EDA (ETS d'Inginyeria informàtica). Curs 2020-2021

Pràctica 5. Implementació de la Cua de Prioritat d'un Servidor d'Impresió mitjançant un Montícle (una sesió)

Departament de Sistemes Informàtics i Computació. Universitat Politècnica de València

1. Objectius

El principal objectiu d'aquesta pràctica és que l'alumne aplique al disseny d'una aplicació concreta els conceptes sobre Cua de Prioritat i Monticle Binari que ha estudiat al Tema 5 de l'assignatura. Específicament, al concloure aquesta pràctica l'alumne haurà de ser capaç de ...

- Dissenyar una classe Java que represente un Monticle Binari amb l'arrel en 0, una implementació del Model Cua de Prioritat que empren un gran nombre d'aplicacions en la vida real.
- Implementar la Cua de Prioritat que utilitza una aplicació de Simulació d'un Servidor d'impressió eficient.

2. Descripció del problema

Un servidor d'impressió és, com el seu nom indica, un servidor que connecta (al menys) una impressora a xarxa per a que qualsevol dels seus clients puga imprimir els seus treballs en ella. Com en un moment donat hi poden haver diversos treballs a imprimir i, òbviament, una impressora només pot imprimir un a l'hora, aquest tipus de servidor ha de poder emmagatzemar i gestionar els treballs en espera d'impressió.

El model de gestió més senzill que pot utilitzar un servidor d'impressió és el FIFO (*First In, First Out*), i.e. el model d'una Cua: una Cua emmagatzema, en ordre d'arribada, els treballs a imprimir, de manera que el primer de la Cua (*First In*) també és el primer que s'elimina d'aquesta per a ser imprès (*First Out*), una vegada que la impressora quede lliure. Ara bé, aquesta gestió FIFO presenta un problema d'eficiència quan, per exemple, els treballs que han arribat abans a l'servidor tenen moltes més pàgines que els que han arribat més tard: la cua pot fer-se eterna! Afortunadament, per a millorar significativament el temps mitjà d'espera dels treballs al servidor només cal fer el següent:

a. Assignar als treballs a imprimir una prioritat que siga inversament proporcional al seu nombre de pàgines.

b. Gestionar els treballs en espera d'impressió que emmagatzema el servidor amb una Cua de Prioritat implementada amb un Monticle Binari.

Per comprovar-ho, en aquesta pràctica es realiatzarà la implementació d'un simulador dels dos tipus de servidors d'impressió descrits, que anomenarem *Servidor Cua* i *Servidor Cua de Prioritat*, i es comparerà el temps mitjà que s'espera un treball a imprimir en cada un d'ells.

2.1. Les classes d'una aplicació de simulació d'un servidor d'impresió

Les principals classes que componen una aplicació de simulació d'un servidor d'impressió són:

- `Trabajo`, representant un treball a imprimir (*print job*). Un treball `TÉ UN` títol que l'identifica, un cèrt nombre de pàgines i l'instant de temps (en segons) en què "entra" (es enviat) a servidor d'impressió. Convé ressaltar, a més, que el mètode `compareTo` d'aquesta classe permet establir quin de dos treballs a imprimir donats ha de ser atès amb major prioritat: aquell que tinga un nombre de pàgines menor.
- `ServidorDeImpresion` que, com el seu nom indica, representa un servidor d'impressió (*print server*). Per ser més exactes, atès que un servidor d'impressió pot gestionar de maneres diferents dels treballs a l'espera de ser impresos que emmagatzema, aquesta classe és una interfície Java que estableix les operacions que realitza un servidor d'impressió; a saber:

- `insertar(Trabajo t)`, que afegeix un nou treball al servidor;
- `hayTrabajos()`, que comprova si encara queden treballs al servidor;
- `getTrabajo()`, que retorna el treball del servidor que serà imprès;
- `imprimirTrabajo()`, que elimina del servidor el treball que serà imprès i retorna els segons que aquest tardarà a imprimir-se, d'acord amb la velocitat de la impressora.

