Full de càlcul senzill

Identificador de l'equip: 1.1

Joan Aluja Oraá: joan.aluja@estudiantat.upc.edu
Marc Clapés Marana: marc.clapes.marana@estudiantat.upc.edu
Marc Duch Buechler: marc.duch@estudiantat.upc.edu
Andreu Orensanz Bargalló: andreu.orensanz@estudiantat.upc.edu

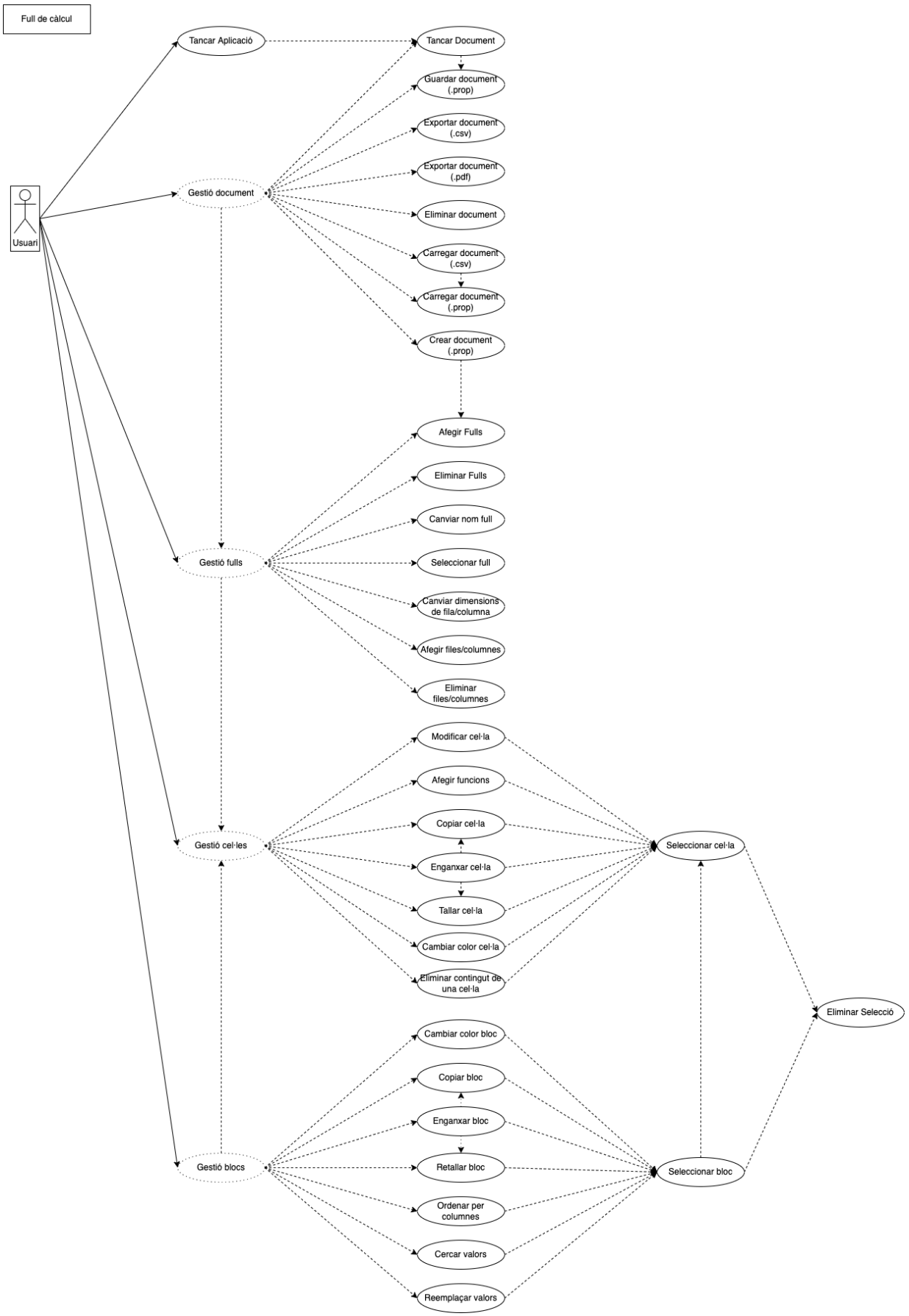
Versió del lliurament: 1.1

1ª entrega PROP

Índex

1. Diagrama de casos d'ús

A continuació mostrem el diagrama de casos d'ús. El diagrama es troba detallat al directori DOCS.



1.1 Descripció Casos d'ús

Nom: Tancar Aplicació

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari farà clic al botó de tancar l'aplicació
2. El sistema seguirà el procediment de **Tancar Document**
3. El sistema termina l'aplicació

Errors possibles i cursos alternatius:

1a. No hi ha un document obert, el sistema termina l'aplicació (salta al pas 3).

Nom: Tancar document

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica que vol tancar el document obert.
2. Si el document obert té canvis que no s'han guardat, el sistema mostrarà a l'usuari una finestra amb 3 opcions
    a. **Guardar el Document**. Segueix al punt 4
    b. **Tancar Document** sense guardar-lo. Segueix al punt 3
    c. Cancel·lar (surt del cas d'ús)
3. Si s'ha seleccionat la opció b), notificarà a l'usuari que està a punt de tancar el document sense guardar-lo amb dues opcions.
    a. Sí, segueix al punt 4.
    b. No, torna al punt 2.
4. El sistema tanca el document obert i torna a "l'estat inicial" de la applicació

Errors possibles i cursos alternatius:

1a. No hi ha un document obert (no fa res/no surt la opció)

Nom: Guardar document

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica que vol guardar el document actual
    a. Ex: Ctrl + S
2. Si el document no tenia nom, haurà d'assignar-li un. Per defecte es guardarà com un document .prop
3. El sistema s'encarregarà de guardar les dades necessàries en el nou format .prop.

