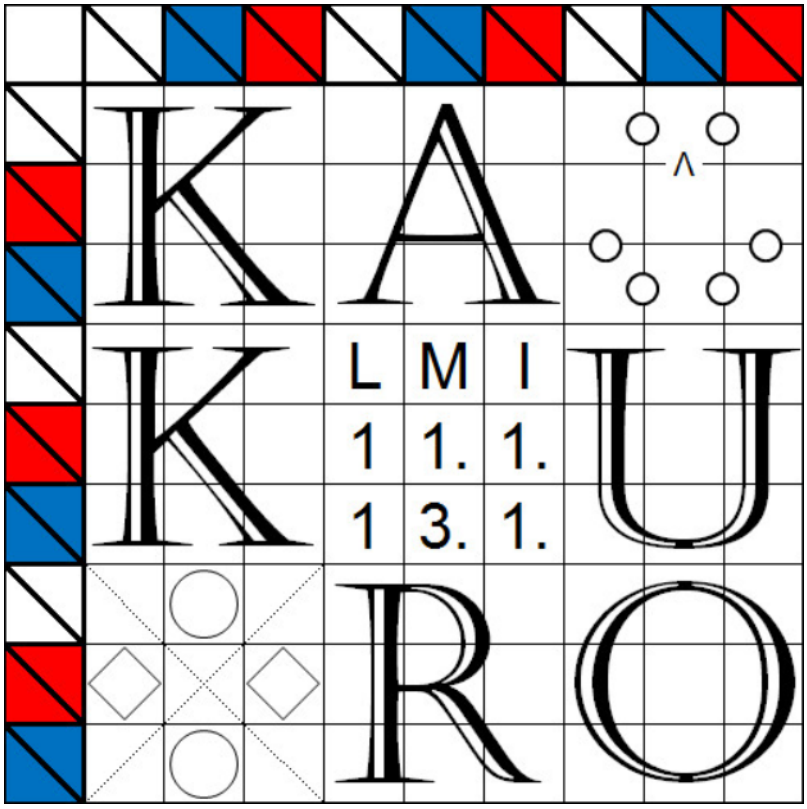
Projecte de Programació

Kakuros - Versió 1.0

Long Chen, Sofian Aoulad Belayachi Djillali,
Joan Domingo Navarro, Dídac Clària Hernández

