són:
    - Per fer l'algorisme de Kruskal eficient, s'utilitza l'estructura de dades Merge Find Set amb les millores de compressió de comins i unió per talles.
    - A l'hora de fer l'eliminació de repetits, s'utilitza el tercer punt especificat: Dur a terme varis passos de simplificació. En cada pas buscar el conjunt de nodes repetits consecutius que, al eliminar-los, donen el cicle amb menor cost. Repetir aquest procés fins que no quedin nodes repetits.

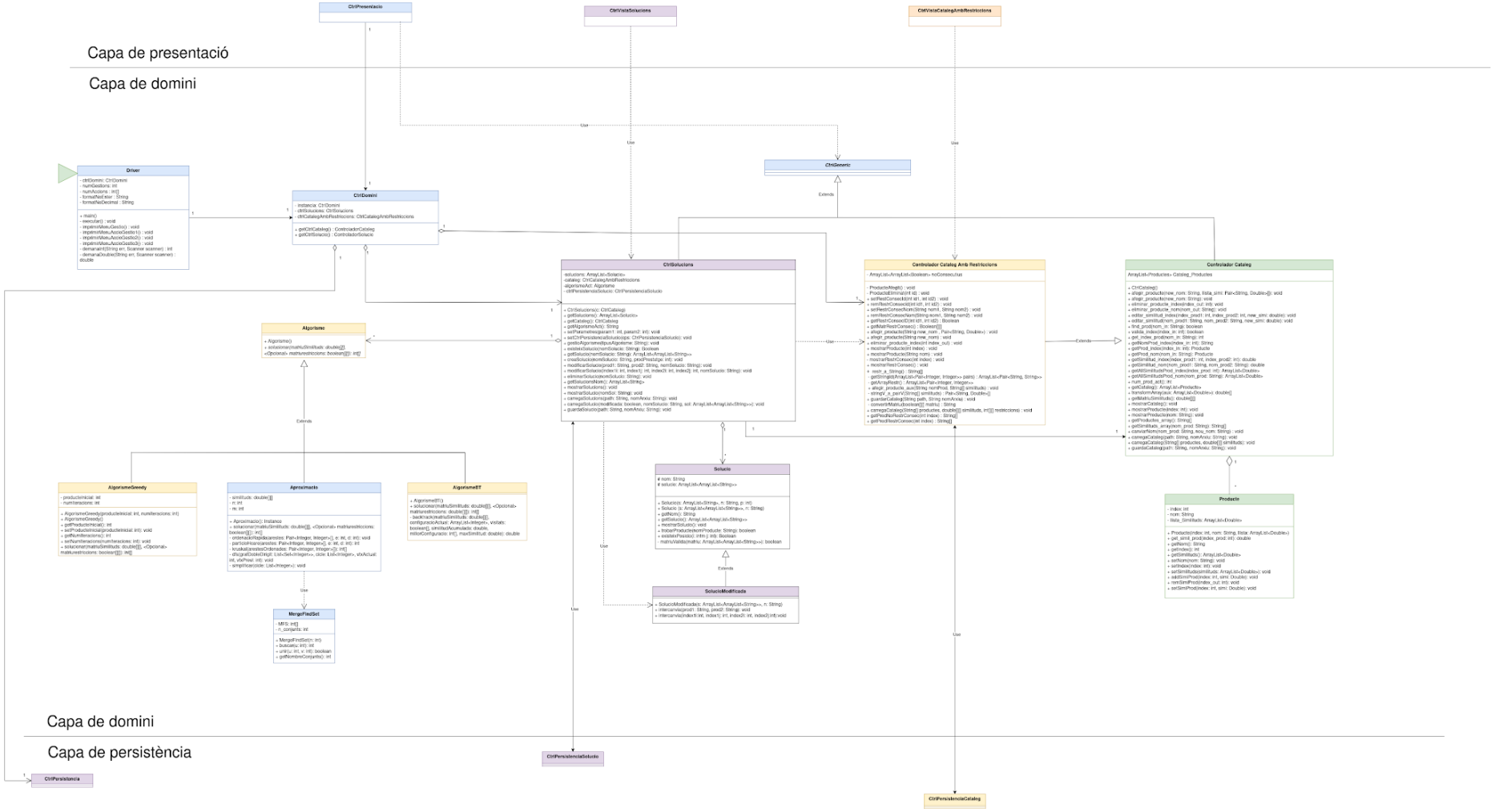
1.2 Millores respecte la primera entrega

En la primera entrega, vam entregar la capa de domini completa. Tot i que considerem que vam fer una bona feina, hem realitzat algunes millores:

1. En la primera entrega, vam deixar el terreny preparat per implementar les restriccions entre productes. Ara les hem implementat en tot el projecte. Es pot escollir qualsevol parell de productes per afegir una restricció i de forma gràfica. També es poden consultar les restriccions que hi ha presents entre productes i es poden eliminar les desitjades.
2. Ara les solucions es presenten mitjançant una matriu, on cada fila representa un prestatge. Aquesta millora redueix el cost computacional d'operacions com l'intercanvi de productes dins d'una solució.

3. Hem afegit la possibilitat de modificar un producte present en el catàleg. Aquesta millora cobreix la funcionalitat de modificació de productes.
4. Hem afegit la possibilitat de consultar únicament la similitud entre dos productes. En la primera entrega es mostraven les similituds amb tots els altres productes. Aquesta millora és excel·lent quan es gestionen molts productes.
5. S'ha millorat l'eficiència de l'algorisme greedy mitjançant un procés de selecció. Per una explicació detallada, consultar l'apartat "5.3 Algorisme Greedy" del document "Estructures de dades i algorismes".
6. Tal i com es demana en la pràctica, hem afegit una capa de persistència que permet guardar i carregar les dades de manera eficient des d'arxius del ordinador. També s'ha incorporat una capa de presentació que facilita la interacció amb el sistema, fent-lo més accessible i intuïtiu per a l'usuari.
7. Ara es permet afegir restriccions durant l'acció d'afegir productes, així facilitem a l'usuari donar més ràpidament tota la informació del nou producte, sense necessitat de moure's entre vistes.

2. Diagrama estàtic del model conceptual de dades del domini



2.1 Descripció del diagrama

Nom: Producte
Descripció Classes: Representa un producte que formarà part del catàleg del sistema.
Cardinalitat: Cada producte forma part d'un sol catàleg.
Descripció Atributs:

- private String nom: representa el nom únic que se li ha donat al producte.
- private int index: índex que coincideix amb l'índex que te producte dins del catàleg.
- private ArrayList<Double> llista_Similituds: representa les similituds d'aquest producte amb els altres productes al catàleg en cada moment.

Descripció Mètodes:

- public Producte(int index, String nom, ArrayList<Double> llista)
  - index: L'índex a Cataleg_Productes de la classe Cataleg on es troba el producte.
  - nom: El nom de producte a crear.
  - llista: Una llista amb totes les similituds del producte amb els altres.
  - Descripció: Es crea una nova instància de producte.
- addSimiProd(int index, Double simi)
  - index: L'índex del producte dins de llista_similituds.
  - simi: La nova similitud a afegir.
  - Descripció: Afegeix una nova similitud a la llista de similituds del producte.
- remSimiProd(int index_out)
  - index_out: L'índex de la similitud a eliminar.
  - Descripció: Elimina una similitud de la llista de similituds del producte.
- setSimiProd(int index, Double simi)
  - index: L'índex del producte dins de la llista de similituds.
  - simi: La nova similitud a establir.
  - Descripció: Assigna una nova similitud amb el producte a l'índex especificat de la llista de similituds.

Descripció Relacions:

- Relació d'associació amb la classe "Cataleg": Indica quins productes formen part de la prestatgeria.

Nom: CtrlGeneric
Descripció Classes: Controlador genèric del qual hereden CtrlCataleg i CtrlSolucions. Permet reduir l'acoblament al passar les instàncies de CtrlCataleg i CtrlSolucions a la capa de presentació. És una classe abstracte
Cardinalitat: No es pot instanciar, és abstracte
Descripció Atributs: -
Descripció Mètodes: -
Descripció Relacions: -

Nom: CtrlCataleg

Descripció Classes: Controlador de Catàleg (creat per l'usuari) per tenir una estructura on distribuir els productes, fer les diferents funcionalitats del programa.
Cardinalitat: Només hi ha un controlador de catàleg en tot el programa
Descripció Atributs:

- private ArrayList<Producte> Cataleg_Productes: Representa el conjunt de productes es troben dins del sistema.

Descripció Mètodes:

- Cataleg()
  - Descripció: Mètode de construcció d'un catàleg buit, sense cap producte.
- afegir_producte(String new_nom, Pair<String, Double>[] llista_simi)
  - new_nom: Nom del producte a afegir.
  - llista_simi: Similituds dels productes, format <Nom_Productes, Similitud>.
  - Descripció: S'afegeix un producte al final del catàleg si no es troba un producte amb el mateix nom. També es configuren les similituds associades amb el nou producte. Aquest mètode s'ha de fer servir per a la creació de productes dins del sistema.
- eliminar_producte_index(int index_out)
  - index_out: Índex del producte a eliminar.
  - Descripció: S'elimina un producte i totes les similituds associades a aquest. Advertència: Tots els índexs inicialitzats localment abans de l'execució de la funció seran erronis.
- eliminar_producte_nom(String nom_out)
  - nom_out: Nom del producte que es vol eliminar.
  - Descripció: S'elimina un producte i totes les similituds associades si aquest es troba dins del catàleg. Advertència: Tots els índexs inicialitzats localment abans de l'execució de la funció seran erronis.
- editar_similitud_index(int index_prod1, int index_prod2, double new_simi)
  - index_prod1: Índex del primer producte.
  - index_prod2: Índex del segon producte.
  - new_simi: Nova similitud entre els dos productes.
  - Descripció: Modifica la similitud entre els dos productes donats tal que sigui igual a la similitud proporcionada.
- editar_similitud(String nom_prod1, String nom_prod2, double new_simi)
  - nom_prod1: String que representa el nom del primer producte.
  - nom_prod2: String que representa el nom del segon producte.
  - new_simi: Nova similitud entre els dos productes.
  - Descripció: Modifica la similitud entre els dos productes donats tal que sigui igual a la similitud proporcionada. Si algun dels productes no es troba al catàleg, es dona error.
- find_prod(String nom_in)
  - nom_in: Nom del producte a buscar.
  - Descripció: Retorna True si el producte es troba dins del catàleg, i False si no es troba.
- valida_index(int index_in)
  - index_in: Índex a validar.
  - Descripció: Valida si l'índex rebut és vàlid i, per tant, es pot fer servir per accedir al catàleg. Aquesta validació es basa en la propietat de remove de ArrayList, que no deixa espais lliures en l'eliminació. Per tant, si l'índex està dins del rang de mida del catàleg, serà vàlid. Retorna TRUE si l'índex és vàlid, i FALSE si no ho és.

- num_prod_act()
    - Descripció: Retorna un enter que és igual al nombre de productes actuals al catàleg.
- transformArray(ArrayList<Double> aux)
    - aux: ArrayList a transformar.
    - Descripció: Transforma un ArrayList de Double en un array de tipus double[] amb els mateixos valors que el ArrayList original. Retorna un double[] amb els mateixos valors de aux.
- getMatriuSimilituds()
    - Descripció: Transforma el catàleg de productes en una matriu de similituds. Cada fila representa les similituds d'un producte amb els altres productes. Retorna una matriu de tipus double[][] de mida Num_productes^2.
- mostrarCataleg()
    - Descripció: Imprimeix al terminal tot el catàleg de productes. Per a cada producte, es mostra el seu nom i les similituds amb la resta de productes. Si el catàleg està buit, mostra un missatge indicant-ho. Crida internament el mètode mostrarProducte(int index) per mostrar la informació de cada producte.
- mostrarProducte(int index)
    - index: Índex del producte que es vol mostrar.
    - Descripció: Mostra els detalls del producte especificat per l'índex, incloent el seu nom i les seves similituds amb altres productes del catàleg. Si l'índex no és vàlid, mostra un missatge d'error. Les similituds es presenten en format: NomProducte -> valorSimilitud.
- mostrarProducte(String nom)
    - nom: Nom del producte que es vol mostrar.
    - Descripció: Busca el producte amb el nom especificat i mostra els seus detalls, incloent el nom i les similituds amb altres productes. Si el producte no existeix, llença l'excepció ProducteNoValid. Utilitza el mètode mostrarProducte(int index) per mostrar la informació si el producte és trobat.
- mostrarCataleg()
    - Descripció: Mostra tots els productes que hi ha a catàleg amb tota la seva informacio detallada
- getProductes_array()
    - Descripció: Transforma el ArrayList de productes en un vector String[], amb només els noms dels productes. Per poder enviar entre capes les dades.
- getSimilituds_array(String nom_prod)
    - nom_prod: Nom del producte del cual es vol les similituds
    - Descripció: Transforma el ArrayList de similituds del producte en un vector String[], amb només els valors de les similituds. Per poder enviar entre capes les dades.
- canviarNom(String nom_prod, String nou_nom)
    - nom_prod: Nom del producte que es vol modificar
    - nou_nom: Nou nom per el producte
    - Descripció: Es modifica el nom del producte identificat per nom_prod per el nou nom, “nou_nom”.
- carregarCataleg(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.