Errors possibles i cursos alternatius:

1a. El document amb nom "x" ja existeix. (Sortirà un avís indicant que s'ha d'utilitzar un altre nom).

1b. El document es guarda amb una extensió .csv o .pdf -> **Exportar Document**

1c. El document es guarda amb una extensió que no coneix. El sistema informarà al usuari que el format que ha escollit no el reconeix.

1d. No hi ha un document obert (no fa res/no surt la opció)

Nom: Exportar Document

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indicarà al sistema que vol exportar el document obert
2. L'usuari escollirà el format amb el que vol exportar (.csv, .pdf, .prop). Si el document no tenia un nom, li haurà d'assignar-li un.
3. El sistema s'encarregarà de guardar les dades necessàries amb el format especificat en un nou document (document diferent al que s'està editant).

Errors possibles i cursos alternatius:

Format .csv:

i. Les funcions es guardarà com strings
ii. Les referències a cel·les es mantindran (strings)
iii. Només es podrà exportar un sol full (el "seleccionat")
iv. No es guardarà el color de una cel·la
v. No es guardarà el tamany de les files/columnes
vi. Guardarà el subconjunt rectangular de cel·les més petit possible (amb una cantonada fixa a dalt a la esquerra)

Format .pdf:

vii. Les funcions es guardarà el resultat
viii. Les referències es traduiran al valor de la cel·la referida (recursiu)
ix. Només es podrà exportar un sol full (el "seleccionat")
x. No es guardarà el color de una cel·la
xi. El tamany de les files/columnes queda reflectit en el resultat final
xii. Guardarà el subconjunt rectangular de cel·les més petit possible (amb una cantonada fixa a dalt a la esquerra)

Nom: Eliminar document

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica que vol eliminar un document
2. El sistema s'encarregarà d'obrir un "explorador d'arxius" amb els documents en un PATH per defecte (ex: la localització del .jar)
3. Escollirà el document (.prop o .csv) que vulgui eliminar.
4. El sistema avisarà a l'usuari si vol seguir amb la operació (eliminar es permanent)
    a. Sí, segueix al pas 5
    b. No, torna al pas 3
5. El sistema elimina el fitxer en questió

Errors possibles i cursos alternatius:

En qualsevol moment l'usuari pot avortar l'operació

5a. El document seleccionat està en ús (per la nostra app / altres?).
El sistema informa a l'usuari de que no es pot elimiar el document en qüestió perquè està en ús. Torna al pas 3.

Nom: Carregar document

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indicarà al sistema que vol carregar un document
2. El sistema obrirà un "explorador d'arxius" amb els documents en un PATH per defecte (ex: la localització del .jar)

3. L'usuari escollirà el document que vulgui carregar.
    a. El document haurà de ser de tipus .prop o .csv
4. L'usuari confirma la selecció
5. El sistema s'encarregarà d'obrir el document en qüestió.

Errors possibles i cursos alternatius:

3a. El document en qüestió és un .csv, el sistema **Crea un Document** amb un full ("Full 1") amb les dimensions mínimes per incloure els elements del .csv, qualsevol valor que depengui del document (ex: colors de cel·les/tamany visual de files/columnes) serà el per defecte

3b. El document en qüestió es un .prop (extensió propia), carregarà el document segons les especificacions de la extensió

xiii. (tamanys files/cols // colors de cel·les // etc...)

4c. Ja hi ha un document obert. El sistema avisarà a l'usuari que té un document obert i li preguntarà si:

xiv. Vol **Tancar el Document** actual

xv. Cancel·lar l'operació de càrrega

Nom: Crear document

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica al sistema que vol crear un document
2. El sistema mostra una finestra amb un camp amb el nom del document i les dimensions màximes que podran tenir els fulls.
3. L'usuari pot assignar-li un nom (opcional) al document creat i canviar les dimensions.
4. L'usuari indica al sistema que vol crear el document amb el nom assignat (o un per defecte)
5. El sistema crea un document amb un full amb les dimensions especificades

Errors possibles i cursos alternatius:

En qualsevol moment, l'usuari pot cancel·lar l'operació.

2a. Les dimensions han de ser majors a 0 ($m/n > 0$)

Nom: Afegir fulls

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indicarà al sistema que vol crear un full.
    a. Ex: Clicant l'icona "+".
2. L'usuari indicarà el nombre de files i columnes (n, m) del nou full.
    a. n & m han de ser inferiors al limit establert al crear un document
3. El sistema assigna el nom del full a "Full n" on n és el número de full creat, sent 1 el primer full que es crea quan es crea un document.
4. El sistema crea un nou full amb les mateixes files i columnes que les que ha definit l'usuari al pas 2

Errors possibles i cursos alternatius:

2b. Les dimensions del full són incorrectes

xvi. $m/n = 0$

xvii. $m/n >$ Limit del document

xviii. Pels punts anteriors avisaríem a l'usuari de l'error i se li mostraria els límits establerts.

L'usuari podrà avortar en qualsevol punt

Nom: Eliminar fulls

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica al sistema que vol eliminar un full en particular
    a. Ex: Clicant la 'x' de la pestanya del full.
2. El sistema informa a l'usuari si esta segur:
    a. Si, continua al pas 3
    b. No, avorta el cas d'ús
3. El sistema elimina el full i les seves cel·les

Errors possibles i cursos alternatius:

1a. Només hi ha un full i s'intenta eliminar. Apareix un text d'error dient que un document ha de contenir almenys un full (l'opció no surt?)