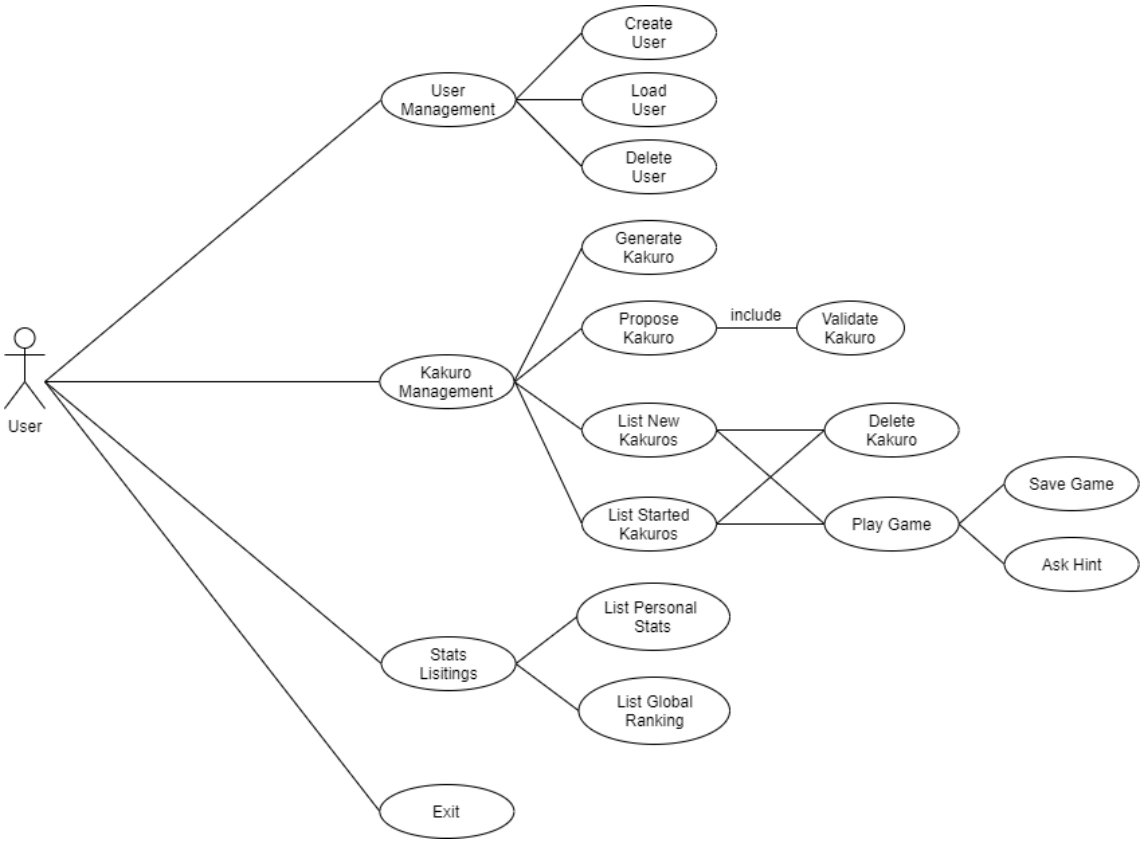
November 2020



Índex

1 Casos d'ús

1.1 Diagrama de casos d'ús



1.2 Descripció breu dels casos d'ús

1.2.1 User management

Nom: Create User
Actor principal: Usuari
Precondició: l'usuari nou no existeix
Detonant: l'usuari vol crear un nou usuari
Escenari principal: l'usuari dona al sistema un usuari i una contrasenya, el sistema crea un nou usuari amb un nom i contrasenya que guarda al control de persistència a disc en un fitxer, l'usuari passa a ser l'actiu.
Extensions: Cap

Nom: Load User
Actor principal: Usuari
Precondició: l'usuari ja existeix.
Detonant: l'usuari vol cambiar d'usuari.
Escenari principal: l'usuari demana canviar d'usuari, selecciona l'usuari a entrar, posa la contrasenya, el sistema actualitzar l'usuari a actiu - no actiu.
Extensions: l'usuari ja és actiu, no passa res.

Nom: Delete User
Actor principal: Usuari
Precondició: l'usuari existeix.
Detonant: l'usuari vol esborrar l'usuari.
Escenari principal: l'usuari demana esborrar l'usuari, selecciona l'usuari a esborrar, posa la contrasenya, el sistema elimina l'usuari, el seu repositori i execucions.
Extensions: Cap

1.2.2 Kakuro Management

Nom: Generate Kakuro
Actor principal: Usuari
Precondició: los condicions de generació de kakuros (mida, caselles blanques, caselles negres, xifres col·locades, dificultat) han de ser vàlides.
Detonant: l'usuari vol crear un nou kakuro i afegir-lo al seu repositori.
Escenari principal: el sistema crea un kakuro amb solució única tenint en compte les condicions de generació i s'afegeix al repositori de l'usuari actiu.
Extensions: Cap

Nom: Propose Kakuro
Actor principal: Usuari
Precondició: Cap
Detonant: l'usuari vol proposar un kakuro.
Escenari principal: el sistema llegeix el kakuro proposat per l'usuari i s'afegeix al seu repositori, assignant-li una dificultat automàticament.
Extensions: Cap
Nom: Validate Kakuro
Actor principal: Usuari
Precondició: l'usuari ha proposat un kakuro.
Detonant: l'usuari vol veure si el kakuro té solució única.
Escenari principal: el sistema resol el kakuro que l'usuari ha proposat, i indica si té solució única, si l'usuari ho demana, la ensenya per pantalla.
Extensions: Cap

Nom: List New Kakuros
Actor principal: Usuari
Precondició: Cap
Detonant: l'usuari vol consultar els kakuros no iniciats del seu repositori.
Escenari principal: el sistema mostra la llista de kakuros no començats. Si un kakuro s'ha començat, apareixerà modificat a la llista de kakuros començats, i buit a la llista de nous kakuros.
Extensions: si no hi ha cap kakuro, apareix una llista buida