    - Descripció: Processa les dades d'un fitxer de productes. Valida el directori, carrega el contingut del fitxer, i divideix les dades en línies per extreure informació sobre cada producte
- guardarCataleg(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.
    - Descripció: Desa les dades a un fitxer de productes. Valida el directori, desa el contingut al fitxer, i divideix les dades en línies per extreure posteriorment informació sobre cada producte.
- carregaCataleg(productes: String[], similituds: double[][]): void
    - Productes: Array de productes a afegir del catàleg
    - similituds: Matriu de similituds dels productes a carregar
    - Descripció: El controlador de catàleg rep la informació que ha d'anar afegint al catàleg, i l'afegiex. El cataleg pasa a tenir unicaments els productes donats com a parametre

Descripció Relacions:
- Relació d'agregació amb la classe "Producte": Indica quins productes formen part del catàleg.
- Relació d'associació amb la classe "CtrlSolucions", el controlador de catàleg te la instància Controlador de solucions
- Relació de superclasse amb la classe "CtrlCatalegRestriccions".

Nom: Algorisme
Descripció Classes: Algorisme (creat per l'usuari) per donar una solució al problema, fent servir 3 possibles algorismes.
Cardinalitat: Hi ha un algorisme per el únic controlador de solucions
Descripció Atributs:
Descripció Relacions:
- Relació d'associació amb la classe "Control de solucions"

Descripció Mètodes:
- Algorisme()
    - Descripció: Constructor de la classe Algorisme.
- solucionar(double[][] matriuSimilituds, boolean[][] matriuRestriccions)
    - matriuSimilituds: Matriu de similituds entre productes, on matriuSimilituds[i][j] representa la similitud entre els productes i i j.
    - matriuRestriccions: Matriu de resttriccions entre productes, on matriRestriccions[i][j] representa si existeix una restricció entre els productes i i j.
    - Descripció: Depenent de la subclasse s'executa un algorisme concret.

Nom: AlgorismeBT
Descripció Classes: Algorisme de backtracking que dóna la solució òptima.
Cardinalitat: – (Es subclasse)
Descripció Atributs:
Descripció Mètodes:
- AlgorismeBT()
    - Descripció: Constructor de la classe AlgorismeBT.
- noPle(int[] millorConfiguracio)
    - millorConfiguracio: Representa una configuració de productes.

    - Descripció: Retorna True si encara hi ha espais no ocupats a la configuració (és a dir, és una solució incompleta), false en cas contrari.
- solucionar(double[][] matriuSimilituds, boolean[][] matriuRestriccions)
    - matriuSimilituds: Matriu de similituds entre productes, on matriuSimilituds[i][j] representa la similitud entre els productes i i j.
    - matriuRestriccions: Matriu de restriccions entre productes, on matriuRestriccions[i][j] representa si existeix una restricció entre els productes i i j.
    - Descripció: Mètode principal per trobar la configuració òptima de la prestatgeria. Retorna un vector d'índexs de productes al catàleg que representa la millor configuració trobada.
- backtrack(double[][] matriuSimilituds, ArrayList<Integer> configuracioActual, boolean[] visitats, double similitudAcumulada, int[] millorConfiguracio, double maxSimilitud)
    - matriuSimilituds: Matriu de similituds entre productes.
    - configuracioActual: Configuració actual d'índexs de productes.
    - visitats: Vector de seguiment per evitar repetir productes.
    - similitudAcumulada: Similitud total acumulada fins al punt actual de la configuració.
    - millorConfiguracio: Millor configuració trobada fins ara.
    - maxSimilitud: La màxima similitud trobada fins ara.
    - Descripció: Mètode recursiu per calcular totes les configuracions possibles de productes en la prestatgeria i trobar la millor configuració en termes de similitud total. Retorna el màxim de similitud trobat fins al moment per la configuració actual.

Descripció Relacions:
Relació de subclasse amb la classe “Algorisme”.

Nom: AlgorismeGreedy
Descripció Classes: Algorisme voraç que dóna una solució…
Cardinalitat: – (Es subclasse)
Descripció Atributs:
- private int producteInicial: Índex del producte inicial des del qual comença l'algorisme.
- private int numIteracions: Nombre d'iteracions que ha de realitzar l'algorisme.

Descripció Mètodes:
- AlgorismeGreedy(int producteInicial, int numIteracions)
    - producteInicial: L'índex del producte amb el qual comença l'algorisme. No pot ser negatiu ni superior a la quantitat de productes al catàleg, com a precondició.
    - numIteracions: El nombre d'iteracions que realitzarà l'algorisme. No pot ser negatiu ni 0, com a precondició.
    - Descripció: Constructor de la classe AlgorismeGreedy.
- solucionar(double[][] matriuSimilituds, boolean[][] matriuRestriccions)
    - matriuSimilituds: Matriu de similituds entre productes, on matriuSimilituds[i][j] és la similitud entre els productes i i j.
    - matriuRestriccions: Matriu de restriccions entre productes, on matriuRestriccions[i][j] representa si existeix una restricció entre els productes i i j.
    - Descripció: Mètode que resol el problema utilitzant l'algorisme voraç (greedy). Retorna un vector d'índexs de productes al catàleg que representa la millor configuració trobada.

Descripció Relacions:
- Relació de subclasse amb la classe “Algorisme”.

Nom: Aproximacio
Descripció Classes: Algorisme d'aproximació que dóna una solució…
Cardinalitat: – (Es subclasse)
Descripció Atributs:

- similituds: double[][]: Matriu de similituds : Matriu de similituds, donada per solucionar
- n: int : Mida de la matriu de similituds,
- m: int : Nombre d'arestes del graf que no tenen restriccions

Descripció Metodes:

- solucionar(double[][] matriuSimilituds, boolean[][] matriuRestriccions)
    - matriuSimilituds: Matriu de similituds entre productes, on matriuSimilituds[i][j] és la similitud entre els productes i i j.
    - matriuRestriccions: Matriu de restriccions entre productes, on matriuRestriccions[i][j] representa si existeix una restricció entre els productes i i j.
    - Descripció: Mètode que resol el problema utilitzant l'algorisme d'Aproximació. Retorna un vector d'índexs de productes al catàleg que representa la millor configuració trobada.
- ordenacioRapida(arestes: Pair<integer, Integer>[], e: int. d: int)
- particióHoare(arestes: Pari<Integerm Integer>[], e: int, d: int)
- kruskal(arestesOrdenades: Pair<Integer, Integer>[])
- dfs(grafDobleDirigit: List<Set<Integer>>, cicle: List<Integer>, vtxActual: int, vtxPrevi: int)
    - Descripció: Algoritme de cerca de profunditat.
- simplificar(cicle: List<Integer>)

Descripció Relacions:

- Relació de subclasse amb la classe "Algorisme":

Nom: CtrlSolucions
Descripció de la classe: Gestiona les solucions del sistema, permetent crear, modificar, eliminar, i consultar solucions, així com gestionar l'algorisme utilitzat per resoldre-les.
Cardinalitat: Només n'hi ha un en tot el programa.
Descripció atributs:

- ArrayList<Solucio> solucions: Llista de solucions que tracta.
- private Cataleg cataleg: És la instància del catàleg.
- private Algorisme algorismeAct: Algorisme predeterminat en aquell moment, per defecte és d'aproximació.
- private CtrlPersistenciaSolucio ctrlPersistenciaSolucio: Controlador per a la persistència de les solucions.

Descripció mètodes:

- CtrlSolucions(Cataleg c)
    - c: instància del catàleg, que pasara a ser el atribut catàleg.
    - Descripció: Constructor de la classe. Inicialitza les solucions, el catàleg i l'algorisme per defecte (aproximació).
- getAlgorismeAct()
    - Descripció: Retorna el tipus de l'algorisme actiu actualment (pot ser "backtracking", "greedy" o "aproximacio").
- setParametres(int param1, int param2)
    - param1, param2: Paràmetres utilitzats per configurar l'algorisme greedy.
    - Descripció: Configura l'algorisme greedy amb els valors especificats.
- gestioAlgorisme(String tipusAlgorisme)

    - tipusAlgorisme: Nom del tipus d'algorisme que l'usuari desitja utilitzar.
    - Descripció: Aquesta funció és cridada per l'usuari per establir l'algorisme desitjat. Si el tipus d'algorisme no és vàlid, es llença una excepció.
- existeixSolucio(String nomSolucio)
    - nomSolucio: Nom de la solució que es vol comprovar.
    - Descripció: Verifica si existeix una solució amb el nom especificat.
- creaSolucio(String nomSolucio, int prodPrestatge)
    - nomSolucio: Nom de la nova solució que es vol crear.
    - prodPrestatge: Número de productes per prestatge.
    - Descripció: Aquesta funció és cridada per l'usuari quan vol crear una nova solució. La nova solució es resol amb l'algorisme actual.
- modificarSolucio(String prod1, String prod2, String nomSolucio)
    - prod1: Nom del primer producte a intercanviar.
    - prod2: Nom del segon producte a intercanviar.
    - nomSolucio: Nom de la solució que es vol modificar.
    - Descripció: Intercanvia dos productes dins una solució específica si aquests existeixen.
- modificarSolucio(int index1i, int index1j, int index2i, int index2j, String nomSolucio)
    - index1i, index1j, index2i, index2j: Índexs dels productes a intercanviar dins la solució.
    - nomSolucio: Nom de la solució que es vol modificar.
    - Descripció: Intercanvia dos productes dins una solució específica proporcionant els seus índexs.
- eliminarSolucio(String nomSolucio)
    - nomSolucio: Nom de la solució que es vol eliminar.
    - Descripció: Aquesta funció és cridada per l'usuari quan vol eliminar la solució amb nom especificat. El mètode elimina la solució del sistema.
- getSolucionsNom():
    - Descripció: Retorna una llista amb els noms de totes les solucions.
- mostrarSolucions():
    - Descripció: Mostra totes les solucions que hi ha al sistema. Si no hi ha solucions, informa l'usuari.
- mostrarSolucio(nomSol: String)
    - nomSol: Nom de la solució que l'usuari vol consultar.
    - Descripció: Mostra la solució amb el nom especificat. Si no existeix, llança una excepció.
- carregaSolucions(String path, String nomArxiu)
    - path: Directori on es troba el fitxer.
    - nomArxiu: Nom del fitxer amb les solucions.
    - Descripció: Carrega les solucions des d'un fitxer. Si el fitxer no és vàlid, es llença una excepció.
- carregaSolucio(boolean modificada, String nomSolucio, ArrayList<ArrayList<String>> sol)
    - modificada: Indica si la solució està modificada.
    - nomSolucio: Nom de la solució.
    - sol: Distribució dels productes de la solució.
    - Descripció: Carrega una solució al sistema, indicant si és modificada o no.
- guardaSolucio(String path, String nomArxiu)

    - path: Directori on es vol guardar el fitxer.
    - nomArxiu: Nom del fitxer on es vol guardar la solució.
    - Descripció: Desa totes les solucions del sistema en un fitxer.