2b. Si hi ha una cel·la d'un altre full que fa referència a una cel·la del full que volem eliminar, el sistema ens avisa sobre aquestes referències i l'usuari haurà de confirmar si les vol fer.

Nom: Canviar nom full

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica al sistema que vol canviar el nom d'un full
    a. Ex: Fa doble clic sobre el nom del full per tal de canviar-lo.
2. El sistema mostra una finestra amb un camp de text
3. L'usuari escriu el nom que vol que tingui el full.
4. El sistema canvia el nom del full i actualitza qualsevol referència que hi havia
    a. Si hi havia una cel·la d'un altre full que feia referència al full que s'ha renombrat, aquesta referencia s'actualitzarà.

Errors possibles i cursos alternatius:

L'usuari pot avortar en qualsevol moment

3a. Ja hi ha un full amb el mateix nom introduït.

Nom: Seleccionar full

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari selecciona un full
    a. Ex: Fa clic sobre el nom del full
2. El sistema mostra el nou full

Errors possibles i cursos alternatius:

-

Nom: Canviar dimensions de fila/columna

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari selecciona la vora d'una fila/columna
2. L'usuari arrossega la vora a les dimensions desitjades
3. El sistema guarda els canvis fets

Errors possibles i cursos alternatius:

2a. El sistema no deixa fer més petit o més grans les dimensions d'una cel·la si sobrepassa un límit que es determinarà a la capa de presentació.

Nom: Afegir files/columnes

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica al sistema que vol afegir una fila o afegir una columna
    a. Per tal de crear una altra fila sota l'última o una columna a la dreta de l'última, respectivament.
2. El sistema afegeix una fila/columna
    a. Les files afegides tindran el número de l'última més 1.
    b. Les columnes aniran assignades per una lletra de l'abecedari, per tant serà la següent de l'última columna del full.

Errors possibles i cursos alternatius:

1a. Es vol afegir una fila o columna quan ja està al màxim predefinit a la creació del document

2b. Quan la última fila sigui la última lletra de l'abecedari, la 'Z', a la següent columna creada se li assignarà les lletres 'AA' i així continuant la series. (Això només es veurà a la interfície, internament les columnes també van numerades com les files)

Nom: Eliminar files/columnes

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **Selecciona una cel·la**
2. L'usuari indica que vol eliminar una fila o una columna
3. El sistema avisa l'usuari amb una finestra si/no
4. El sistema elimina la fila/columna en la que es troba la *cel·la seleccionada*
    a. El sistema s'encarregarà de mantenir la numeració de les files i les columnes si la fila/columna eliminada no es la última.

Errors possibles i cursos alternatius:

1a. Es vol eliminar una fila o columna però només n'hi ha una. Surt un error dient que no es pot eliminar totes les columnes o files d'un full.

Nom: Seleccionar cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari fa click sobre una cel·la
2. La cel·la clickada queda seleccionada
3. S'**elimina selecció** anterior

Errors possibles i cursos alternatius:

-

Nom: Modificar cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **Selecciona una cel·la (Cas d'us Seleccionar cel·la)**
2. L'usuari entra per teclat el valor que vol entrar (Pot ser un string, int, double, data o una funció).
3. El sistema guarda el valor a la cel·la i/o el calcula si es una funció.
    a. El sistema borra el contingut que hi havia abans (si n'hi havia)

Errors possibles i cursos alternatius:

2a. L'usuari selecciona el camp de text de la part superior de l'aplicació
    xix. Edita directament el text que hi ha a la cel·la

2b. La funció entrada no es reconeguda (Es mostra un error)

3c. Si el valor entrat es una funció es fa el càlcul corresponent (Si algun element de la funció fa referència a un altre cel·la el sistema va a buscar el valor [recursivament si escau]. Opcionalment, si el valor està a un altre full el sistema anirà a buscar-lo [recursivament si escau]).

3b. La funció no es pot aplicar sobre algun dels tipus dels elements (Ex. Sumar una data a un string).

Nom: Afegir funcions

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **Selecciona una cel·la (Cas d'us Seleccionar cel·la)**
2. L'usuari indica que vol insertar una funció
    a. Fa click dret i selecciona el menú de funcions.
3. L'usuari selecciona una funció
4. El sistema obre una finestra
5. L'usuari omple el contingut dels camps depenent de la funció seleccionada.
    a. Si requereix dos elements s'insereixen les coordenades dels valors o els propis valors. Si es requereix un bloc, s'insereixen les coordenades dels extrems dels blocs.
6. El sistema calcula el valor de la funció (Si algun element de la funció fa referència a un altre cel·la el sistema va a buscar el valor [recursivament si escau]).

Errors possibles i cursos alternatius:

1a. No hi ha cap cel·la seleccionada

2a. La funció entrada no es reconeguda pel sistema.

2b. Si algun element de la funció fa referència a un altre cel·la el sistema va a buscar el valor [recursivament si escau]. Opcionalment, si el valor està a un altre full el sistema anirà a buscar-lo [recursivament si escau]

2c. El tipus de valor no era l'esperat per la funció (Ex. Sumar una data a un string).

Nom: Copiar cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la (Cas d'us Seleccionar cel·la)**
2. L'usuari indica que vol copiar el contingut d'una cel·la
    a. Ctrl + C
3. El sistema copia el contingut (Valor/Funció/Referència a cel·la) de la cel·la seleccionada (amb CTRL + C). Aquest valor es guarda temporalment a memòria per poder ser enganxat varies vegades. El valor que te la cel·la de la qual copiem es manté.

Errors possibles i cursos alternatius:

3a. La cel·la de la cual copiem no té cap contingut. Es guarda a memòria aquesta informació (String buit).

