

Practica de PRO2. MEGA-PRO2-LOAD
versión 1 21-may-2012

Generado por Doxygen 1.7.3

Lunes, 21 de Mayo de 2012 16:26:18

Índice general

1	MEGA-PRO2-LOAD	1
2	Índice de clases	3
2.1	Lista de clases	3
3	Índice de archivos	5
3.1	Lista de archivos	5
4	Documentación de las clases	7
4.1	Referencia de la Clase Cjt_pelicula	7
4.1.1	Descripción detallada	8
4.1.2	Documentación del constructor y destructor	8
4.1.2.1	Cjt_pelicula	8
4.1.2.2	~Cjt_pelicula	8
4.1.3	Documentación de las funciones miembro	8
4.1.3.1	leer_peliculas	8
4.1.3.2	getPeliculaId	9
4.1.3.3	mida	9
4.1.3.4	escribir_peliculas	10
4.1.4	Documentación de los datos miembro	10
4.1.4.1	vecpel	10
4.2	Referencia de la Clase Cjt_peticion	10
4.2.1	Descripción detallada	11
4.2.2	Documentación del constructor y destructor	11
4.2.2.1	Cjt_peticion	11
4.2.2.2	~Cjt_peticion	12
4.2.3	Documentación de las funciones miembro	12
4.2.3.1	pushList	12
4.2.3.2	anadir_peticion	12
4.2.3.3	incrementId	13
4.2.3.4	mida	13
4.2.3.5	peticion_sinFinalizar	14
4.2.3.6	pelicula_solicitada	14
4.2.4	Documentación de los datos miembro	15
4.2.4.1	cont	15
4.2.4.2	listpet	15
4.3	Referencia de la Clase Cjt_servidor	15
4.3.1	Descripción detallada	17
4.3.2	Documentación del constructor y destructor	17

4.3.2.1	Cjt_servidor	17
4.3.2.2	~Cjt_servidor	17
4.3.3	Documentación de las funciones miembro	18
4.3.3.1	llegir_estructura_arbre	18
4.3.3.2	llegir_serv	18
4.3.3.3	marcaje	19
4.3.3.4	condicio1	19
4.3.3.5	condicio2	21
4.3.3.6	roundUp	22
4.3.3.7	llegir_arbre_serv	22
4.3.3.8	actualizar	23
4.3.3.9	actualizar_servidor	23
4.3.3.10	calcular_tiempoFinal	24
4.3.3.11	numeroServidores	24
4.3.3.12	escriure_servidorOcupats	25
4.3.4	Documentación de los datos miembro	25
4.3.4.1	estructura	25
4.3.4.2	servidors	25
4.4	Referencia de la Clase Pelicula	25
4.4.1	Descripción detallada	26
4.4.2	Documentación del constructor y destructor	26
4.4.2.1	Pelicula	26
4.4.2.2	Pelicula	27
4.4.2.3	~Pelicula	27
4.4.3	Documentación de las funciones miembro	27
4.4.3.1	leer_pelicula	27
4.4.3.2	getId	28
4.4.3.3	getTamano	28
4.4.4	Documentación de los datos miembro	29
4.4.4.1	id	29
4.4.4.2	tam	29
4.5	Referencia de la Clase Peticion	29
4.5.1	Descripción detallada	30
4.5.2	Documentación del constructor y destructor	30
4.5.2.1	Peticion	30
4.5.2.2	Peticion	31
4.5.2.3	Peticion	31
4.5.2.4	~Peticion	32
4.5.3	Documentación de las funciones miembro	32
4.5.3.1	setTf	32
4.5.3.2	setId	32
4.5.3.3	getId	33
4.5.3.4	getT_i	33
4.5.3.5	getT_f	33
4.5.3.6	getIdPelicula	34
4.5.4	Documentación de los datos miembro	34
4.5.4.1	id	34
4.5.4.2	t_i	34
4.5.4.3	t_f	34
4.5.4.4	idpelicula	34