Descripció de les relacions:
- Relació amb la classe “CtrlDomini”.
- Relació amb la classe “Algorisme”.
- Relació d’agrupació amb la classe “Solucio”.
- Relació amb la classe “CtrlGeneric”.
- Relació amb la classe “CtrlPersistenciaSolucio”.
- Relació amb la classe “CtrlVistaSolucions”.

Nom: Solucio
Descripció de la classe: Representa una distribució òptima de productes en prestatges. Conté una matriu que organitza els productes en un ordre específic, i cada solució es defineix per un nom únic.
Cardinalitat: Una per cada solució creada.
Descripció atributs:
- protected ArrayList<ArrayList<String>> solucio: Matriu que conté els noms dels productes en l’ordre de distribució òptima. Cada fila representa un prestatge amb els productes associats.
- protected String nom: Nom de la solució.

Descripció mètodes:
- Solucio(ArrayList<String> s, String n, int p)
    - s: Llista amb els noms dels productes.
    - n: Nom de la solució.
    - p: Número de productes per prestatge.
    - Descripció: Crea una solució amb el número de prestatges i el nom indicats.
- Solucio(ArrayList<ArrayList<String>> s, String n)
    - s: Matriu amb els noms dels productes.
    - n: Nom de la solució.
    - Descripció: Crea una solució amb la matriu i el nom proporcionats. En cas de que la matriu no sigui vàlida, llença una excepció.
- mostrarSolucio()
    - Descripció: Mostra el nom dels productes d'una solució en ordre. Si la solució no té productes, imprimeix un missatge indicant-ho.
- trobarProducte(String nomProducte)
    - nomProducte: El nom d'un producte.
    - Descripció: Comprova si un producte amb el nom x es troba en la solució. Retorna true si el producte es troba, i false en cas contrari.
- existeixPosicio(int i, int j)
    - i: Índex de la fila de la matriu.
    - j: Índex de la columna de la matriu.
    - Descripció: Comprova si existeix un producte a la posició indicada per i i j dins de la matriu. Retorna true si la posició és vàlida i conté un producte, i false en cas contrari.
- matriuValida(ArrayList<ArrayList<String>> matriu)
    - matriu: Matriu que representa la prestatgeria.
    - Descripció: Comprova que no hi hagi repetits i totes les prestatgeries tenen la mateixa mida. Si no es compleix alguna de les dues condicions, retorna fals.

Descripció de les relacions:

- Relació amb la classe d'agregació "CtrlSolucions".

Nom: SolucioModificada

Descripció de la classe: Representa una distribució de productes concreta que no te perquè ser òptima pels productes que conté.

Cardinalitat: Una per cada solució que es modifiqui.

Descripció Mètodes:

- SolucioModificada(ArrayList<ArrayList<String>> s, String n)
  - s: Matriu amb el nom del productes dins la solució modificada.
  - n: Nom de la solució modificada.
  - Descripció: Constructor de la classe SolucioModificada que inicialitza els atributs.
- intercanvia(String prod1, String prod2)
  - prod1: Nom del primer producte a intercanviar.
  - prod2: Nom del segon producte a intercanviar.
  - Descripció: Intercanvia els productes prod1 i prod2 si ambdós es troben en la solució.
- intercanvia(int index1i, int index1j, int index2i, int index2j)
  - index1i: Índex de la fila del primer producte.
  - index1j: Índex de la columna del primer producte.
  - index2i: Índex de la fila del segon producte.
  - index2j: Índex de la columna del segon
  - Descripció: Intercanvia els productes en les posicions indicades per index1i, index1j i index2i, index2j. Si les posicions no contenen productes vàlids o els productes a intercanviar són els mateixos, llança una excepció IntercanviNoValid.

Descripció de les relacions:

- Relació de subclasse amb la classe "Solucio".
- Relació amb la classe "CtrlSolucions".

Nom: CtrlDomini

Descripció de la classe: Classe de control de domini que es relaciona, amb el driver, per gestionar les instancies de controlador de catàleg amb restriccions i de solucions.

Cardinalitat: Només hi ha un

Descripció Atributs:

- CtrlDomini instancia: Instància única de la classe CtrlDomini.
- CtrlCatalegAmbRestriccions ctrlCatalegAmbRestriccions: Instancia única del controlador de cataleg amb restriccions
- CtrlSolucions ctrlSolucions: Instància única del controlador de solucions
- ctrlPersistencia: Instància única del controlador de persistència

Descripció Mètodes:

Descripció de les relacions:

- Relació d'associació amb la classe Driver.
- Relació d'agregació amb la classe de controlador de solucions.
- Relació d'associació amb la classe de controlador de catàleg amb restriccions.
- Relació d'agregació amb la classe "CtrlPersistencia".

Nom: CtrlCatalegambRestriccions

Descripció de la classe: Subclasse del controlador de catàleg que extendeix les funcionalitats per poder fer ús de les restriccions.
Cardinalitat: Només hi ha un
Descripció Atributs:

- ArrayList<ArrayList<Boolean> noConsecutius: Matriu de booleans per determinar laes similituds entre els productes, on dos productes no poden ser consecutius.

Descripció Mètodes:

- ProducteAfegit()
  - Descripció: Modifica la mida de la matriu de restriccions ja que s'afegit un producte.
- ProducteEliminat(int id) : void
  - id: Atribut que indica el index del producte que se ha eliminat
  - Descripció: Modifica la mida de la matriu de restriccions ja que s'eliminat un producte. A més s'elimina qualsevol restriccions que tingui el producte eliminat.
- setRestConsecId(int id1, int id2) : void
  - id1 : índex del primer producte a afegir restricció.
  - id2 : índex del segón producte a afegir restricció.
  - Descripció: S'afegeix la restricció entre el dos productes donats.
- remRestrConsecId(int id1, int id2) : void
  - id1 : índex del primer producte a eliminar restricció.
  - id2 : índex del segón producte a eliminar restricció.
  - Descripció: S'elimina la restricció entre el dos productes donats
- setRestrConsecNom(String nom1, String nom2) : void
  - nom1 : nom del primer producte a afegir restricció.
  - nom2 : nom del segón producte a afegir restricció.
  - Descripció: S'afegeix la restricció entre el dos productes donats.
- remRestrConsecNom(String nom1, String nom2) : void
  - nom1 : nom del primer producte a eliminar restricció.
  - nom2 : nom del segón producte a eliminar restricció.
  - Descripció: S'elimina la restricció entre el dos productes donats
- getRestrConsecID(int id1, int id2) : Boolean
  - id1: índex del primer producte per trobar restriccions
  - id2: índex del segón producte per trobar restriccions
  - Descripció: Es retorna un boolean que determina si existeix una restricció entre els dos productes donats.
- getMatrRestrConsec() : Boolean[][]
  - Descripció: Retorna la matrius de retriccions en format Boolean[][]
- afegir_producte(String new_nom , Pair<String, Double>) : void
  - Descripció: Reescripció del metode de Controlador Catàleg, només inclou la crida de producteAfegit().
- afegir_producte(String new_nom) : void
  - Descripció: Reescripció del metode de Controlador Catàleg, només inclou la crida de producteAfegit().
- eliminar_producte_index(int index_out) : void
  - Descripció: Reescripció del metode de Controlador Catàleg, només inclou la crida de producteEliminat().
- mostrarProducte(int index) : void

    - Descripció: Reescripció del metode de controlador Catàleg, només inclou la escriptura de les restriccions de cada producte.
- mostrarProducte(String nom) : void
    - Descripció: Reescripció del metode de controlador Catàleg, només inclou la escriptura de les restriccions de cada producte.
- mostrarRestrConsec(int index) : void
    - index : index del producte a mostrar les restriccions:
    - Descripció: Mostra les restriccions del producte concret
- mostrarRestrConsec() : void
    - Descripció: Mostra totes les restriccions de cada producte dins del catàleg.
- restr_a_String() : String[]
    - Descripció: Converteix les restriccions en una llista de strings de text. Cada string representa una parella de productes amb format "producte1;producte2".
- getStringId(ArrayList<Pair<Integer, Integer>> pairs) : ArrayList<Pair<String, String>>
    - Descripció: Converteix una llista de parelles d'índexs de productes a parelles amb els seus noms.
    - pairs (ArrayList<Pair<Integer, Integer>>): Llista de parelles d'índexs de productes.
- getArrayRestr() : ArrayList<Pair<Integer, Integer>>
    - Descripció: Obté les restriccions com a llista de parelles d'índexs de productes que no poden ser consecutius.
- afegir_producte_aux(String nomProd, String[] similituds) : void
    - Descripció: Funció auxiliar per afegir un producte al catàleg.
    - nomProd (String): Nom del producte a afegir.
    - similituds (String[]): Llista de similituds amb els altres productes en format string.
- stringV_a_pairV(String[] similituds) : Pair<String, Double>[]
    - Descripció: Converteix un array de strings que representen similituds en un array de parelles (producte, similitud).
    - similituds (String[]): Un array de strings que conté els valors de similitud en format numèric, un per cada producte existent al catàleg.
- guardarCataleg(String path, String nomArxiu) : void
    - Descripció: Guarda en el fitxer indicat tota la informació del catàleg. Inclou nom de productes, matriu de similituds i matriu de restriccions.
    - path (String): Lloc on està el fitxer.
    - nomArxiu (String): Nom del fitxer on es vol guardar.
- convertirMatriu(boolean[][] matriu) : String
    - Descripció: Converteix una matriu de booleans en una matriu de strings separats per espais, on true és "1" i false és "0". Les files es separen per salt de línia.
    - matriu (boolean[][]): Matriu a convertir.
- carregaCataleg(String[] productes, double[][] similituds, int[][] restriccions) : void
    - Descripció: Carrega del fitxer indicat tota la informació del catàleg. Inclou nom de productes, matriu de similituds i matriu de restriccions.
    - productes (String[]): Nom dels productes en un ordre determinat.
    - similituds (double[][]): Matriu de similituds entre els productes en l'ordre en què es donen.
    - restriccions (int[][]): Matriu de les restriccions entre els productes en l'ordre en què es donen.
- getProdNoRestrConsec(int index) : String[]

    - Descripció: Obté tots els productes amb els que el producte indicat no té restriccions.
    - index (int): ID del producte al catàleg.
- getProdRestrConsec(int index) : String[]
    - Descripció: Obté tots els productes amb els que el producte indicat té restriccions.
    - index (int): ID del producte al catàleg.