Nom: Tallar cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la (Cas d'us Seleccionar cel·la)**
2. L'usuari indica que vol retallar el contingut de la cel·la
    a. Ctrl + X
3. El sistema copia el contingut (Valor/Funció/Referència a cel·la) de la cel·la seleccionada (amb CTRL + X). Aquest valor es guarda temporalment a memòria (*clipboard*) per poder ser enganxat varies vegades.
4. El sistema borra el valor de la cel·la seleccionada.

Errors possibles i cursos alternatius:

3a. La cel·la de la cual tallem no té cap contingut. Es guarda a memòria aquesta informació (String buit).

Nom: Enganxar cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la (Cas d'us Seleccionar cel·la)**
2. L'usuari indica que vol enganxar una cel·la
    a. Ctrl + V
3. El sistema enganxa el contingut (Valor/Funció/Referència a cel·la) guardat via copia (CTRL + C) o tallat (CTRL + X) d'una cel·la.

Errors possibles i cursos alternatius:

3a. No hi ha cap cel·la copiada (CTRL + C) o tallada (CTRL + X) i per tant es cancel·la l'operació i no s'enganxa cap contingut en la cel·la seleccionada.
3b. El sistema informa a l'usuari que la cel·la seleccionada, és d'un altre full del document (full diferent del cual copiem/tallem), i diu de quin full es.
3c. Les referències s'actualitzen per fer referència a les cel·les de l'altre full
    xx. Si una referencia ja era d'un altre full, aquesta es manté igual
3d. El contingut que s'ha d'enganxar és buit (String buit). Per tant es borra el contingut de la cel·la a la qual s'enganxa.

Nom: Canviar color cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la (o un bloc de cel·les)**
2. L'usuari selecciona el color que li vol donar a aquella cel·la utilitzant un menú desplegable (Selecció de colors predefinits).
3. El sistema canvia el color de la cel·la seleccionada al color escollit per l'usuari. (El color només és guarda en capa de persistència si s'utilitza el format .prop)

Errors possibles i cursos alternatius:

Nom: Eliminar contingut d'una cel·la

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la (Cas d'us Seleccionar cel·la)**
2. L'usuari indica que vol eliminar el contingut de la cel·la
    a. SUPR
3. El sistema borra el contingut que té la cel·la seleccionada.

Errors possibles i cursos alternatius:

3a. No hi ha cap cel·la seleccionada i per tant no es realitza l'operació d'esborrat.

Nom: Eliminar selecció

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

4. L'usuari indica que vol eliminar la selecció actual
    a. ESC
5. El sistema es desfà de la selecció
    a. Deixa de tenir un *bloc seleccionat*
    b. Deixa de tenir una *cel·la seleccionada*

Errors possibles i cursos alternatius:

No hi ha cap cel·la/bloc seleccionat. No fa res

Nom: Seleccionar Bloc

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la** (representarà la primera cantonada de un rectangle)
    a. (Click esq)
2. L'usuari **selecciona una segona cel·la** (pot ser la mateixa)
    a. (SHIFT + Click esq)
3. S'**Elimina selecció** anterior
4. Les cel·les que es troben entre les columnes / files de les dues cel·les seleccionades passen a formar part del "*bloc seleccionat*"
    a. Es forma una àrea rectangular entre les dues cel·les seleccionades

Errors possibles i cursos alternatius:

1a. L'usuari només selecciona una cel·la. El procés de "seleccionar un bloc" avorta. El cas d'ús passa ser el de **Seleccionar una cel·la**

2a. La segona cel·la és la mateixa que la primera -> Bloc seleccionat és una sola cel·la

Nom: Copiar Bloc

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona un Bloc**
2. L'usuari indica al sistema que vol copiar un bloc
    a. Ctrl + C
3. El sistema es guarda una copia del bloc en memòria (*clipboard*)

Errors possibles i cursos alternatius:

1a. No hi ha un bloc seleccionat. No fa res

Nom: Enganxar Bloc

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona una cel·la**
2. L'usuari indica que vol enganxar un bloc que té guardat en memoria (*clipboard*)
    a. Ctrl + V
3. El sistema enganxa d'esquerra a dreta i de dalt a baix el bloc prèviament copiat o retallat.
    a. El sistema copia el contingut (Valors/Funcions/Referències a cel·les) de cada cel·la del *bloc seleccionat (clipboard)*, i l'enganxa d'esquerra a dreta i de dalt a baix de el bloc seleccionat.

Errors possibles i cursos alternatius:

3a. La cel·la es de un full different.

4a. El sistema informa a l'usuari que el contingut del *bloc seleccionat*, és d'un altre full del document (full diferent al que s'enganxa), i diu de quin full és.

5a. Les referències s'actualitzen per fer referència a les cel·les de l'altre full

xxi. Si una referència ja era d'un altre full, aquesta es manté igual

4b. Una de les cel·les a la que es vol enganxar ja té un valor (no només la seleccionada). El sistema envia un missatge si/no avisant a l'usuari que si decideix seguir amb la operació d'enganxar el bloc, es carregarà les dades.