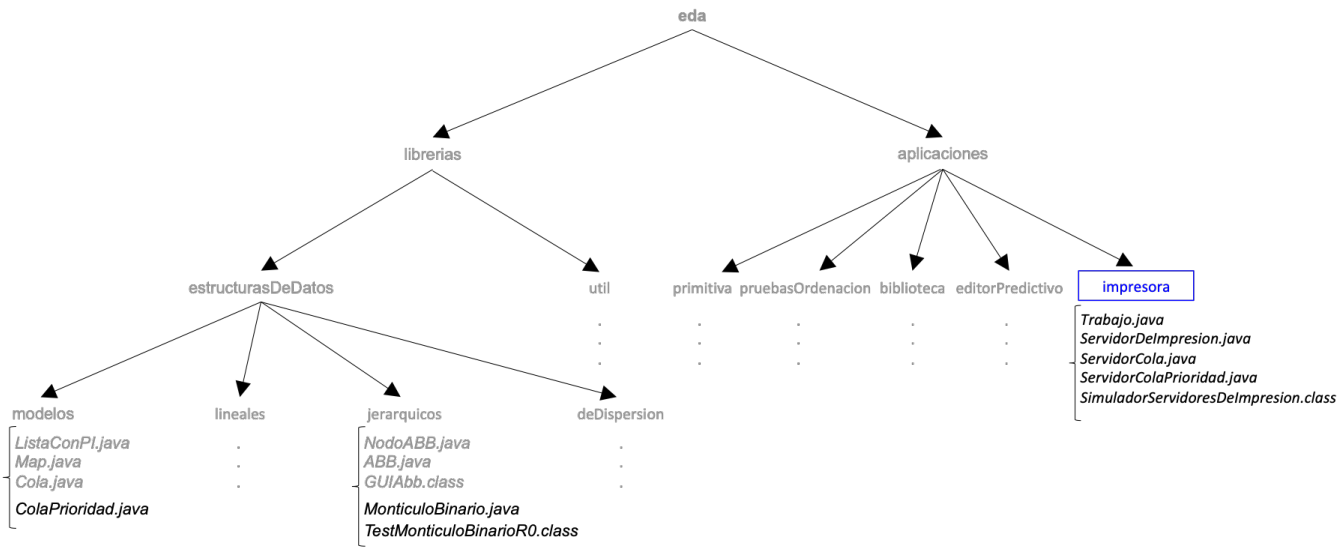
A més, en aquesta interfície es defineix la velocitat de la impressora associada a un servidor d'impressió, o nombre de pàgines per minut que pot imprimir (`PAGINAS_POR_MINUTO`.)

- `ServidorCola`, que representa un *Servidor Cua*. Així, implementa la interfície `ServidorDeImpresion` i `TÉ UNA Cola<Trabajo> c` com a únic atribut.
- `ServidorColaPrioridad`, que representa un *Servidor Cua de Prioritat*. Per això, aquesta classe implementa la interfície `ServidorDeImpresion` i `TÉ UNA ColaPrioridad<Trabajo> cP` com únic atribut.
- `SimuladorServidoresDeImpresion`, un programa Java que simula el funcionament de dos servidors d'impressió, un *Servidor Cua* i un *Servidor Cua de Prioritat*, per calcular el temps mitjà d'espera que requereix la impressió d'un treball en cada un d'ells; basant-se aquest valor es determinarà quin dels dos servidors és el més eficient.

3. Activitats

Abans de portar a terme les activitats que es proposen en aquest apartat, cal que l'alumne actualitze l'estructura de paquets i fitxers del seu projecte *BlueJ eda* seguint els passos que s'indiquen a continuació.