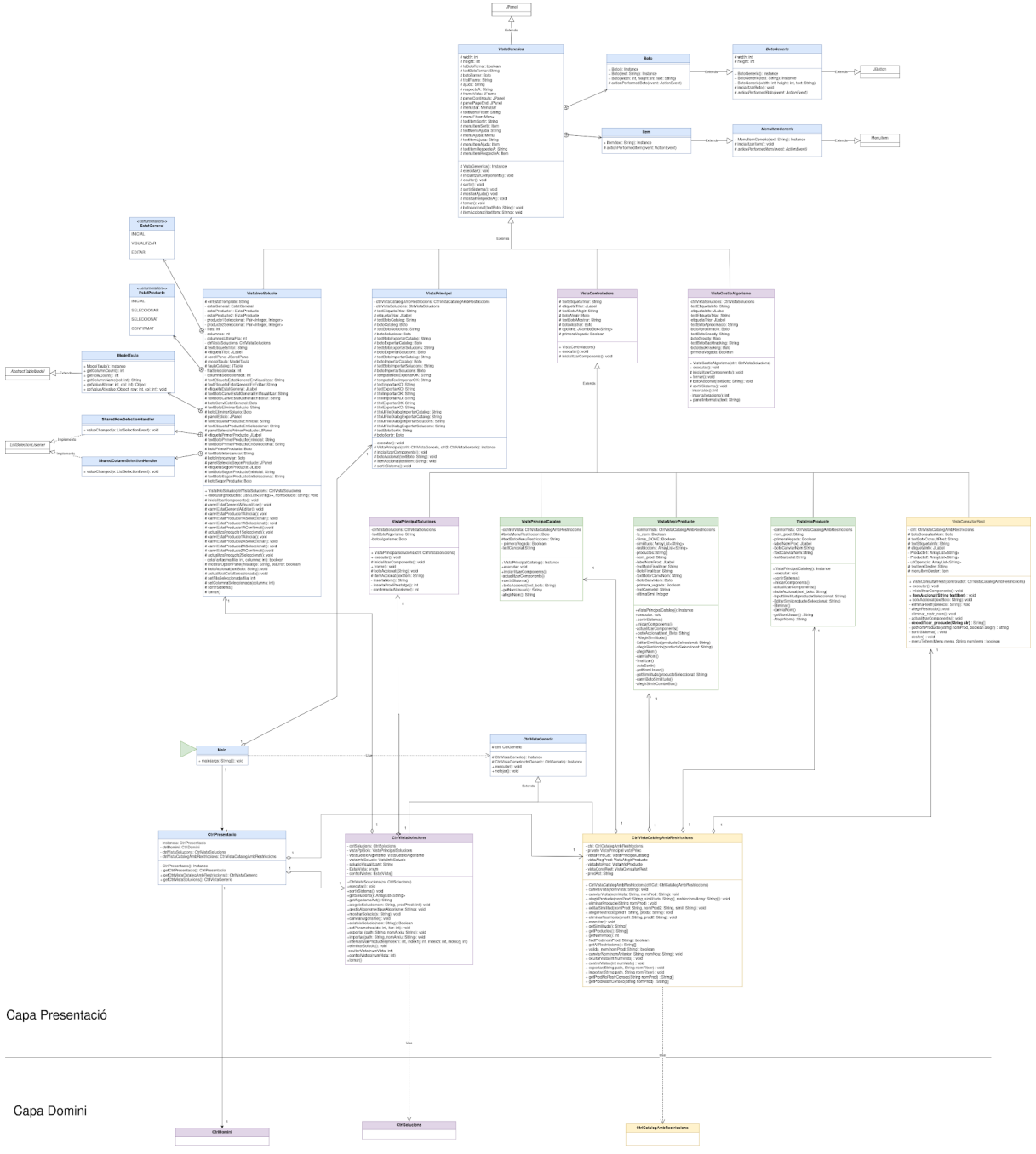
Descripció de les relacions:
- Relació de subclasse amb la classe “Controlador de catàleg”.
- Relació d’agregació amb la classe “Controlador Domini”.

Nom: Driver
Descripció de classe: Classe main del programa, gestionar la interacció amb l’usuari per gestionar les seves accions.
Cardinalitat: Només hi ha un.
Descripció d’atributs:
- CtrlDomini ctrlDomini: Instancia única del controlador de domini.
- int num Gestions: Numero de menus de gestió que es poden escriure
- int[] numAccions: Número d'accions possibles de cada menú
- String FormatNoEnter: String de error input enter esperat no es un enter.
- String FormatNoDecimal: String de error, input decimal esperat no es decimal

Descripció de mètodes:
- executar():
    - Descripció: Mètode principal de execució del programa, que gestiona tota la navegabilitat del programa entre el seus menús.
- imprimirMenuGestio():
    - Descripció: Mostra el menú inicio que indica els diferents menús de gestió del programa
- imprimirMenuAccioGestio1():
    - Descripció: Mostra les diferents accions dintre del menú de gestió de productes
- imprimirMenuAccioGestio2():
    - Descripció: Mostra les diferents accions dintre del menú de gestió de solucions
- imprimirMenuAccioGestio3():
    - Descripció: Mostra les diferents accions dintre del menú de gestió de restriccions
- demanaInt(String err, Scanner scanner)
    - err: String: Missatge d’error correspon al string atribut formatNoEnter
    - scanner Scanner: Funcionalitat per poder rebre el enter del usuari
    - Descripció: S’asegura de poder rebre el enter necesari que li ha de donar el usuari
- demanaDouble(String err, Scanner scanner)
    - err: String: Missatge d’error correspon al string atribut formatNoDecimal
    - scanner Scanner: Funcionalitat per poder rebre el enter del usuari
    - Descripció: S’asegura de poder rebre el decimal necessari que li ha de donar el usuari

Descripció de les relacions:
- Relació associativa amb control de Domini.

3. Diagrama estàtic del model conceptual de dades de la presentació



3.2 Descripció del diagrama

Nom: Main
Descripció de la classe: Proporciona el mètode main que serà invocat a l'executar el programa.
Cardinalitat: Una per a tot el sistema.

Descripció del Atributs: -

Descripció Mètodes:

- main(String[] args)
    - args[]: Una taula amb els paràmetres passats a l'executar el programa. No s'utilitza.
    - Descripció: Dins d'un AWT event-dispatching thread, inicializa la VistaPrincipal passant-li com a paràmetres els controladors pertinents. Més avall (aquí) està explicat en detall.

Descripció de les relacions:

- Relació d'associació amb CtrlPresentacio.
- Relació amb CtrlVistaGeneric.
- Relació d'agregació amb VistaPrincipal.

Nom: CtrlPresentacio
Descripció de la classe: Manté les instàncies dels controladors específics de presentació. També és l'encarregat de demanar a la capa de domini les instàncies dels controladors de domini. És un singleton. Més avall (aquí) està explicat en detall.
Cardinalitat: Una per a tot el sistema.

Descripció del Atributs:

- instancia: La propia instància de la classe seguint el patró singleton
- ctrlDomini: La instància del controlador de domini.
- ctrlVistaSolucions: La instància del CtrlVistaSolucions corresponen a la capa de presentació.
- ctrlVistaCatalegAmbRestriccions: La instància del CtrlVistaCatalegAmbRestriccions corresponen a la capa de presentació.

Descripció Mètodes:

- CtrlPresentacio()
    - Descripció: Declara l'única constructora com a privada per prohibir crear instàncies de la classe. Així garanteix ser singleton.
- getCtrlPresentacio()
    - Descripció: Retorna l'atribut instancia. En cas que aquest sigui null, l'instancia i demana els atributs pertinents.
- getCtrlVistaCatalegAmbRestriccions()

    - Descripció: Retorna l'atribut ctrlVistaCatalegAmbRestriccions.
- getCtrlVistaSolucions()
    - Descripció: Retorna l'atribut ctrlVistaSolucions.

Descripció de les relacions:

- Relació d'associació amb Main.
- Relació d'associació amb CtrlDomini.
- Relació d'agregació amb CtrlVistaSolucions.
- Relació d'agregació amb CtrlVistaCatalegAmbRestriccions.

Nom: CtrlVistaGeneric
Descripció de la classe: Proporciona una estructura comuna dels controladors específics de la capa de presentació que hereden d'ella. També permet reduir l'acoblament. És una classe abstracta. Està explicat amb més detall aquí.
Cardinalitat: No es pot instanciar, és abstracte.

Descripció del Atributs:

- ctrl: Representa el controlador específic de la capa de domini amb el qual les subclasses treballen.

Descripció Mètodes:

- CtrlVistaGeneric()
    - Descripció: Constructora per defecte buida.
- CtrlVistaGeneric(CtrlGeneric ctrlGeneric)
    - ctrlGeneric: El controlador específic de la capa de domini amb el qual les subclasses treballen
    - Descripició: Constructora de la classe. Assigna l'atribut ctrl al paràmetre ctrlGeneric.
- executar()
    - Descripció: Mètode comú en totes les subclasses el qual serà cridat per altres classes perquè el controlador comenci a treballar i mostri la vista
- netejar()
    - Descripció: Mètode comú en totes les subclasses el qual buida les dades de les estructures de dades i reinicia la vista.

Descripció de les relacions: -

Nom: BotoGeneric
Descripció de la classe: Proporciona un esquema de la lògica de control d'esdeveniments que serà comú per tots els botons que heredin d'aquesta classe. És una classe abstracte. Està explicada en detall aquí.
Cardinalitat: No es pot instanciar, és abstracte.

Descripció del Atributs:

- width: És l'amplada del botó en pixels.
- height: És l'alçada del botó en pixels.

Descripció Mètodes:

- BotoGeneric()
  - Descripció: Constructora de la classe. Després d'inicialitzar la superclasse crida al mètode inicialitzarBoto().
- BotoGeneric(String text)
  - text: És el text que es mostrarà com a etiqueta del botó.
  - Descripció: Constructora de la classe. Després d'inicialitzar la superclasse crida al mètode inicialitzarBoto().
- BotoGeneric(int width, int height, String text)
  - width: És l'amplada que tindrà el botó en pixels.
  - height: És l'alçada que tindrà el botó en pixels.
  - text: És el text que es mostrarà com a etiqueta del botó.
  - Descripció: Constructora de la classe. Després d'inicialitzar la superclasse estableix les variables de classe width i height i crida al mètode inicialitzarBoto().
- inicialitzarBoto()
  - Descripció: Inicialitza els components gràfics del botó i afegeix un ActionListener que crida al mètode actionPerformedBoto.
- actionPerformedBoto(ActionEvent event)
  - event: L'event que ha disparat el ActionListener
  - Descripció: És un mètode abstracte. Les subclasses hauran d'implementar aquest mètode.

Descripció de les relacions:

- Relació d'herència amb JButton.

Nom: MenuItemGeneric
Descripció de la classe: Proporciona un esquema de la lògica de control d'esdeveniments que serà comú per tots els items que heredin d'aquesta classe. És una classe abstracte. Està explicada en detall aquí.
Cardinalitat: No es pot instanciar, és abstracte.

Descripció del Atributs: -

Descripció Mètodes:

- MenuItem(String text)
  - text: El text que tindrà com a etiqueta l'ítem.
  - Descripció: Constructora de la classe. Després d'inicialitzar la superclasse crida al mètode inicialitzarItem().
- inicialitzarItem()

    - Descripció: Afegeix un ActionListener que crida al mètode actionPerformedItem.
- actionPerformedItem(ActionEvent event)
    - event: L'event que ha disparat el ActionListener
    - Descripció: És un mètode abstracte. Les subclasses hauran d'implementar aquest mètode.

Descripció de les relacions:

- Relació d'herència amb MenuItem.

Nom: CtrlVistaSolucions
Descripció de la classe: Controlador que actua com a intermediari entre la vista i el controlador de domini "CtrlSolucions". Aquesta classe gestiona les interaccions amb la vista de solucions, permetent obtenir, afegir, modificar, eliminar solucions al sistema, gestionar l'algorisme utilitzat, i carregar o guardar solucions des de fitxers. Explicació detallada a l'apartat 3.2.3. CtrlVistaSolucions.

Cardinalitat: Una en tot el sistema.

Descripció del Atributs:

- ctrlSolucions: Una instància de CtrlSolucions, que és el controlador de domini que gestiona les operacions de solucions en el sistema.
- vistaPplSols: Una instància de la vista VistaPrincipalSolucions, que mostra les solucions principals al sistema.
- vistaGestioAlgorisme: Una instància de la vista VistaGestioAlgorisme, que permet gestionar i canviar l'algorisme actual utilitzat pel sistema.
- vistaInfoSolucio: Instància de la vista que permet mostrar la informació detallada d'una solució específica.
- solucioVisualitzant: Nom de la solució que actualment està sent visualitzada.

Descripció del Mètodes:

- executar()
    - Descripció: Inicialitza i mostra la vista principal de solucions.
- getSolucions()
    - Descripció: Sol·licita al controlador de domini CtrlSolucions que li retorni els noms de les solucions actuals al sistema en forma d'un ArrayList de String.
- getAlgorismeAct()
    - Descripció: Retorna el tipus de l'algorisme actual que s'està utilitzant, mitjançant la crida al controlador CtrlSolucions.
- afegeixSolucio(String nom, int prodPrest)
    - nom: Nom de la nova solució.
    - prodPrest: Nombre de productes per prestatge.
    - Descripció: Intenta afegir una nova solució al sistema amb el nom especificat i el número de productes per prestatge.
- gestioAlgorisme(String tipusAlgorisme)
    - tipusAlgorisme: Tipus del nou algorisme.