Nom: Retallar Bloc

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **selecciona un Bloc**
2. L'usuari indica que vol retallar el bloc
    a. Ctrl + X
3. El sistema es guarda una copia del bloc en memoria (*clipboard*)
4. El sistema borra el contingut del *bloc seleccionat*

Errors possibles i cursos alternatius:

Nom: Ordenar per columnes

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari **Selecciona un Bloc**

2. L'usuari indica al sistema que vol ordenar un bloc
    a. Click botó ordenar
3. Els elements inclosos en el *bloc seleccionat*, s'ordenen de forma ascendent i per columnes.
4. L'ordenació farà servir el següent criteri:
    a. Números (ints/floats...) (ascendent)
    b. Dates (asc)
    c. Text (strings) (alfabèticament a->z)
    d. En el cas de les funcions, farà servir el resultat de la funció

Errors possibles i cursos alternatius:

1a. L'usuari no ha seleccionat un bloc. El sistema avisa l'usuari de que no ha seleccionat un bloc

3a. L'usuari torna a apretar el botó de ordenar. Els passos segueixen igual, pero en ordre descendent. El criteri també s'inverteix (text - dates - nums)

Nom: Cercar Valors

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica que vol cercar un valor
    a. CTRL + F
2. El sistema mostra una finestra per introduir el valor que es busca
3. L'usuari escriu el valor que vol buscar
4. L'usuari prem el botó "buscar"
5. El sistema resalta totes les cel·les que tenen caràcters coincidents (Ex: busquem **10**, ens resaltarà tots els **10**0/**10**00/**10**000...)
    a. En el cas de les funcions, només tindrà en compte el resultat
6. L'usuari indica al sistema si vol anar al següent resultat o a l'anterior
    a. pot anar pitjant les fletxes de dreta/esquerra per anar resaltant les cel·les coincidents
7. En ordre d'esquerra a dreta i de dalt a baix

Errors possibles i cursos alternatius:

1a. L'usuari té un *bloc seleccionat*. La cerca es fa sobre el conjunt de cel·les del *bloc seleccionat*

5b. L'usuari no ha introduït un valor a buscar/comparar. El sistema avisa a l'usuari que no ha introduït un valor a buscar

Nom: Reemplaçar Valors

Actor: Usuari

Comportament (diàleg entre els actors i el sistema):

1. L'usuari indica que vol reemplaçar valors
    a. CTRL+ R
2. El sistema mostra una finestra amb un camp per introduir el valor que es vol buscar, una capsa per marcar si es vol reemplaçar tota la cel·la o només les parts coincidents, i finalment un camp amb el nou valor que es vol reemplaçar
3. L'usuari escriu en els camps els valors que vol substituir i si vol, o no, substituir la cel·la sencera
4. L'usuari prem el botó "buscar i reemplaçar"

5. El sistema resalta totes les cel·les que tenen caràcters coincidents
    a. El sistema no ressaltarà cel·les amb funcions
6. El sistema ressaltarà la primera cel·la i enviarà un missatge de confirmació al usuari sobre si vol o no reemplaçar aquella cel·la en concret
7. L'usuari respon
8. El sistema procedeix segons la resposta
    a. Si: La cel·la es reemplaça segons l'opció seleccionada, passa a la següent cel·la
    b. No: La cel·la es manté igual, passa a la següent cel·la

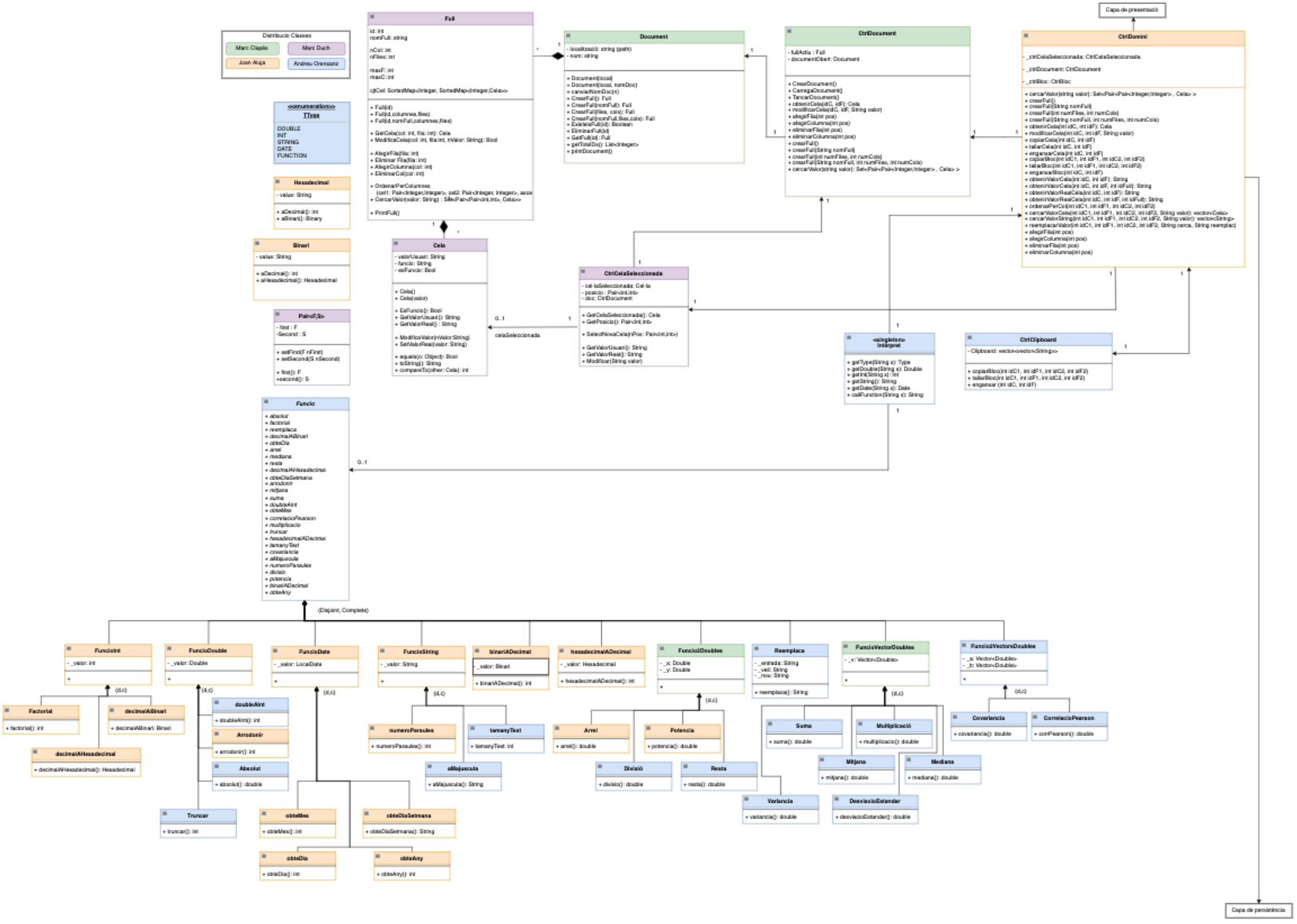
Errors possibles i cursos alternatius:

1a. L'usuari no ha seleccionat un bloc, es pren com a *bloc seleccionat* tot el full
5b. L'usuari no ha omplert un dels camps (text a buscar/ text amb el que reemplaçarà). El sistema l'informa del error i espera a que l'usuari torni a donar-li al botó (pas 4)
6c. No hi ha cel·les coincidents. El sistema mostra un missatge explicant que no s'han trobat cel·les coincidents a la selecció
7-9d. L'usuari prem una de les fletxes horitzontals, el sistema interpreta que no es vol fer el canvi i torna al pas 7 amb la cel·la anterior/posterior depenent de la fletxa premuda

2. Diagrama del model conceptual

2.1. Disseny del diagrama de model conceptual

A continuació mostrem el diagrama del model conceptual. El diagrama es pot trobar detallat a la carpeta DOCS.



Restriccions textuals:

- Claus primàries:
  (**Document**, nom+localització)
  (**Full**, id)
  (**Cela**, fila+columna+idFull)

2.2. Descripció de les classes

2.2.1. Document

La classe Document és la classe que conté tots els elements que formen el full de càlcul, és a dir, fulls i aquests contenen cel·les. S'encarrega principalment de la gestió dels fulls, així com crea-los, eliminar-los i modificar-los.

2.2.2. Full

La classe Full representa una sola fulla de càlcul. Aquesta classe és l'encarregada de gestionar les seves cel·les (les que pertanyen al full en si). S'encarrega de modificar les cel·les existents o de afegir-les si no estaven ja al full, ordenar-les, o cercar les cel·les per valors.

2.2.3. Cel·la (Cela)

La classe Cel·la (Cela.java) és la classe més granular del sistema. La seva funció és guardar el valor que introdueix l'usuari, sigui un funció o un altre tipus, i retornar, en el cas de les funcions, el seu resultat.

2.2.4. Interpret

La classe Interpret fa d'analitzador sintàctic de l'input d'una cel·la, en determina el seu tipus i transforma els tipus String a Double, Integer, LocalDate o Funció (tipus definit al enum TType). Si l'input d'una cel·la és una referència al contingut d'una altra, l'Interpret s'encarrega d'obtenir aquest valor. També s'encarrega de gestionar totes les funcions i retornar-ne el resultat.

2.2.5. CtrlDomini

El controlador de Domini s'encarrega d'instanciar la resta de controladors de la capa de domini i proporciona una interfície més pròxima a l'usuari dels mètodes de la resta de controladors com el CtrlDocument o el CtrlClipboard.
També s'encarrega de comunicar-se amb l'intèrpret per realitzar els càlculs de funcions quan es modifica una cel·la.

2.2.6. CtrlDocument

El controlador de document s'encarregarà de la gestió dels mètodes de les classes de Document i Full. Es comunica amb Document per tal de realitzar totes les funcionalitats referent a aquest.

2.2.7. CtrlCelaSeleccionada

La classe CtrlCelaSeleccionada representa la cel·la que l'usuari seleccionarà a la capa de presentació. Representa el punt d'entrada d'un usuari amb una cel·la en particular.

2.2.8. CtrlClipboard

El controlador de clipboard s'encarrega de gestionar els mètodes copiar, tallar i enganxar cel·les i blocs al full corresponent. Les cel·les copiades queden dins una matriu d'ArrayList de Cela que fa la funció de porta-retalls (*clipboard*).

2.2.9. Funció

La classe funció és una classe abstracta que conté tots els mètodes necessaris per realitzar totes les operacions del sistema. Aquests mètodes estan implementats a les subclasses que van classificades segons els paràmetres que necessitin les funcions.

2.2.10. Binari

La classe Binari gestiona la conversió de tipus entre diferents bases a binari i serveix com a tipus.

2.2.11. Hexadecimal

La classe Hexadecimal gestiona la conversió de tipus entre diferents bases a hexadecimal i serveix com a tipus.

2.2.12. Pair<F, S>

La classe Pair<F, S> és una classe genèrica per crear tuples de dos elements, utilitzada principalment per simplificar l'ús dels index columna / fila o retornar un Pair(columna, fila) amb la seva cel·la.

3. Relació de les classes implementades per membre de l'equip

A continuació mostrem una taula amb les classes que ha implementat cada membre del grup. De cada classe feta per un membre, aquest mateix s'ha encarregat del seu testeig i la seva documentació i, en el cas dels controladors, de la implementació del seu driver corresponent.