4.6	Referencia de la Clase Servidor	35
4.6.1	Descripción detallada	36
4.6.2	Documentación del constructor y destructor	36
4.6.2.1	Servidor	36
4.6.2.2	Servidor	37
4.6.2.3	~Servidor	37
4.6.3	Documentación de las funciones miembro	37
4.6.3.1	leer_servidor	37
4.6.3.2	actualizar	38
4.6.3.3	actualizar_pelicula	38
4.6.3.4	setTiempo	39
4.6.3.5	setIdpet	39
4.6.3.6	addPeliculas	39
4.6.3.7	deletePeliculas	40
4.6.3.8	getId	40
4.6.3.9	getTiempo	41
4.6.3.10	getAncho	41
4.6.3.11	getIdpet	41
4.6.3.12	getPeliculas	42
4.6.3.13	contPelicula	42
4.6.4	Documentación de los datos miembro	42
4.6.4.1	id	42
4.6.4.2	ancho	42
4.6.4.3	tiempo	43
4.6.4.4	idpet	43
4.6.4.5	lpet	43
5	Documentación de archivos	45
5.1	Referencia del Archivo Cjt_pelicula.cpp	45
5.2	Referencia del Archivo Cjt_pelicula.hpp	46
5.3	Referencia del Archivo Cjt_peticion.cpp	47
5.4	Referencia del Archivo Cjt_peticion.hpp	47
5.5	Referencia del Archivo Cjt_servidor.cpp	48
5.6	Referencia del Archivo Cjt_servidor.hpp	48
5.7	Referencia del Archivo Pelicula.cpp	49
5.8	Referencia del Archivo Pelicula.hpp	49
5.9	Referencia del Archivo Peticion.cpp	50
5.10	Referencia del Archivo Peticion.hpp	50
5.11	Referencia del Archivo pro2.cpp	51
5.11.1	Documentación de las funciones	51
5.11.1.1	main	51
5.12	Referencia del Archivo Servidor.cpp	52
5.13	Referencia del Archivo Servidor.hpp	53

Capítulo 1

MEGA-PRO2-LOAD

Capítulo 2

Índice de clases

2.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

Cjt_pelicula (Representa un conjunto de Pelicula)	7
Cjt_peticion (Representa un conjunto de Peticion)	10
Cjt_servidor (Representa un conjunto de Servidor)	15
Pelicula (Representa una Pelicula)	25
Peticion (Representa una Peticion)	29
Servidor (Representa un Servidor)	35

Capítulo 3

Indice de archivos

3.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

Cjt_película.cpp	45
Cjt_película.hpp	46
Cjt_peticion.cpp	47
Cjt_peticion.hpp	47
Cjt_servidor.cpp	48
Cjt_servidor.hpp	48
Película.cpp	49
Película.hpp	49
Peticion.cpp	50
Peticion.hpp	50
pro2.cpp	51
Servidor.cpp	52
Servidor.hpp	53

Capítulo 4

Documentación de las clases

4.1. Referencia de la Clase Cjt_pelicula

Representa un conjunto de [Pelicula](#).

Métodos públicos

- [Cjt_pelicula](#) ()
Creadora por defecto. Se ejecuta automaticamente al declarar un Cjt_peliculas.
- [~Cjt_pelicula](#) ()
Destructora.
- void [leer_peliculas](#) ()
Lee las peliculas.
- [Pelicula](#) [getPeliculaId](#) (int &id)
Consulta de la pelicula segun del ID.
- int [mida](#) ()
Consulta el mida del Cjt_peliculas.
- void [escribir_peliculas](#) ()
Escribe la lista de peliculas.

Atributos privados

- vector< [Pelicula](#) > [vecpel](#)
Vector que contiene las peliculas.

4.1.1. Descripción detallada

Representa un conjunto de [Pelicula](#).

Definición en la línea 10 del archivo Cjt_pelicula.hpp.

4.1.2. Documentación del constructor y destructor

4.1.2.1. Cjt_pelicula::Cjt_pelicula ()

Creadora por defecto. Se ejecuta automaticamente al declarar un Cjt_peliculas.

Precondición

cierto

Postcondición

El resultado es un Cjt_peliculas vacio

Definición en la línea 4 del archivo Cjt_pelicula.cpp.

```
    {  
    vecpel = vector<Pelicula> (0);  
    }
```

4.1.2.2. Cjt_pelicula::~Cjt_pelicula ()

Destructora.

Definición en la línea 8 del archivo Cjt_pelicula.cpp.

```
{ }
```

4.1.3. Documentación de las funciones miembro

4.1.3.1. void Cjt_pelicula::leer_peliculas ()

Lee las peliculas.

Precondición

cierto

Postcondición

cierto

Definición en la línea 10 del archivo Cjt_pelicula.cpp.

```
int n;
n = readint();
vecpel = vector<Película> (n);
for(int i = 0; i < n; ++i) {
    Película p;
    p.leer_película(i);
    vecpel[i] = p;
}
```

4.1.3.2. Película Cjt_película::getPelículaId (int & id)

Consulta de la película según del ID.