    - Descripció: Permet canviar l'algorisme actual, enviant el tipus d'algorisme desitjat al controlador CtrlSolucions.
- mostrarSolucio(String s)
    - s: Nom de la solució que es vol mostrar.
    - Descripció: Mostra la solució corresponent.
- canviarAlgorisme()
    - Descripció: Crida el mètode executar() de la vista VistaGestioAlgorisme, que permet a l'usuari gestionar i canviar l'algorisme actual.
- existeixSolucio(String nom)
    - nom: Nom d'una solució.
    - Descripció: Verifica si existeix una solució al sistema amb el nom especificat. Retorna true si existeix, o false si no existeix.
- setParametres(int idx, int iter)
    - idx: Índex del producte inicial.
    - iter: Nombre d'iteracions.
    - Descripció: Estableix els paràmetres d'algorisme, específicament l'índex i el nombre d'iteracions. Si els paràmetres no són vàlids, captura l'excepció FormatInputNoValid i imprimeix el missatge d'error.
- guardarSolucions(String path, String nomArxiu)
    - path: Ubicació del fitxer.
    - nomArxiu: Nom del fitxer.
    - Descripció: Guarda les solucions actuals del sistema en un fitxer.
- carregarSolucions(String path, String nomArxiu)
    - path: Ubicació del fitxer.
    - nomArxiu: Nom del fitxer.
    - Descripció: Carrega solucions des d'un fitxer al sistema.
- intercanviarProductes(int index1i, int index1j, int index2i, int index2j)
    - index1i, index1j: Índexs del primer producte.
    - index2i, index2j: Índexs del segon producte.
    - Descripció: Intercanvia dos productes en una solució.
- eliminarSolucio()
    - Descripció: Elimina la solució actualment visualitzada.

Descripció de les relacions:

- Subclasse de “CtrlVistaGeneic”.
- Relació amb “CtrlSolucions”.
- Relació d’agregació amb “VistaPrincipalSolucions”.
- Relació d’agregació amb “VistaInfoSolucio”.
- Relació d’agregació amb “VistaGestioAlgorisme”.

Nom: CtrlVistaCatalegAmbRestriccions
Descripció de la classe: Subcontrolador de la capa de presentació que s’encarrega de gestionar el flux entre les vistes de gestió catàleg i comunicar els inputs amb la capa de domini.

Cardinalitat: Només hi ha un.
Descripció Atributs:

- CtrlCatalegAmbRestriccions ctrl: Instància del controlador del catàleg amb restriccions.
- VistaPrincipal vistaPrinc: Vista principal de l'aplicació.
- VistaPrincipalCataleg vistaPrincCat: Vista principal del catàleg.
- VistaAfegirProducte vistaAfegProd: Vista per afegir productes al catàleg.
- VistaInfoProducte vistaInfoProd: Vista per consultar informació d'un producte concret i modificar-ho.
- VistaConsultarRest vistaConsRest: Vista per consultar les restriccions del catàleg i modificar-les.
- String prodAct: Nom del producte actiu (en focus).

Descripció Mètodes:

- CtrlVistaCatalegAmbRestriccions(CtrlCatalegAmbRestriccions ctrlCat) : void
    - Descripció: Constructor de la classe.
    - ctrlCat (CtrlCatalegAmbRestriccions): Instància del controlador del catàleg amb restriccions.
- canviaVista(String nomVista) : void
    - Descripció: Canvia la vista actual a una altra vista segons el nom indicat.
    - nomVista (String): Nom de la vista a la qual es vol canviar.
- canviarVista(String nomVista, String nomProd) : void
    - Descripció: Canvia a una vista que requereix informació d'un producte específic.
    - nomVista (String): Nom de la vista a la qual es vol canviar.
    - nomProd (String): Nom del producte per obtenir informació específica.
- afegirProducte(String nomProd, String[] similituds, String[] restriccionsArray) : void
    - Descripció: Afegeix un producte al catàleg juntament amb les seves similituds i restriccions.
    - nomProd (String): Nom del producte a afegir.
    - similituds (String[]): Llista de similituds del producte amb altres productes.
    - restriccionsArray (String[]): Llista de restriccions associades al producte.
- eliminarProducte(String nomProd) : void
    - Descripció: Elimina un producte del catàleg.
    - nomProd (String): Nom del producte que es vol eliminar.
- editarSimilitud(String nomProd, String nomProd2, String simil) : void
    - Descripció: Modifica la similitud entre dos productes.
    - nomProd (String): Nom del primer producte.
    - nomProd2 (String): Nom del segon producte.
    - simil (String): Nova similitud a establir entre els dos productes.
- afegirRestriccio(String prod1, String prod2) : void
    - Descripció: Afegeix una restricció de consecutivitat entre dos productes.
    - prod1 (String): Nom del primer producte.
    - prod2 (String): Nom del segon producte.
- eliminarRestriccio(String prod1, String prod2) : void
    - Descripció: Elimina una restricció de consecutivitat entre dos productes.
    - prod1 (String): Nom del primer producte.
    - prod2 (String): Nom del segon producte.

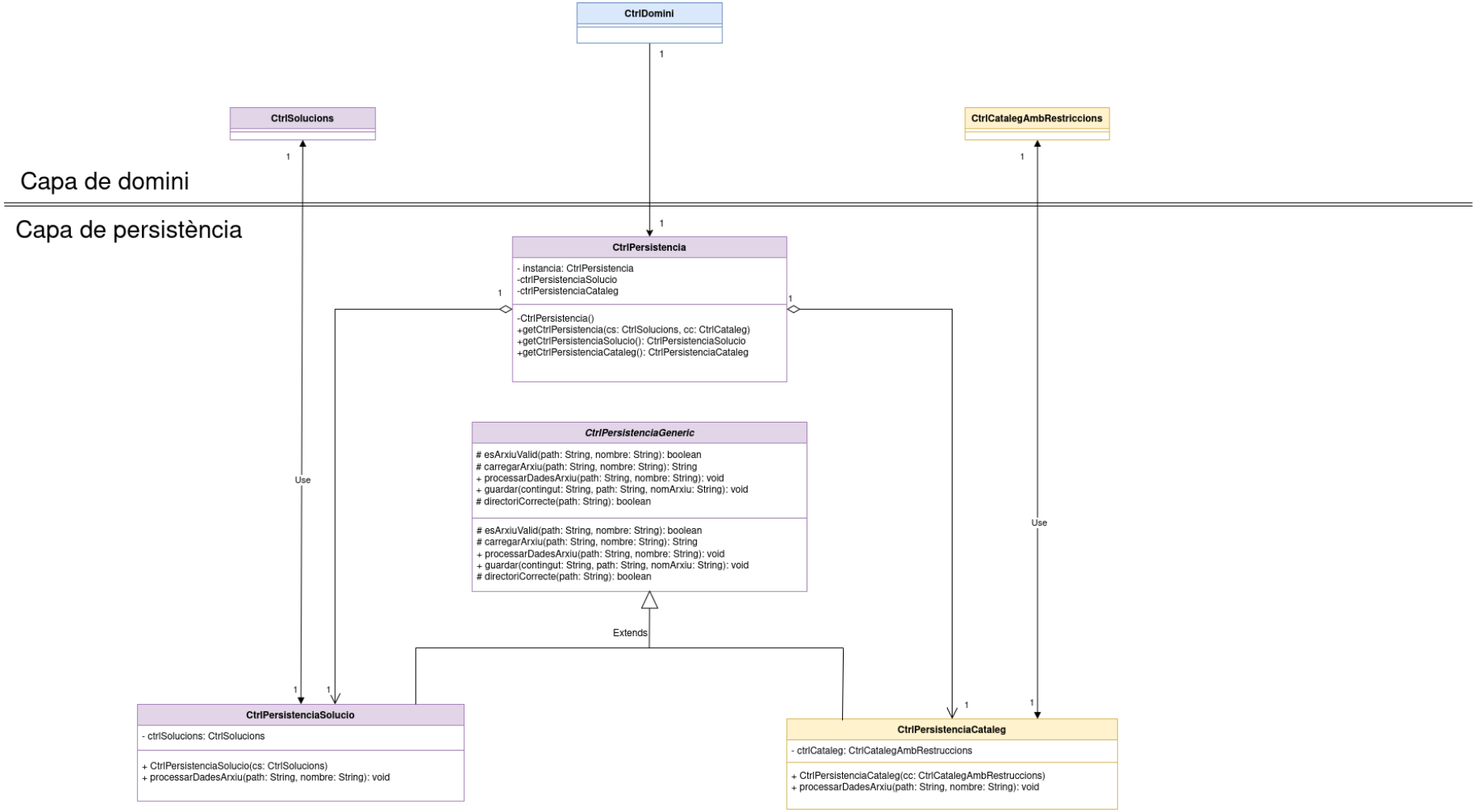
- executar() : void
    - Descripció: Executa la vista principal del catàleg.
- getSimilituds() : String[]
    - Descripció: Obté les similituds del producte actiu amb la resta de productes del catàleg.
- getProductes() : String[]
    - Descripció: Obté tots els noms dels productes registrats al catàleg.
- getNumProd() : int
    - Descripció: Obté el nombre total de productes al catàleg.
- findProd(String nomProd) : boolean
    - Descripció: Verifica si un producte amb un nom específic existeix al catàleg.
    - nomProd (String): Nom del producte a cercar.
- getAllRestriccions() : String[]
    - Descripció: Obté totes les restriccions disponibles en format de cadena de text.
- valida_nom(String nomProd) : boolean
    - Descripció: Valida si el nom del producte especificat existeix al sistema.
    - nomProd (String): Nom del producte a validar.
- canviarNom(String nomAnterior, String nomNou) : void
    - Descripció: Canvia el nom d'un producte del catàleg.
    - nomAnterior (String): Nom antic del producte.
    - nomNou (String): Nou nom del producte.
- ocultarVista(int numVista) : void
    - Descripció: Oculta la vista corresponent en funció de l'identificador proporcionat.
    - numVista (int): Identificador de la vista a ocultar.
- controlVistes(int numVista) : void
    - Descripció: Mostra la vista corresponent i amaga la resta de vistes.
    - numVista (int): Identificador de la vista a mostrar.
- exportar(String path, String nomFitxer) : void
    - Descripció: Guarda el catàleg actual del sistema en un fitxer especificat.
    - path (String): Lloc on està el fitxer.
    - nomFitxer (String): Nom del fitxer on es vol guardar.
- importar(String path, String nomFitxer) : void
    - Descripció: Carrega el catàleg des d'un fitxer especificat al sistema.
    - path (String): Lloc on està el fitxer.
    - nomFitxer (String): Nom del fitxer des del qual es vol carregar.
- getProdNoRestrConsec(String nomProd) : String[]
    - Descripció: Obté tots els productes amb els quals el producte indicat no té restriccions.
    - nomProd (String): Nom del producte.
- getProdRestrConsec(String nomProd) : String[]
    - Descripció: Obté tots els productes amb els quals el producte indicat té restriccions.
    - nomProd (String): Nom del producte.

Descripció de les relacions:

- Subclasse de “CtrlVistaGeneric”
- Relació amb “CtrlCatalegAmbRestriccions”
- Relació d’agregació amb “VistaInfoProducte”

- Relació d’agregació amb “VistaPrincipalCataleg”
- Relació d’agregació amb “VistaAfegirProducte”
- Relació d’agregació amb “VistaConsultarRest”

4. Diagrama estàtic del model conceptual de dades de la persistència



4.2 Descripció del diagrama

Nom: CtrlPersistencia
Descripció de la classe: és un controlador que gestiona la persistència de dades del sistema. Aquesta classe actua com un punt central per accedir a controladors específics encarregats de la gestió d'arxius i de la validació de dades. És responsable de garantir que només hi hagi una instància de cada controlador de persistència, assegurant així el patró *singleton*.
Cardinalitat: Una en tot el sistema.

Descripció del Atributs:

- instancia: Instància única de la classe CtrlPersistencia.
- CtrlPersistenciaSolucio ctrlPersistenciaSolucio: Instància única del controlador encarregat de gestionar la persistència de les solucions.
- CtrlPersistenciaCataleg ctrlPersistenciaCataleg: Instància única del controlador encarregat de gestionar la persistència dels productes.