Joan Aluja	Marc Clapés	Marc Duch	Andreu Orensanz
BinariADecimal	Document	Full	Interpret
DecimalABinari	Funcio2Doubles	Cela	doubleAInt
HexadecimalADecimal	FuncioVectorDoubles	CtrlCelaSeleccionada	AMajuscula
DecimalAHexadecimal	CtrlDocument	Pair	TamanyText
Factorial			Truncar
Arrel			Absolut
Potencia			Reemplaca
ObteAny			Suma
ObteMes			Multiplicació
ObteDia			Resta
ObteDiaSetmana			Divisió
NumeroParaules			Variancia
Arrodonir			DesviacioEstandar
FuncioInt			Mitjana
FuncioDouble			Mediana
FuncioString			CorrelacioPearson
FuncioDate			Covariancia
CtrlDomini			Funcio
Binari			Funcio2VectorsDoubles
Hexadecimal			CtrlClipboard
			TType

Controladors → Verd
Tipus → Blau

4. Estructures de dades i algorismes utilitzats

4.1. Estructures de dades

A continuació expliquem cada una de les estructures de dades que hem considerat per tal de crear el nostre full de càlcul i quina d'elles hem acabat fent servir.

4.1.2 Full

Tal i com hem especificat anteriorment, la classe full representa una agrupació de cel·les que, tal i com apareix al UML, pot ser buit, que simplement vol dir que només guardarem aquelles cel·les que tinguin un valor no-nul.

Abans d'entrar en detall sobre les estructures plantejades, definirem 2 variables, el número de cel·les que no estan buides seran representades per la x, i la segona variable n, representa el nombre de files, i per tal de simplificar alguns dels costos, el nombre de columnes.

D'aquesta manera, podem dir que el màxim nombre de cel·les (x) serà el nombre de files per el nombre de columnes. Amb això podem definir els límits de $x \mid 0 \leq x \leq n^2$.

A la hora de decidir com implementar l'agrupació de cel·les d'un full, vam haver de decidir-nos entre varies opcions:

- **Matriu bidimensional:**

  Per representar una taula d'elements com un full de càlcul, hi ha una estructura que és molt fàcil d'entendre i d'implementar, la matriu. Implementada mitjançant dos col·leccions d'objectes aniuats, tals que la primera col·lecció ("exterior") tindrà la segona col·lecció, i és aquesta la que guardarà els elements finals de la matriu.

  Depenent de com vulguem implementar les funcionalitats de Full, podem interpretar la col·lecció externa com les columnes o com les files sense cap problema.

  - **Matriu d'Arrays**

    L'estructura més evident per implementar una matriu és un Array d'Arrays, la base de les col·leccions d'elements a Java.

    La ventatja principal d'aquesta implementació es que a l'hora d'accedir a un element segons la seva fila i columna, el temps d'accés és instantani, $\Theta(1)$ independentment del nombre d'elements o el tamany.

    En quant als contres, en tenim uns quants:

    Començant per l'espai ocupat en memòria; independentment del nombre de cel·les (x), el espai ocupat en memòria sempre serà $\Theta(n^2)$. Tal i com es guarden els objectes a Java (en aquest cas Cel·la), aquest cost no es tan gran com en altres llenguatges, per tot aixi, el cost segueix sent quadratic i

a la pràctica, el nombre de cel·les que s'omplen en un full molt gran serà molt petit en comparació.
L'altre gran problema apareix a l'hora d'afegir o eliminar files/columnes. Els Arrays tenen un tamany fix, cosa que vol dir que a la hora de fer més gran o més petit un full, hauriem de clonar totes les cel·les a un altre matriu nova, que faria que el cost d'aquestes operacions seria sempre $\Omega(\boldsymbol{n^2})$, sense plantejar-nos la complexitat de moure tots els elements segons on s'ha afegit la fila/columna.
Una solució que ens vam plantejar per el problema anterior era inicialitzar la matriu amb un tamany superior al especificat per l'usuari/donat per defecte, pero això simplement empitjora el problema de l'espai ocupat.

- **Matriu de LinkedList**

  Per tal de millorar el problema d'afegir/eliminar files o columnes, vam pensar en implementar la matriu amb LinkedList o DoubleLinkedList.

  Amb aquesta implementació, milloràvem el problema d'afegir/eliminar files o columnes, pero només per un dels dos eixos.
  Si la primera LinkedList representa les columnes, afegir-les/eliminar-les era tan fàcil com buscar el node corresponent i afegir una nova LinkedList i actualitzar els punters o eliminar el node (amb la LinkedList anterior), cost temporal lineal $\Theta(\boldsymbol{n})$. Ara bé, si volem afegir o eliminar una fila (on els índexs de les files estan a la segona LinkedList), el cost augmenta dràsticament, ja que per cada columna, hauriem de trobar la fila corresponent i modificar els nodes segons si s'afegeix o s'elimina la fila. El millor dels casos, eliminem només la primera fila i el cost es $\Omega(\boldsymbol{n})$, i el pitjor, eliminem l'ultima, $O(\boldsymbol{n^2})$, la mitja, però, segueix sent $\Theta(\boldsymbol{n^2})$. Millor que amb Arrays estàtiques, pero no per molt.
  La gran desavantatge que té en comparació a la matriu d'Arrays és el seu cost d'accés. On l'anterior tenia un cost fix $\Theta(\mathbf{1})$, la implementació amb LinkedList trigaría en el pitjor dels casos $O(\boldsymbol{n} + \boldsymbol{n})$, i de mitja $\Theta(\boldsymbol{n})$. No es horrible, pero tenint en compte que l'accés a les cel·les és un element fonamental d'un full de càlcul, havíem de buscar una estructura amb un cost d'accés millor que lineal.

Diccionaris

Les matrius de col·leccions, estàtiques o no, tenen el mateix problema: el seu cost espaial sembre sera $\Theta(\boldsymbol{n^2})$. Tot i així, l'accés instantani que permet la matriu d'Arrays es practicament impossible de millorar.