Nom: List Started Kakuros
Actor principal: Usuari
Precondició: Cap
Detonant: l'usuari vol consultar els kakuros ja començats del seu repositori.
Escenari principal: el sistema mostra la llista de kakuros començats.
Extensions: si no hi ha cap kakuro, apareix una llista buida

Nom: Delete Kakuro
Actor principal: Usuari
Precondició: el kakuro que es vol esborrar existeix.
Detonant: l'usuari vol esborrar un kakuro.
Escenari principal: el sistema esborra el fitxer del kakuro i les seves aparicions en les llistes, en ranking i en estadística personal.
Extensions: Cap

Nom: Play Game
Actor: Usuari
Precondició: el kakuro existeix.
Detonant: l'usuari vol completar un kakuro, seguir-lo completant o demanar que el completi el sistema.
Escenari principal: s'inicia l'execució d'un kakuro per un usuari, amb un id del joc, i inicia els contadors de temps, número de pistes i puntuació. El pot resoldre un usuari o el sistema dependent del booleà "automàtic".
Excepcions: Cap
Nom: Save Game
Actor: Usuari
Precondició: Cap
Detonant: l'usuari vol guardar el progrés d'un joc.
Escenari principal: l'execució del joc es pausa, igual que el contador de temps, i es guarda el transcurs de l'execució. Es guardarà en un fitxer per ser continuada en un futur si no està completat i s'afegirà a la llista de començats, altrament també s'afegiran els punts als rankings personal i global.
Excepcions: Cap

Nom: Ask Hint
Actor: Usuari
Precondició: Cap
Detonant: l'usuari vol que el sistema li completi una casella blanca.
Escenari principal: el sistema ofereix a l'usuari un valor de casella blanca que no estigui omplert. Cada pista que s'ofereix penalitza fent baixar els punts. Al demanar totes les pistes possibles el sistema mostrarà el kakuro resol i el kakuro tindrà el booleà "automàtic" a cert.
Excepcions: Cap

1.2.3 Stats Listing

Nom: List Global Ranking
Actor: Usuari
Precondició: Cap
Detonant: l'usuari vol veure el ranking global
Escenari principal: el sistema mostra a l'usuari el Ranking de tots els usuaris, mostrant la seva puntuació màxima i el número de kakuro resolts
Excepcions: Cap