Precondición

$1 \leq id \leq \text{tamaño Cjt_películas}$

Postcondición

[Película](#)

Definición en la línea 21 del archivo Cjt_película.cpp.

```
{
int i = 0;
bool trobat = false;
while (i < vecpel.size() and not trobat) {
    if(vecpel[i].getId() == id-1) trobat = true;
    else ++i;
}
return vecpel[i];
}
```

4.1.3.3. int Cjt_película::mida ()

Consulta el mida del Cjt_películas.

Precondición

cierto

Postcondición

mida del Cjt_películas

Definición en la línea 31 del archivo Cjt_película.cpp.

```
{
return vecpel.size();
}
```

4.1.3.4. void Cjt_pelicula::escribir_peliculas ()

Escribe la lista de películas.

Precondición

cierto

Postcondición

cierto

Definición en la línea 35 del archivo Cjt_pelicula.cpp.

```

{
    int n = vecpel.size();
    cout << "Lista de peliculas: " << endl;
    for (int i = 0; i < vecpel.size(); ++i) {
        cout << " ID: " << vecpel[i].getId() << endl;
        cout << " Tamany: " << vecpel[i].getTamano() << endl;
        cout << "-----" << endl;
    }
}

```

4.1.4. Documentación de los datos miembro

4.1.4.1. vector<Película> Cjt_pelicula::vecpel [private]

Vector que contiene las películas.

Definición en la línea 14 del archivo Cjt_pelicula.hpp.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- [Cjt_pelicula.hpp](#)
- [Cjt_pelicula.cpp](#)

4.2. Referencia de la Clase Cjt_peticion

Representa un conjunto de [Peticion](#).

Métodos públicos

- [Cjt_peticion \(\)](#)
Creadora por defecto. Se ejecuta automáticamente al declarar un cjt_peticiones.
- [~Cjt_peticion \(\)](#)
Destructora.

- void `anadir_peticion` (`Peticion` p)
Añadir una peticion al conjunto.
- void `incrementId` (`Peticion` &p)
Autoincremento del id de la peticon (id unico)
- int `mida` () const
Consulta de el numero de peticiones del conjunto.
- void `peticion_sinFinalizar` (int t)
Escribe las peticiones sin finalizar en un cierto tiempo.
- void `pelicula_solicitada` (int t1, int t2, int n)
Escritura de la pelicula mas solicitada entre los dos tiempos.

Métodos privados

- void `pushList` (`Peticion` p)
Metodo para insertar un elemento en la lista ordenado ascendente por tiempo final.

Atributos privados

- int `cont`
Variable para tener el autoincremento del id de la peticion.
- list< `Peticion` > `listpet`
Lista que contiene todas las peticiones aceptadas.

4.2.1. Descripción detallada

Representa un conjunto de `Peticion`.

Definición en la línea 13 del archivo Cjt_peticion.hpp.

4.2.2. Documentación del constructor y destructor

4.2.2.1. `Cjt_peticion::Cjt_peticion ()`

Creadora por defecto. Se ejecuta automaticamente al declarar un cjt_peticiones.

Precondición

cierto

Postcondición

El resultado es una [Petición](#) sin parametros

Definición en la línea 6 del archivo Cjt_peticion.cpp.

```
    {  
        cont = 0;  
        list<Petición> listpet;  
    }
```

4.2.2.2. Cjt_peticion::~~Cjt_peticion ()

Destructora.

Definición en la línea 11 del archivo Cjt_peticion.cpp.

```
{}
```

4.2.3. Documentación de las funciones miembro**4.2.3.1. void Cjt_peticion::pushList (Petición p) [private]**

Metodo para insertar un elemento en la lista ordenado ascendente por tiempo final.

Precondición

lista inicializada, id del servidor

Postcondición

cierto

Definición en la línea 14 del archivo Cjt_peticion.cpp.

```
    {  
        list<Petición>::iterator it = listpet.begin();  
        if (not listpet.empty()) {  
            while (it != listpet.end() and (*it).getT_f() < p.getT_f()){  
                ++it;  
            }  
            listpet.insert(it,p);  
        }  
        else listpet.insert(listpet.begin(),p);  
    }
```

4.2.3.2. void Cjt_peticion::anadir_peticion (Petición p)

Añadir una petición al conjunto.