Descripció del Mètodes:

- CtrlPersistencia()
  - Descripció: Constructor privat de la classe. Impedeix que es creïn múltiples instàncies, assegurant el patró *singleton*.
- getCtrlPersistencia(cs, cg): Constructor privat de la classe. Impedeix que es creïn múltiples instàncies, assegurant el patró *singleton*.
  - cs: Instància del controlador de solucions (CtrlSolucions).
  - cg: Instància del controlador de catàleg amb restriccions (CtrlCatalegAmbRestriccions).
  - Descripció: Retorna la instància única de CtrlPersistencia. Si aquesta no existeix, la crea inicialitzant també el controlador de persistència de solucions (CtrlPersistenciaSolucio) amb l'instància de CtrlSolucions passada com a paràmetre.
- getCtrlPersistenciaSolucio()
  - Descripció: Retorna l'instància única de CtrlPersistenciaSolucio, el controlador encarregat de gestionar la persistència de les solucions.
- getCtrlPersistenciaCataleg()
  - Descripció: Retorna l'instància única de CtrlPersistenciaCataleg, el controlador encarregat de gestionar la persistència dels productes.

Descripció de les relacions:

- Relació amb "CtrlDomini".
- Relació d'agregació amb "CtrlPersistenciaSolucio".
- Relació d'agregació amb "CtrlPersisitenciaCataleg".

Nom: CtrlPersistenciaGeneric
Descripció de la classe: classe abstracta que proporciona funcionalitats genèriques per gestionar la persistència de dades al sistema. Aquesta classe inclou mètodes per validar directoris i fitxers,

carregar dades des d'arxius, i guardar dades al sistema. Explicació més detallada aquí.
Cardinalitat: Una en tot el sistema.

Descripció del Mètodes:

- esArxiuValid(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.
    - Descripció: Verifica si un fitxer existeix, és accessible i es pot llegir. Retorna true si el fitxer és vàlid, false altrament.
- carregarArxiu(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.
    - Descripció: legeix i retorna el contingut d'un fitxer si aquest és vàlid.
- processarDadesArxiu(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.
    - Descripció: Valida que el directori sigui correcte i prepara el fitxer per al seu processament.
- guardar(String contingut, String path, String nomArxiu)
    - contingut: Dades que es volen guardar al fitxer.
    - path: Ruta on es vol guardar el fitxer.
    - nomArxiu: Nom del fitxer.
    - Descripció: Desa les dades especificades en un fitxer al directori indicat.
- directoriCorrecte(String path)
    - path: Ruta del directori.
    - Descripció: Verifica que un directori existeixi, sigui vàlid i es pugui escriure.

Descripció de les relacions: -

Nom: CtrlPersistenciaSolucio
Descripció de la classe: controlador especialitzat en gestionar la persistència de solucions dins del sistema. Explicació més detallada a l'apartat de controladors.
Cardinalitat: Una en tot el sistema.

Descripció del Atributs:

- ctrlSolucions: Instància del controlador de solucions que s'utilitza per gestionar les dades carregades des dels fitxers.

Descripció del Mètodes:

- CtrlPersistenciaSolucio(CtrlSolucions cs)
    - cs: Instància del controlador de solucions (CtrlSolucions).

    - Descripció:Constructor que inicialitza l'atribut ctrlSolucions amb la instància proporcionada.
- processarDadesArxiu(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.
    - Descripció: Processa les dades d'un fitxer de solucions. Valida el directori, carrega el contingut del fitxer, i divideix les dades en línies per extreure informació sobre cada solució.

Descripció de les relacions:

- Relació de subclasse amb "CtrlPersistenciaGeneric".
- Relació d'agregació amb "CtrlPersistencia".
- Relació amb "CtrlSolucions".

Nom: CtrlPersistenciaCataleg
Descripció de la classe: controlador especialitzat en gestionar la persistència dels productes dins del sistema.
Cardinalitat: Una en tot el sistema.

Descripció del Atributs:

- ctrlCataleg: Instància del controlador de catàleg que s'utilitza per gestionar les dades carregades des dels fitxers.

Descripció del Mètodes:

- CtrlPersistenciaCataleg(CtrlCatalegAmbRestruccions cc)
    - cc: Instància del controlador de catàleg.
    - Descripció:Constructor que inicialitza l'atribut ctrlCataleg amb la instància proporcionada.
- processarDadesArxiu(String path, String nombre)
    - path: Ruta del fitxer.
    - nombre: Nom del fitxer.
    - Descripció: Processa les dades d'un fitxer de productes. Valida el directori, carrega el contingut del fitxer, i divideix les dades en línies per extreure informació sobre cada producte.

Descripció de les relacions:

- Subclasse de CtrlPersistenciaGeneric.
- Relació amb CtrlCatalegAmbRestriccions.
- Relació d'agregació amb "CtrlPersistencia".

5. Classes de la capa de presentació

En la capa de presentació del nostre projecte, per una banda, hem prioritzat la reutilització de codi al màxim i el principi obert tancat. Per aconseguir-ho, explotem al màxim l'herència entre classes quan aquestes presenten patrons comuns. Per l'altra banda, s'ha prioritzat la usabilitat i l'experiència d'usuari (UX). Les vistes han estat dissenyades de tal forma que presenten una integritat estètica, ergonomia, i sempre mostren ajuda quan l'usuari ho necessita. En cas que hi hagi errors o avisos, es mostren de forma clara i entenedora cap a l'usuari. Així doncs, l'usuari pot sentir-se còmode i familiar a mesura que va experimentant amb la interfície gràfica.

5.1 Vistes

5.1.1 VistaGenèrica

Aquesta classe hereta de JPanel i és abstracte. La finalitat d'aquesta vista genèrica és doble. Per una banda, d'igual forma que amb el botó genèric i ítem genèric, ens vam adonar que totes les vistes seguien el mateix procediment: crear l'objecte JFrame, afegir-li un títol, afegir la barra de menú amb els submenús i ítems pertinents, afegir un botó de tornar enrere… Per tant, la vista genèrica defineix l'esquema general que totes les vistes genèriques segueixen. Defineix mètodes i atributs com a protected per tal que es puguin redefinir. Vam decidir que totes les vistes tindrien un mètode comú executar que inicialitza i mostra la vista i un mètode comú inicialitzarComponents per iniciar tots els components. El mètode inicialitzarComponents de la vista genèrica estableix un esquema comú preparant el JPanel (de l'herència) com a contenidor central des d'on les subclasses afegiran els seus component. S'ha d'executar abans que el de les subclasses. Per l'altra banda, defineix un disseny comú per a totes les vistes, garantint una consistència de l'aplicació.
La vista genèrica té niuada la classe Boto que hereta de BotoGeneric i implementa el mètode actionPerformedBoto. Aquest mètode obté el text del botó (getText()) i fa una crida al mètode botoAccionat(String textBoto) de la classe VistaGenerica. Així doncs, sigui quin sigui el botó clicat, s'acaba executant el mètode botoAccionat(String textBoto) i només cal diferenciar pel textBoto per saber quin ha estat clicat. Les subclasses de VistaGenerica poden utilitzar lliurement instàncies de Boto i hauran d'implementar aquest mètode. De forma anàloga, vam fer el mateix amb la classe niuada Item que hereta d'ItemGeneric.

Així doncs, la vista genèrica segueix el principi obert tancat, el DRY, proporciona una consistència estètica a l'aplicació i mètodes únics per tots els botons i ítems clicats.

5.1.2 VistaPrincipal

Aquesta vista hereta de VistaGenerica i s'encarrega de mostrar el menú principal on es pot escollir quina funcionalitat es vol realitzar. És cridada des del Main, és a dir, és la primera vista a executar-se. Té les instàncies de CtrlVistaSolucions i CtrlVistaCatalegAmbRestriccions per poder fer peticions. Segueix el mateix esquema que el Driver que teníem en la primera entrega. Mitjançant botons, es pot escollir entre:

- Gestionar el catàleg i les restriccions: Agrupa les funcionalitats alta/baixa/modificació/consulta de productes, alta/baixa/modificació/consulta de similituds entre productes i alta/baixa/consulta de restriccions entre productes. Demana a la instància de CtrlVistaCatalegAmbRestriccions que s'executi la vista pertinent.

- Gestionar les solucions: Agrupa les funcionalitats: càlcul d'una distribució segons un algorisme a determinar i consulta/modificació/baixa de solucions existents. Demana a la instància de CtrlVistaSolucions que s'executi la vista pertinent
- Exportar a fitxer: Representa la funcionalitat de guardar les dades en fitxer. Demana a les respectives instàncies de CtrlVistaCatalegAmbRestriccions i de CtrlVistaSolucions que guardin les seves dades en fitxer.
- Importar a fitxer: Representa la funcionalitat de llegir les dades des de fitxer. Demana a les respectives instàncies de CtrlVistaCatalegAmbRestriccions i de CtrlVistaSolucions que llegeixin les seves dades des de fitxer.

5.1.3 VistaInfoSolucio

Aquesta vista hereta de VistaGenèrica i s'especialitza en les funcionalitats consulta/modificació/baixa de solucions existents. És cridada des del controlador CtrlVistaSolucions i té la instància de CtrlVistaSolucions. La vista està agrupada en dues seccions: la primera secció mostra una taula JTable amb els productes de la solució simbolitzant la prestatgeria, la segona secció és un panel d'edició on es poden fer els intercanvis de productes.

5.1.3.1 La taula

La vista rep com a paràmetre en el mètode executar una llista de llistes amb els noms dels productes que correspon a la geometria de la distribució. Per mostrar aquesta llista, la JTable ens va semblar la més adequada. Per defecte, la JTable manté una lògica de model on duplica la matriu Object[][] la qual es passa en la constructora i utilitza internament Vector. Tanmateix, la JTable permet aïllar la lògica de model de la presentació de la taula implementant la classe AbstractTableModel i passant-li una instància a la constructora. Així el model pot treballar directament i eficientment amb la llista de llistes (vegeu https://docs.oracle.com/javase/8/docs/api/javax/swing/table/TableModel.html). És per això que la classe VistaInfoSolucio té niuada la classe ModelTaula que hereta de AbstractTableModel.

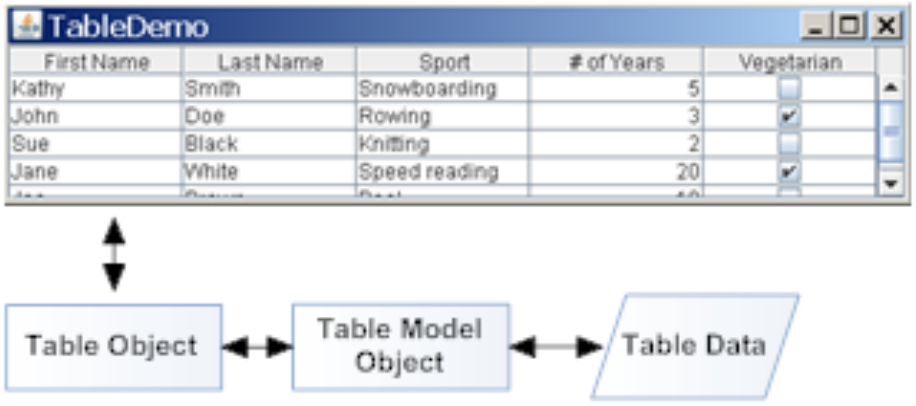


Figura 5: Lògica de la JTable (extreta de https://docs.oracle.com/javase/tutorial/uiswing/components/table.html).

Per altra banda, calia afegir els Listener pertinents per detectar les cel·les que són clicades. Donat que la JTable treballa internament amb JList, vàrem crear les classes niuades a VistaInfoSolucio

anomenades SharedRowSelectionHandler i SharedColumnSelectionHandler que implementen la interfície ListSelectionListener. Aquestes classes s'instancien i es passen com a listeners a la taula. Quan una casella es clica, els listeners informen l'esdeveniment a la classe principal (VistaInfoSolucio).