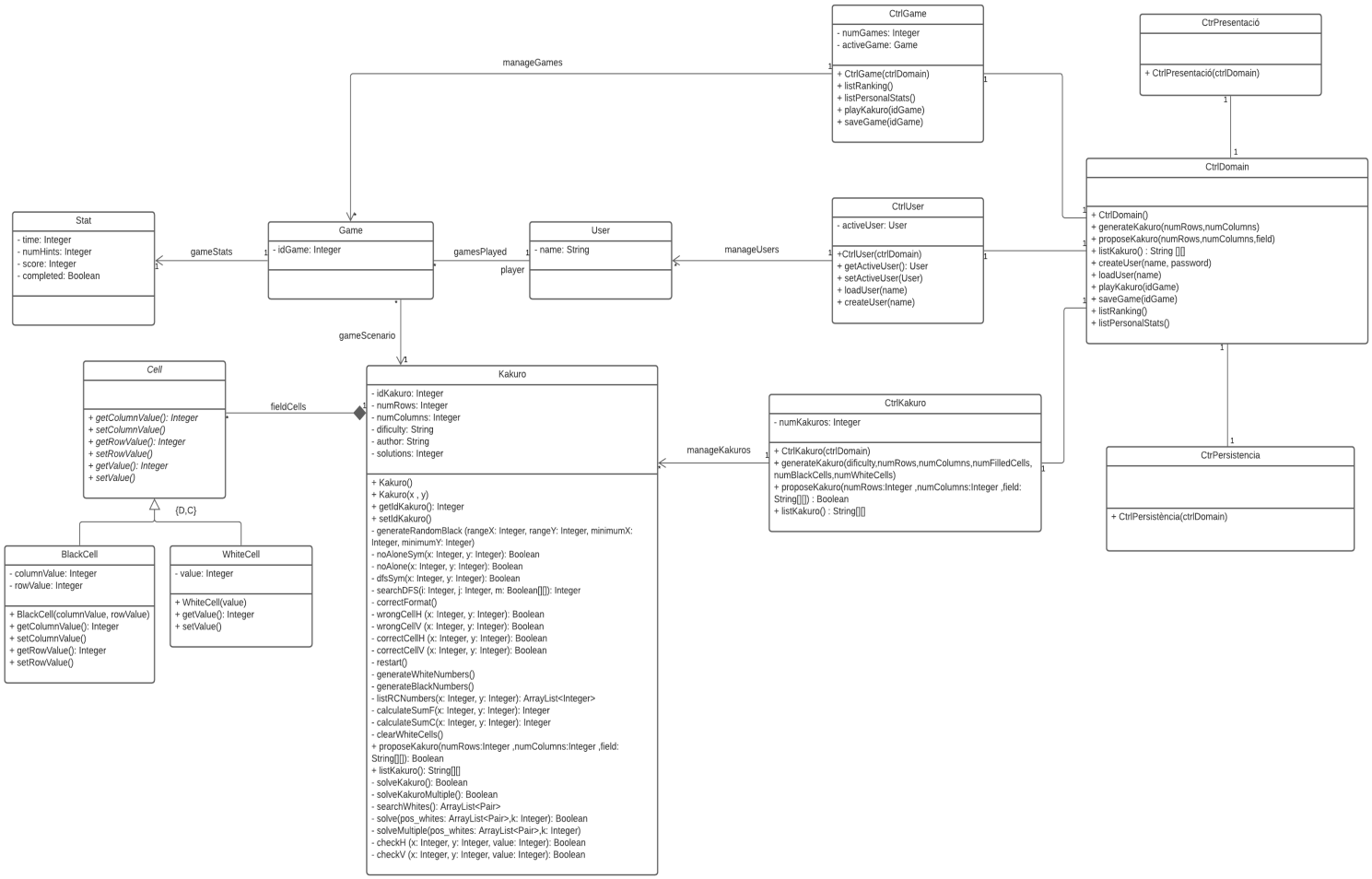
Nom: List Personal Stats
Actor: Usuari
Precondició: Cap
Detonant: l'usuari vol veure la seva estadística personal
Escenari principal: el sistema mostra el ranking personal de l'usuari, composat pels punts màxims aconseguits en cada kakuro executat.
Excepcions: Cap

1.2.4 Exit

Nom: Exit
Actor: Usuari
Precondició: Cap
Detonant: sortir del sistema.
Escenari principal: cap usuari ni cap joc està actiu, acaba l'execució.
Excepcions: Cap

2 Model conceptual de les dades

2.1 Diagrama UML estàtic complet del model



Degut a que potser no es pot visualitzar bé hem afegit dins de la carpeta DOCS la imatge original del diagrama.

2.2 Breu descripció de cada classe

La descripció i documentació de les classes, atributs i mètodes està dins dels propis fitxers del codi font. Per a documentar-ho hem fet ús de l'eina Javadoc i es pot veure el resultat de la documentació dins de la carpeta DOCS/Javadoc.

3 Breu descripció de les estructures de dades i algorismes utilitzats

3.1 Estructures de dades emprades

El nostre projecte consisteix en utilitzar el controlar de Domini, el qual interactua amb tres altres controladors: el controlar de Game, el controlador d'Usuari i el controlador de Kakuro.

El controlador de Game s'encarrega d'inicialitzar el joc de kakuro identificat o de continuar el seu darrer intent, i també té la funcionalitat de guardar l'estat d'un joc de kakuro actiu. A més a més, permet consultar el ranking global i l'estadística personal del jugador.

Quan un joc de kakuro es passa a ser actiu, el controlador d'Usuari activa al mateix moment l'usuari que l'està jugant.