Precondición

cierto

Postcondición

cierto

Definición en la línea 25 del archivo Cjt_peticion.cpp.

```
                                {  
    pushList(p);  
}
```

4.2.3.3. void Cjt_peticion::incrementId (Peticion & p)

Autoincremento del id de la peticon (id unico)

Precondición

peticion inicializada

Postcondición

cierto

Definición en la línea 29 del archivo Cjt_peticion.cpp.

```
                                {  
    p.setId(cont);  
    ++cont;  
}
```

4.2.3.4. int Cjt_peticion::mida () const

Consulta de el numero de peticiones del conjunto.

Precondición

cierto

Postcondición

numero de peticiones

Definición en la línea 34 del archivo Cjt_peticion.cpp.

```
                                {  
    return listpet.size();  
}
```

4.2.3.5. void Cjt_peticion::peticion_sinFinalizar (int t)

Escribe las peticiones sin finalizar en un cierto tiempo.

Precondición

variable de tiempo

Postcondición

cierto

Definición en la línea 38 del archivo Cjt_peticion.cpp.

```

{
    list<Peticion> l(listpet);
    bool b = false;
    cout << "Peticiones pendientes " << endl;
    if (not listpet.empty()) {
        list<Peticion>::iterator it;
        for(it = listpet.begin(); it != listpet.end() ; ++it) {
            if((*it).getT_f() > t) {
                b = true;
                cout << (*it).getId();
                cout << " " << (*it).getIdPelicula();
                cout << " " << (*it).getT_i();
                cout << " " << (*it).getT_f() << endl;
            }
        }
    }
    if (not b) cout << "0" << endl;
}

```

4.2.3.6. void Cjt_peticion::pelicula_solicitada (int t1, int t2, int n)

Escritura de la pelicula mas solicitada entre los dos tiempos.

Precondición

t1 <= t2 , n = Cjt_peliculas.mida()

Postcondición

cierto

Definición en la línea 57 del archivo Cjt_peticion.cpp.

```

{
    //Pre: t1 <= t2, n es el numero de peliculas en el Conjunto de peliculas
    vector<int> pel(n,0); //contiene el numero de peticiones
    pair<int,int> max(n,0); //first=id second=numero de peticions
    list<Peticion>::iterator it = listpet.begin();

    //Inv: pel= vector con la misma mida que el conjunto de peliculas donde cada po

```

```

        sicion del vector(idPelicula-1) indica el numero de
//  peticiones a esa pelicula
//  max = estructura para tener guardado el id y el numero de peticiones maximo
//  de la lista de peticiones.
//  it = iterador que recorre la lista desde inicio a fin.
while (it != listpet.end()) {
    if((*it).getT_i() >= t1 and (*it).getT_i() <= t2) {
        ++pel[(*it).getIdPelicula()-1];
        if(max.second <= pel[(*it).getIdPelicula()-1]) {
            if(max.second == pel[(*it).getIdPelicula()-1]) {
                if (max.first > (*it).getIdPelicula()) {
                    max.first = (*it).getIdPelicula();
                }
            }
        }
    }
    else {
        max.first = (*it).getIdPelicula();
        max.second = pel[(*it).getIdPelicula()-1];
    }
    ++it;
}
if (max.second == 0) max.first = 0;
cout << "Pelicula mas solicitada" << endl;
cout << max.first << " " << max.second << endl;
//Post: Mostramos por pantalla la pelicula mas solicitada con t1 <= ti <= t2
}

```

4.2.4. Documentación de los datos miembro

4.2.4.1. `int Cjt_peticion::cont` [private]

Variable para tener el autoincremento del id de la peticion.

Definición en la línea 18 del archivo Cjt_peticion.hpp.

4.2.4.2. `list<Peticion> Cjt_peticion::listpet` [private]

Lista que contiene todas las peticiones aceptadas.

Definición en la línea 21 del archivo Cjt_peticion.hpp.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- [Cjt_peticion.hpp](#)
- [Cjt_peticion.cpp](#)

4.3. Referencia de la Clase Cjt_servidor

Representa un conjunto de [Servidor](#).