5.1.3.2 La lògica d'estats

Els productes es poden seleccionar directament clicant en la taula. Per aconseguir això, ens vam adonar que hi ha moltes fases: iniciar l'intercanvi de dos productes, seleccionar el primer producte, permetre canviar el primer producte seleccionat, confirmar el primer producte… I d'igual forma pel segon producte. A més a més, la selecció de productes ha de ser concurrent, és a dir, no s'ha de fer esperar acabar la selecció del primer producte per iniciar la del segon producte. Per fer-ho escalable i coherent, vam decidir implementar estats amb enumeracions de Java. Hi ha un estat general que indica si estem en mode visualitzar, editar (permet l'intercanvi de productes) o inicial (just acaba d'obrir-se la vista, és equivalent a visualitzar). Només quan ens trobem en mode editar, inicia el graf d'estat pels dos productes a seleccionar. Els estats són:

1. Inicial: No s'ha començat la selecció. Cal clicar un botó per començar-la
2. Seleccionar: L'usuari ha de seleccionar un producte de la taula
3. Seleccionat: L'usuari ha seleccionat un producte de la taula, però pot seleccionar un altre per canviar-lo (per exemple si s'ha equivocat). Quan estigui segur pot clicar el botó per confirmar el producte.
4. Confirmar: El producte s'ha confirmat.

Aquests estats han de treballar de forma paral·lela per cadascun dels productes. És a dir, cada un dels dos productes per seleccionar té el seu graf d'estats.

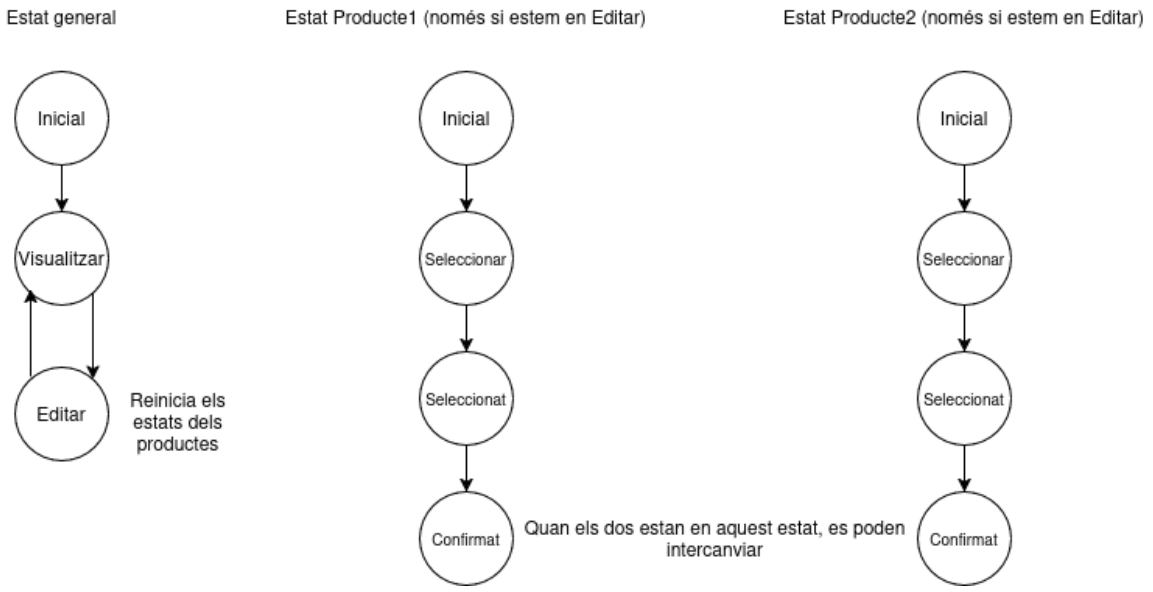


Figura 6: Diagrames d'estats

5.1.4 VistaConsultarRest

Des d'aquesta vista es poden consultar totes les restriccions presents al catàleg, així com eliminar i afegir-ne. Conserva l'esquema de "vistaControladors", la qual actua de superclasse, on es reimplementa el botó mostrar per donar la funcionalitat d'eliminar la restricció seleccionada en aquell moment. A més, s'afegeix un de nou per tal de poder eliminar restriccions entre productes de manera més pràctica si la llista és massa gran, a més de consultar les restriccions de productes individuals.

Així doncs, es pot afegir una restricció entre productes via escrivint el nom d'ambdós en subseqüents subvistes aniuades amb opció per un input de text des de teclat o selecció al *comboBox*, així com eliminar-les de la mateixa manera. A més a més, s'inclouen subvistes de confirmació i d'advertència.

La vista i selecció es dóna a través d'un *comboBox* que hereda de la superclasse, mostrant en cada element d'aquest una parella de productes que tenen restringit succeir-se consecutivament a una solució.

D'aquesta forma, la vista agrupa tres funcionalitats: consultar les restriccions actuals a través del *comboBox* ja sigui individualment o en general, crear de noves via el botó d'afegir i esborrar-ne a través dels botons d'eliminar, ja sigui la seleccionada o la introduïda per text.

A més, la vista implementa un item al menú Fitxer anomenat "Desfer" que permet modificar els últims canvis fets.

5.1.5 VistaPrincipalCatàleg

Aquesta vista presenta els diversos productes que es troben al catàleg, com també un menú interactiu d'accions relacionades amb el catàleg i la seva gestió.

Aquesta vista hereta funcionalitats de la vista de controladors. Fa ús d'un *comboBox*, per mostrar a l'usuari els diversos productes del catàleg, amb la capacitat de poder seleccionar un d'aquests productes. Dels productes només es mostra el seu nom, que l'identifica.

El menú interactiu, obre entrada a les tres funcionalitats principals del catàleg, està organitzat amb tres botons, dos d'ells heretats de la vista controladors i la resta creats en aquesta vista a partir de la classe de botó genèric, el primer "Mostrar Producte", "Afegir Producte", "Consultar Restriccions" Les funcionalitats de tot tres components, són les següents.

Mostrar producte, canvia a la vista *VistaInfoProducte* per mostrar la informació del producte seleccionat dins del comboBox. Afegir Producte, canvia a la *VistaAfegirProducte*, Consultar restriccions, canvia a la *VistaConsultarRest* per realitzar accions respecte a les restriccions dins del catàleg.

Un cas alternatiu d'aquesta vista es dona el cas d'afegir un primer producte al catàleg, prenent el botó "Afegir Producte", ja que, com només serà necessari que l'usuari doni el nom del nou producte, aquest es demana directament en aquesta vista. Amb l'ús d'un panel d'entrada d'informació a l'usuari i redirigint l'ordre de creació del producte al controlador de presentació del catàleg. Per tant, es dona un funcionament alternatiu d'afegir productes o no es crida o és canvia a la *VistaAfegirProductes*.

Aquesta vista s’encarrega dons de cobrir l'entrada a les diverses vistes que implementen els casos d'ús del catàleg. Aquesta vista només té com a utilitat, ser un menú per poder entrar a la resta de vistes del catàleg de forma més organitzada, com també dona la possibilitat a l'usuari de veure tots els productes de manera global.

5.1.6 VistaInfoProducte

Aquesta vista presenta la informació detallada d’un producte concret, exactament mostra el seu nom, les seves similituds amb la resta de productes del catàleg. Per altra banda, dona la possibilitat a l'usuari de modificar aquests valors com també d'eliminar el producte.

Aquesta vista hereta funcionalitats de la vista controladors, els components heredats com el combobox els utilitza per mostrar les similituds amb la resta de productes, amb format <nom_producte> -> <similitud>. Es permet seleccionar producte. Els dos botons heretats de vista controlador serveixen per eliminar el producte que es veu actualment, amb una confirmació prèvia, i l’altre per editar la similitud entre el producte de la vista amb el producte seleccionat al combobox a partir d'un panel d'entrada d'informació de l'usuari. Un botó afegit dins de la vista dona la possibilitat de modificar el nom del producte que es veu a través d'un panel d'entrada de valors de l'usuari. La vista també disposa d'un label de text per mostrar el nom del producte que s'està mostrant.

Aquesta vista s’encarrega de cobrir els principals casos d'ús relacionats amb la consulta de dades del catàleg, com també la modificació de productes del catàleg, la seva informació i la seva existència dins del catàleg.

5.1.7 VistaAfegirProductes

En aquesta vista és permet a l'usuari crear un producte i afegir-lo dins del catàleg. A partir d'un menú interactiu i panells d'entrada de valors.

Aquesta classe torna a heretar funcionalitats de la vista controladors, el combobox ens permet visualitzar els productes que ja es troben al catàleg, per poder seleccionar-los per editar similituds i restriccions, com també perquè l'usuari pugui consultar els noms preexistents. Els dos botons heretats, el primer, “Afegir Restricció”, que permet a l'usuari crear una restricció entre el producte seleccionat i el producte nou, el segon, “Afegir Similituds”, que permet l’usuari iniciar el procés interactiu per afegir totes les similituds amb la resta de productes a partir de panells d'entrada de valors un per cada similitud. A més es creen dos botons, “Canviar Nom”, permet canviar el nom del producte que és es vol afegir i “Guardar”, que guarda i afegeix el producte junt amb tota la seva informació donant-li al controlador de presentació de catàleg amb restriccions. Per últim, hi ha un label que mostra el nom actual del nou producte.

La funcionalitat per crear un producte és la següent, inicialment al moment de clicar al botó “Afegir Productes” de la vista principal de catàleg, la vista afegir productes crea un panel d'entrada de valors perquè l'usuari doni el nom del producte que vol crear, que estarà restringit, ja que no podrà ser igual a un altre nom del catàleg. Un cop donat es fa visible la vista on a l'usuari podrà fer ús dels botons per

anar afegint informació, necessàriament haurà de fer afegir similituds al producte. Un cop tota la informació obligatòria sigui donada podrà guardar i crear el producte, aquesta informació és el nom i les similituds del producte. Si l'usuari tornes enrere o tanques la vista se li avisa que tota la informació es perdrà, ja que és el botó guardar és el que afegeix el producte.

Una funcionalitat alternativa en aquesta vista, es dona un cop s'ha afegit totes les similituds, perquè l'usuari tindrà la capacitat a partir del botó d'afegir similituds, que ara mostrarà el seu nou nom d'"Editar Similituds", de poder canviar alguna similitud que hagi donat abans de guardar el producte, aquesta similitud s'escogeix amb la selecció d'un producte al combobox.

Aquesta vista cobreix només el cas d'ús d'afegir un producte dins del catàleg. Encara que aquesta funcionalitat és extensa i requereix diversas funcionalitats. Recordem que afegir un primer producte no es fa en aquesta vista sinó al principal, ja que només es requereix el nom del producte.

5.1.8 VistaControladors

La classe VistaControladors és una classe abstracta que actua com a base per a VistaPrincipalCataleg i VistaPrincipalSolucio ja que ens en vam adonar que comparteixen moltes funcionalitats. Aquesta classe permet que les subclasses heretin i personalitzin aquestes funcionalitats segons les seves necessitats específiques.

Com a subclasse de VistaGenèrica, hereta el comportament fonamental de gestió de la interfície gràfica i el mecanisme per inicialitzar i executar vistes.

Els atributs definits, com ara textEtiquetaTriar, etiquetaTriar, botoAfegir, botoMostrar i opcions, proporcionen la base gràfica comuna per a les subclasses. Aquests elements permeten mostrar informació, afegir nous elements al sistema, i seleccionar entre diverses opcions mitjançant un JComboBox. El camp primeraVegada assegura que la vista només s'inicialitzi una vegada, evitant duplicacions o sobrecàrregues innecessàries.

Aquesta estructura comuna facilita la consistència entre les vistes principals del sistema, alhora que permet la personalització i extensió per a funcionalitats específiques.

5.1.9 VistaPrincipalSolucions

La classe VistaPrincipalSolucions proporciona a l'usuari una interfície gràfica intuïtiva per gestionar les solucions del sistema. Permet realitzar operacions com crear, modificar i visualitzar solucions, així com gestionar l'algorisme actiu. Aquesta vista actua com un pont entre l'usuari i el controlador CtrlVistaSolucions.

A més, la vista hereta el component comboBox "opcions", que es configura per mostrar totes les solucions actuals del sistema. Mitjançant els botons disponibles, l'usuari pot indicar de manera clara i intuïtiva si vol crear una nova solució, visualitzar-ne una existent, o canviar l'algorisme actual.