Per l'últim, el controlador de Kakuro proporciona certes funcionalitats principals com: consultar l'estat d'un kakuro, generar kakuro, validar i proposar kakuro (explicats en el següent apartat)

Per tal de poder funcionar el sistema, hem utilitzat les estructures de dades següents:

- Una matriu de classe Cell per guardar totes les caselles (blanques i negres) del kakuro: Cell[][]
- Una classe addicional amb l'objectiu de poder guardar una posició de dues dimensions: Class Pair
- Un arrayList per guardar les posicions de les caselles blanques del kakuro: ArrayList <Pair>
- Un arrayList que guarda les posicions segures de les caselles blanques per tal de generar el format correcte del kakuro: ArrayList <Pair>
- Una matriu de string per guardar i consultar les informacions del kakuro: String [][]
- Un arrayList per guardar els possibles valors (1...9) a una casella blanca: ArrayList<Integer>
- Una matriu de tipus booleà que ens ajuda a determinar quines posicions de les caselles blanques són visitades i així afavorir la busca DFS: boolean [][]

3.2 Explicació algoritmes

3.2.1 Resoldre Kakuros

Per tal de poder trobar una solució del kakuro, hem utilitzat una funció recursiva BACKTRACKING amb Força Bruta, és a dir, per cada crida recursiva, anem provant un nombre entre 1 i 9 a cada casella blanca sempre comprovant que compleixi las normes del kakuro, és a dir, que no es repeteixi a la seva fila o columna i que la seva suma correspongui a la de les caselles negres. En un moment determinat, si la funció retorna false, es fa un BACKTRACKING: tornar al pas anterior i després eliminar el valor assignat a aquella casella blanca i començar pel següent número, i així successivament fins que arribi al final. La funció retornarà TRUE quan s'han assignat valors a totes les caselles blanques, doncs significa que existeix una solució en aquell kakuro, altrament retornarà FALSE.

Per la següent versió del projecte, pretenem millorar l'algorisme de resoldre kakuros, amb l'ajuda d'una nova classe Combinació que contindrà totes les possibles combinacions per cada numero de caselles blanques de tots els possibles nombres que poden ocupar la casella negra final. La idea de millora és, seguint el principi de que un kakuro te una solució única i per tant cada casella blanca te un valor únic, recórrer el kakuro perquè s'introdueixi valors únics que ja es puguin saber a les caselles, afegint més d'un valor de casella blanca per cada estat del problema.

3.2.2 Generar Kakuros

El generador del kakuro consisteix principalment en dues fases:

A la primera fase, generem el format del kakuro, és a dir, creem primer la taula amb dimensió indicada als paràmetres, ficant les caselles de la primera fila i columna negres, a continuació es generen aleatòriament a cada iteració dues caselles negres simètriques a 180^{o}, comprovant que quedi més d'una casella blanca seguida i que totes les caselles blanques estiguin connectades. Després mirem totes les caselles negres de la taula que tinguin més de 9 caselles blanques i les corregim posant-hi una casella negra possible; si no es pot posar-n'hi una llavors netejarem la taula i repetirem el procés des de l'inici.

A la segona fase, un cop generat el format del kakuro, afegim aleatòriament un nombre a cada casella blanca i calculem la suma de la seva fila o columna corresponent, afegint-la a la casella negra que pertoca. A continuació utilitzem el kakuro_solver que tenim implementat per a determinar si el kakuro creat si té únicament una solució; si no, netegem la taula i tornem a repetir el procés fins que trobi una solució única.

Es pretén millorar el generador de kakuros per la següent versió del projecte. Ara mateix hi han dues idees a la taula. La primera és que si el kakuro_solver especifica que la solució no és única, modificar les caselles blanques introduïdes aleatòriament amb altres valors de la classe Combinació, cambiant si cal també valors de caselles negres. La segona és que els nombres de les caselles blanques es generin mitjançant solucions parcials de totes les possibles combinacions, amb una funció heurística que decideixi les millores solucions fins omplir el kakuro. Ambdues solucions arriben a generar la solució única del kakuro sense la necessitat de reiniciar el procés.

4 Relació de les classes implementades per cada membre del grup

Long Chen:

- Kakuro
- Main
- CtrlKakuro
- Pair

Sofian Aoulad Belayachi Djillali:

- kakuroTEST
- BlackCellTEST

Joan Domingo Navarro:

- Combinació
- WhiteCellTEST

Dídac Clària Hernández:

- Cell
- WhiteCell
- BlackCell
- CtrlDomain
- CtrlPersistencia
- CtrlPresentacio
- CtrlKakuro
- DriverCtrlKakuro
- DriverCtrlWhiteCell

Per futures versions del projecte, seran implementats els casos d'us de les classes Stat, Game, CtrlGame, User, CtrlUser, ja que actualment només llencen excepcions.