Aquí és on entren els diccionaris, amb aquestes estructures de dades ens desfem del concepte de files i columnes, ja que les claus que fan servir no estan restringides a cap tamany.
A més, l'espai que ocupen sempre serà $\Theta(\boldsymbol{x})$ si ens desfem de les cel·les que ja no es facin servir de forma adecuada. Java no permet construir objectes manualment, té una funció anomenada *GarbageCollector* que s'encarrega d'eliminar objectes que no estiguin sent referenciats per tal d'alliberar memoria.

- **HashMap**

  Per aquesta implementació, la idea és tenir un sol HashMap on les claus son strings formatejats tals que: "A1", "C3", etc; i els valors evidentment, com a objectes de tipus Cel·la.

  Com acabem de mencionar, la principal avantatge que te sobre les matrius es que només es guardaran les cel·les que haguem afegit explícitament, ergo l'espai ocupat serà $\Theta(\boldsymbol{x})$.

  Un altre punt positiu es que la cerca de cel·les dins d'un full sempre tindrà cost lineal $\Theta(\boldsymbol{x})$ on les implementacions amb matrius havien de comprovar totes les posicions, tinguin cel·les o no, per tant, tenia un cost quadratic $\Theta(\boldsymbol{n^2})$.

  A la hora d'afegir o eliminar files/columnes, la idea era recorrer tot el HashMap, comprovant si les claus havien estat desplaçades o no, i reinsertant els elements amb les claus actualitzades. Tot això amb un cost lineal en relació a les cel·les: $\Theta(\boldsymbol{x})$.

  La tenir un diccionari amb una sola clau té dos grans inconvenients.
  El primer, i potser més menor, vé per l'implementació del HashMap, la qual fa servir un *Binary Search Tree*, ordena els elements en un arbre binari, cosa que fa que el temps d'accés de mitja sigui $\Theta(\mathbf{1})$, pero en el pitjor dels casos (un cas que realisticament és molt improbable), podria ser lineal amb els objectes del diccionari $\Theta(\boldsymbol{x})$.

  L'altre problema es que a la hora d'ordenar per columnes, trobar els elements a ordenar seria molt difícil, partint de que tenim dues cantonades, és a dir, podem ordenar més d'una columna per crida, si intentem trobar valor associats a les claus incloses en l'àrea seleccionada per les cantonades anteriors, i després, fem un *Insertion Sort* dels elements trobats i els tornem a afegir al seu lloc segons l'ordre. Al final la ordenació te cost quadratic en relació als elements seleccionats per

columna que en el pitjor dels casos seria $\Theta(n^2)$, pero la abstractesa i llibertat del HashMap feia més complicat la implementació de certes operacions.

- **SortedMaps aniuats**

  Finalment arribem a la implementació actual, dos SortedMaps aniuats, on el diccionari "exterior" representa les columnes i l'interior les files, d'aquesta manera les operacions sobre columnes, principalment l'ordenació són més fàcils de implementar e interpretar.

  El SortedMap, com el seu nom indica, té l'avantatge de que els seus elements s'ordenen segons el valor de les seves claus, el que fa que sigui més fàcil optimitzar aquelles funcions que requereixin buscar els elements anteriors o posteriors a una certa posició, com es el cas de afegir/eliminar files i columnes.

  L'altre gran avantatge del SortedMap, o més pròpiament dit, el TreeMap, la classe de Java que l'implementa, és que fa servir un *Black-Red Binary Tree*, un tipus d'arbre binari balancejat que fa que les cerques, tant de columnes com de les files siguin $\Theta(\log(n))$ en el cas de columnes i $\Theta(\log(n))$ per les files, tenint en compte que el cas promig, tant les files com les columnes seran més petits que el nombre d'elements ($x \leq n$),
  aleshores, el temps asimptòtic d'accés quedaria tal que $\Theta(\log(n)) < \Theta(\log(x))$

4.1.3 Interpret

La classe Interpret, com ja hem esmentat anteriorment, només es tracta d'un parser de tipus de dades i també gestiona totes les funcions entrades per l'usuari i s'encarrega d'anar a buscar el resultat, per tant, les estructures de dades emprades han sigut molt senzilles:

- **Vectors**

  Hem utilitzat vectors de String i de Double per guardar la coordenada de les cel·les referenciades i els valors numèrics de la cel·la, respectivament. Hem emprat vector en comptes d'un array, ja que volem crear una estructura buida i anar afegint els valors a mida que es van trobant.

4.1.4 Document

En aquesta classe cal utilitzar una estructura per a guardar els fulls que conté el document. Per tenir un accés més ràpid i eficient hem obtat per un diccionari de

manera que podem accedir per clau al full que volem. Més concretament hem emprat un TreeMap<Integer, Full>:

- **TreeMap**
  Tot i que al principi vam considerar emprar un set, vam acabar agafant un TreeMap amb clau Integer, que representa la *id* del full, i un valor Full, que és el full amb aquesta *id* que té per clau.

4.1.5 CtrlClipboard

En aquest controlador hem hagut de crear una estructura bidimensional que guardés Cel·les. Per fer això hem emprat una matriu d'arrays de cel·les:

- **Matriu d'arrays de cel·les**
  Hem escollit aquesta estructura, ja que a part de copiar cel·les individualment també copiem blocs, de manera que amb una matriu podem copiar al porta-retalls les cel·les tal i com estan col·locades al full. Hem triat aquesta estructura ja que és molt fàcil anar recorrent les cel·les seleccionades al full i anar-les afegint a la matriu que s'inicialitza buida.
  Hem triat aquesta estructura en comptes d'un array de 2 dimensions, ja que un ArrayList no fa falta definir la mida de l'estructura inicialment i es pot anar afegint valors.