En qualsevol moment, l'usuari té la possibilitat d'abortar la creació d'una solució i tornar enrere sense que es guardin els canvis realitzats.

5.1.10 VistaGestioAlgorisme

La vista VistaGestioAlgorisme permet a l'usuari consultar i gestionar l'algorisme actualment actiu al sistema. Mitjançant tres botons clarament identificats que representen les opcions disponibles: Aproximació, Greedy i Backtracking. Quan l'usuari selecciona un nou tipus d'algorisme, la vista comunica aquesta acció al controlador CtrlVistaSolucions, que s'encarrega de processar la sol·licitud i aplicar els canvis al sistema.

.

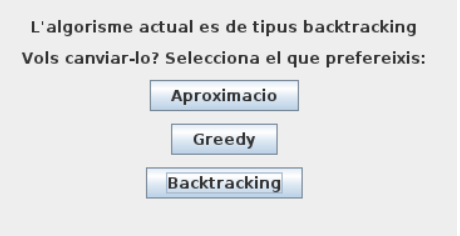


Figura 7: botons per seleccionar el tipus d'algorisme

En el cas específic de l'algorisme Greedy, el sistema requereix que l'usuari introdueixi dos paràmetres addicionals: l'índex del producte inicial i el nombre d'iteracions necessàries per generar un resultat. La vista guia l'usuari durant aquest procés, assegurant-se que les dades es validin correctament abans de continuar. Fins que l'usuari no introdueix tots els paràmetres requerits, el procés no es pot completar.

No obstant això, la vista ofereix flexibilitat, permetent a l'usuari abortar el procés o tornar enrere en qualsevol moment si decideix no continuar amb el canvi d'algorisme.

5.2. Controladors

5.2.1 CtrlVistaGeneric

Aquesta classe és abstracta. Defineix els mètodes i atributs que han de tenir cada un dels controladors de la capa de presentació. Permet complir els principis d'obert tancat a la vegada que permet reduir acoblament generalitzant els controladors.

5.2.2 CtrlPresentacio

Aquesta classe està implementada com a singleton. És el controlador principal de la capa de presentació. El seu objectiu és mantenir les instàncies de tots els controladors de la presentació, que en el nostre cas són CtrlVistaSolucions i CtrlVistaCatalegAmbRestriccions. D'aquesta forma garanteix que els altres controladors de la presentació també són singletons. També és l'encarregada de comunicar-se amb CtrlDomini i demanar-li les instàncies dels controladors específics del domini.

5.2.3 CtrlVistaSolucions

La classe CtrlVistaSolucions és un controlador que actua com a intermediari entre la vista de solucions i el controlador de domini CtrlSolucions, i això permet gestionar de manera eficient totes les operacions relacionades amb les solucions al sistema. Aquesta classe facilita la comunicació entre les diferents capes del sistema, assegurant que les accions realitzades per l'usuari a través de la interfície gràfica es tradueixin en operacions adequades al domini.

5.2.4 CtrlVistaCatalegAmbRestriccions

Aquest controlador s'encarrega de comunicar les vistes relacionades amb el catàleg amb el controlador de domini corresponent, el controlador de catàleg amb restriccions. Serveix de via d'accés dels inputs a la interfície gràfica de la capa de presentació cap a la capa de domini, així com per rebre les dades de la capa de persistència que han de ser mostrades a les vistes. Permet tenir, a més, una gestió eficient de les vistes i de les operacions implicades en cadascuna d'aquestes.

5.3. Altres classes

5.3.1. Main

La classe Main conté el métode main(String[] args) que dona principi a l'execució del programa. Dins d'un AWT event-dispatching thread, el primer que fa és demanar-li al singleton CtrlPresentacio les instàncies de CtrlVistaSolucions i CtrlVistaCatalegAmbRestriccions però guardant-les com a CtrlVistaGeneric (la superclasse) per evitar acoblament innecessari. Seguidament, crea una instància de VistaPrincipal i l'executa passant les instàncies dels controladors com a paràmetres.

5.3.2. BotoGeneric

Aquesta classe hereta de JButton i és abstracte. Totes les vistes de la pràctica tenen diversos botons, per a cada botó inicialment els instanciàvem com a JButton i afegíem els ActionListener pertinents. Ens vam adonar que tots seguien el mateix procediment: crear l'objecte, afegir un ActionListener i crear el mètode actionPerformedBotoX que s'executa quan succeeix l'acció (per exemple clicar). És per això que vam decidir crear una classe abstracta que heretes de JButton i fes aquest procediment comú. El mètode actionPerformedBoto es deixa com a abstracte, ja que cada botó específic ha de fer una funcionalitat particular. Aquest botó genèric segueix el principi obert tancat (obert per estendre el funcionament del botó genèric en herències però tancat a la modificació) i el principi DRY (don't repeat yourself).

5.3.3. ItemGeneric

Aquesta classe hereta de MenuItem i és abstracte. Totes les vistes de la pràctica tenen una barra de menú, la qual es compon de submenús i ítems. Seguint la mateixa filosofia que el botó genèric, vam crear el ítem genèric amb el mètode abstracte actionPerformedItem.

6. Classes de la capa de persistència

Per gestionar l'emmagatzematge i recuperació de dades en el nostre sistema, hem separat les dades dels productes i les solucions. Aquesta separació respon a la necessitat de mantenir la coherència amb el funcionament del sistema, ja que les prestatgeries poden contenir productes que no formen part del catàleg actual. Això pot succeir perquè aquests productes pertanyen a catàlegs antics.

Quan l'usuari vol guardar o importar dades, el sistema requereix que es proporcioni un *path* i un nom de fitxer. En aquests casos, el sistema ha de garantir que:

- El *path* proporcionat sigui vàlid.
- Es pugui accedir al directori i escriure-hi.

- El fitxer es pugui crear, editar o llegir segons sigui necessari.

Per evitar duplicar codi i millorar la reutilització, hem creat una classe abstracta anomenada “CtrlPersistenciaGeneric”. Aquesta classe s'encarrega de realitzar totes les comprovacions relacionades amb els directoris i fitxers. Les classes “CtrlPersistenciaSolucio” i “CtrlPersistenciaCataleg” hereten d'aquesta classe abstracta i es beneficien de la seva funcionalitat comuna.

Els punts clau d'aquest disseny són:

- **Estructura Singleton:** Els controladors de persistència són singletons i la seva creació està centralitzada en la classe “CtrlPersistencia”, que assegura que només hi hagi una instància de cada controlador.
- **Validació de format:** Els controladors “CtrlPersistenciaSolucio” i “CtrlPersistenciaCataleg” són responsables de desxifrar el contingut dels arxius i comprovar que les dades contingudes estiguin en el format correcte. Això garanteix que el sistema pugui treballar amb dades consistents i fiables.
- **Gestió d'errors:** Si l'usuari intenta importar dades al començament del programa i l'arxiu proporcionat no està en el format esperat, el sistema llança una excepció per informar l'usuari i interrompre l'operació. Això evita errors en el funcionament del sistema.

Aquest enfocament assegura un tractament robust i reutilitzable de les operacions d'importació i exportació, mantenint la coherència de les dades.

6.1. Controladors

6.1.1 CtrlPersistencia

La classe CtrlPersistencia és un controlador central del sistema encarregat de gestionar la persistència de dades. Aquesta classe implementa el patró singleton, assegurant que només hi hagi una instància única d'ella mateixa i dels controladors de persistència que gestiona. Actua com un punt d'accés per a la capa de persistència, coordinant la interacció amb controladors especialitzats.
Si es requereix accedir a les dades de les solucions, delega aquesta responsabilitat a CtrlPersistenciaSolucio. Si cal gestionar informació del catàleg de productes, recorre a CtrlPersistenciaCataleg.

6.1.2 CtrlPersistenciaGeneric

És una classe abstracta dissenyada per proporcionar funcionalitats genèriques per gestionar la persistència de dades al sistema. Durant el procés de disseny de la capa de persistència, vam identificar que hi havia moltes funcionalitats comunes en la gestió de la informació tant dels productes com de les solucions. En ambdós casos, cal comprovar que el *path* sigui vàlid i que l’arxiu indicat per l’usuari es pugui llegir i escriure de manera segura, per exemple.

Per tal d’evitar duplicació de codi i promoure la reutilització, hem decidit crear una classe abstracta que encapsuli aquestes funcionalitats comunes. D’aquesta manera, els controladors específics poden heretar d’aquesta classe i beneficiar-se de les funcionalitats genèriques, mantenint un disseny modular i eficient.

6.1.3 CtrlPersistenciaSolucio

test1.txt

```
true
nomSolucio1
cereals oreos llet
filipinos

false
nomSolucio2
carn peix suc verdura salsa

false
nomSolucio3
```

De cada solució guardem la següent informació: si la solució ha estat modificada, el nom de la solució i els productes que conté. Els productes es guarden amb el seu ordre corresponent, distribuint-los en diferents línies per indicar que es troben en prestatgeries diferents.

Figura 8: exemple d'informació d'una solució guardada

Si l'arxiu que s'intenta llegir no té les dades en l'ordre correcte o el contingut viola alguna restricció del programa —per exemple, si s'intenta afegir dues solucions amb el mateix nom—, el programa llança excepcions per tal d'avortar l'acció i garantir la integritat de les dades.

La classe CtrlPersistenciaSolució te els mètodes necessaris per descodificar tota aquesta informació i processar-la abans d'enviar-la a la capa de domini.

6.1.4 CtrlPersistenciaCataleg

Aquest subcontrolador s'encarrega de codificar i descodificar les dades en el fitxer corresponent, concretament aquelles relacionades amb el catàleg: és a dir, els productes que existeixen, les similituds entre ells (via matriu) i les restriccions que hi ha entre aquests. Aquestes dades són emprades pel controlador de domini per actualitzar el catàleg i importar configuracions de forma més eficient.

El controlador ofereix un seguit d'excepcions indicant l'error concret i la línia on es produeix per poder tractar error als arxius. El format és: un productes per línia; línia en blanc per delimitar; matriu de similituds on l'ordre dels productes a la llista anterior correspon amb els índexs de la matriu (valors de 0 a 100); línia en blanc per delimitar; matriu de restriccions similar a la de similituds pero amb 0s i 1s, on 1 representa que els productes amb els índexs de la posició tenen una restricció de consecutivitat.

7. Relació de classes implementades per cada membre de l'equip

Algorisme	Efrain Tito	CtrlSolucions	Eulàlia Peiret
AlgorismeBT	Efrain Tito	MergeFindSet	Lluc Santamaria
AlgorismeGreedy	Efrain Tito	Pair	Alejandro Lorenzo
Aproximacio	Lluc Santamaria	Producte	Alejandro Lorenzo
CtrlCataleg	Alejandro Lorenzo	Solucio	Eulàlia Peiret
CtrlDomini	Lluc Santamaria	SolucioModificada	Eulàlia Peiret
Driver	Lluc Santamaria	CtrlCatalegAmbRestr	Efrain Tito
CtrlVistaSolucions	Eulàlia Peiret	VistaControladors	Eulàlia Peiret
VistaPrincipalSolucions	Eulàlia Peiret	VistaGestioAlgorisme	Eulàlia Peiret
CtrlPersistencia	Eulàlia Peiret	CtrlPersistenciaGeneric	Eulàlia Peiret
CtrlPersistenciaSolucio	Eulàlia Peiret	CtrlPersistenciaCataleg	Efrain Tito
VistaPrincipalCataleg	Alejandro Lorenzo	VistaInfoProducte	Alejandro Lorenzo
VistaAfegirProducte	Alejandro Lorenzo	VistaConsultarRest	Efrain Tito
CtrlVistaCatalegAmbRestriccions	Efrain Tito	VistaGenerica	Lluc Santamaria
BotoGeneric	Lluc Santamaria	VistaInfoSolucio	Lluc Santamaria
MenuItemGeneric	Lluc Santamaria	VistaPrincipal	Lluc Santamaria
Main	Lluc Santamaria	CtrlVistaGeneric	Lluc Santamaria
CtrlPresentacio	Lluc Santamaria	CtrlGeneric	Lluc Santamaria