- Entrar en el projecte *BlueJ eda*.
- Obrir el paquet *aplicaciones* i crear en ell un nou paquet de nom *impresora*, que a de contenir les classes de l'aplicació a desenvolupar en aquesta pràctica.
- Eixir del *BlueJ* **seleccionant la opció *Eixir* de la pestanya *Projecte***.
- Descarregar els fitxers (`.java`) disponibles al *PoliformaT* en els seues corresponents directoris, tal com mostra la figura següent:



- Entrar en el projecte *BlueJ eda* i compilar la classe `ColaPrioridad`, situada al paquet *modelos* de *librerias.estructurasDeDatos*.
- Eixir del *BlueJ*
- Entrar en el projecte *BlueJ eda* i situar-se en el seu paquet *librerias.estructurasDeDatos.jerarquicos*.

3.1. Implementació d'un Montícule Binari amb Arrel en 0

La classe `MonticuloBinario` vista en teoria representa un Monticle Binari amb Arrel en 1 simplement perquè, així, es simplifica el càlcul de la posició de el pare i els fills del seu i-èssim node (per Nivells).

No obstant això, la implementació de Monticle Binari que fan servir la majoria de les aplicacions a la vida real és amb una arrel 0, potser perquè estan escrites en llenguatges en els quals la primera posició d'un array és la zero (com C, Java o Python). Per aquest motiu, principalment, en aquesta activitat l'alumne ha d'implementar en el paquet *jerarquicos* una nova classe, de nom `MonticuloBinarioR0`, que represente un Monticle Binari amb una arrel 0; per això, pot modificar el codi de la classe `MonticuloBinario` (disponible en el mateix paquet) perquè el seu i-èssim node (per Nivells) passe a tindre...

- el seu fill esquerre en la posició $2 * i + 1$, si $2 * i + 1 < talla$;
- el seu fill dret en la posició $2 * i + 2$, si $2 * i + 2 < talla$;
- el seu pare en la posició $(i - 1)/2$, si $i \neq 0$.

Per validar la classe `MonticuloBinarioR0`, n'hi ha prou en executar el programa `TestMonticuloBinarioR0`, disponible en el mateix paquet.

3.2. Execució de l'aplicació de simulació d'un servidor d'impressió

Abans de poder executar l'aplicació de simulació, situada en el paquet *aplicaciones/impresora*, l'alumne ha de completar el codi de les classes `Trabajo` i `ServidorColaPrioridad`; per això ha de tindre en compte el següent:

- Un *Servidor Cola de Prioridad* emmagatzema i gestiona elements de tipus `Trabajo`.
- Un *Servidor Cola de Prioridad* només difereix d'un *Servidor Cua* en el model que fa servir per gestionar els treballs en espera d'impressió, de manera que el codi de les classes `ServidorColaPrioridad` i `ServidorCola` és molt similar.
- El nombre de pàgines d'un treball estableix la seva prioritat.
- La classe `MonticuloBinarioR0` és una Implementació eficient de el model `ColaPrioridad`.

Fet això, l'aplicació es pot validar executant el programa `SimuladorServidoresDeImpresion`: si el codi de les classes `Trabajo` i `ServidorColaPrioridad` és correcte apareix a la pantalla una petita simulació on es compara el temps mitjà d'espera que espera a ser imprès un treball en un *Servidor Cola* i en un *Servidor Cola de Prioridad*. En aquesta simulació es mostra una línia per cadascun dels treballs, en l'ordre en el que es van imprimint, i amb el format que es detalla a continuació:

```
[175] El nombre del viento - P. Rothfuss (55 pag.) Entra al servidor: 59 (6 seg. de espera)
```

- `[175]` ⟶ Instante de tiempo (en segundos) en el que finaliza la impresión del trabajo.
- `El nombre del viento - P. Rothfuss` ⟶ Título del trabajo.
- `(55 pag.)` ⟶ Número de páginas del trabajo.
- `Entra al servidor: 59` ⟶ Instante de tiempo (en segundos) en el que el trabajo "entra" (es enviado) al servidor de impresión.
- `(6 seg. de espera)` ⟶ Tiempo que el trabajo espera en el servidor de impresión.

NOTA: cada vegada que s'executa el programa `SimuladorServidoresDeImpresion` es genera de forma aleatòria un conjunt de treballs i, per tant, la simulació que apareix en pantalla és sempre diferent de les anteriors.