Métodos públicos

- `Cjt_servidor ()`
Creadora por defecto. Se ejecuta automaticamente al declarar una cubeta.
- `~Cjt_servidor ()`
- `void llegir_arbre_serv (int numpel)`
Lectura de la estructura del arbol y de los servidores.
- `void actualizar (int t)`
actualizacion de los servidore si $t >$ tiempo del servidor; tiempo = 0 y estado = false
- `void actualizar_servidor ()`
actualizacion Alta/Baja de peliculas en el servidor
- `void calcular_tiempoFinal (Peticion &pet, Pelicula &pel)`
calculo del tiempo final de la peticion(realiza los calculos apartir de las 3 condiciones)
- `int numeroServidores ()`
consulta de numero de servidores
- `void escriure_servidorOcupats (int t)`
escritura de servidores ocupados en el tiempo t

Métodos privados

- `void llegir_estructura_arbre (Arbre< int > &a, int marca, int &numser)`
Lectura de la estructura del arbol de servidor.
- `void llegir_serv (int &numpel)`
Lectura de un arbol binario de servidor.
- `void marcaje (list< int > &l, int d, int idp)`
metodo para marcar los servidores seleccionados por la peticion el tiempo final y el id de la peticion mostrar por pantalla los servidores seleccionados. : $l = L$, $d \geq 0$, $0 \leq idp < mida$ conjunto peticion
- `bool condicio1 (Arbre< int > &a, list< int > &l, Pelicula &pel, int &ancho, int &alt, int &prof)`
metodo para calcular el tiempo final en una unica unidad de tiempo
- `void condicio2 (Arbre< int > &a, list< int > &l, Pelicula &pel, int &ancho, int &alt, int &prof)`
metodo para calcular el tiempo final cojiendo el mejor ancho de banda segun los criterios

- int [roundUp](#) (int a, int b)

calcula de la division redondeando hacia arriba

Atributos privados

- [Arbre](#)< int > [estructura](#)

Arbol que contiene la estructura de los servidores.

- [vector](#)< [Servidor](#) > [servidores](#)

Vector de servidores.

4.3.1. Descripción detallada

Representa un conjunto de [Servidor](#).

Definición en la línea 15 del archivo Cjt_servidor.hpp.

4.3.2. Documentación del constructor y destructor

4.3.2.1. Cjt_servidor::Cjt_servidor ()

Creadora por defecto. Se ejecuta automaticamente al declarar una cubeta.

Precondición

cierto

Postcondición

es un conjunto de servidores vacio

Definición en la línea 5 del archivo Cjt_servidor.cpp.

```
{ }
```

4.3.2.2. Cjt_servidor::~~Cjt_servidor ()

Definición en la línea 7 del archivo Cjt_servidor.cpp.

```
{ }
```

4.3.3. Documentación de las funciones miembro

4.3.3.1. `void Cjt_servidor::llegir_estructura_arbre (Arbre< int > & a, int marca, int & numser)` [private]

Lectura de la estructura del arbol de servidor.

Precondición

marca de parade de lectura de servidores

Postcondición

a contiene la estructura de servidores

Definición en la línea 10 del archivo Cjt_servidor.cpp.

```
Arbre<int> a1;
Arbre<int> a2;
int x;
cin >> x;
if (x!= marca and numser <= servidores.size()) {
    ++numser;
    llegir_estructura_arbre(a1,marca, numser);
    llegir_estructura_arbre(a2,marca, numser);
    a.plantar(x,a1,a2);
}
}
```

4.3.3.2. `void Cjt_servidor::llegir_serv (int & numpel)` [private]

Lectura de un arbol binario de servidor.

Precondición

n es el numero de peliculas

Postcondición

cierto

Definición en la línea 24 del archivo Cjt_servidor.cpp.

```
{
for(int i = 0; i < servidores.size(); ++i) {
    servidores[i].leer_servidor(i,numpel);
}
}
```


4.3.3.3. void Cjt_servidor::marcaje (list< int > & l, int d, int idp) [private]

metodo para marcar los servidores seleccionados por la peticion el tiempo final y el id de la peticion mostrar por pantalla los servidores seleccionados. : l = L, d >= 0, 0 <= idp < mida conjunto peticion

Postcondición

cierto

Definición en la línea 68 del archivo Cjt_servidor.cpp.

```

{
//Pre: l = L d >= 0, 0 <= idp < mida conjunto peticion
if (not l.empty()) {
    list<int>::const_iterator it= l.begin();
    //Inv: it = iterador que recorre la lista desde inicio a fin.
    while (it != l.end()){
        cout << *it << " ";
        servidores[*it-1].setTiempo(d);
        servidores[*it-1].setIdpet(idp);
        ++it;
    }
    cout << endl;
}
//Post: Mostramos por pantalla la pelicula mas solicitada con t1 <= ti <= t2
}

```

4.3.3.4. bool Cjt_servidor::condicio1 (Arbre< int > & a, list< int > & l, Pelicula & pel, int & ancho, int & alt, int & prof) [private]

metodo para calcular el tiempo final en una unica unidad de tiempo

Precondición

Arbol inicializado, lista inicializada, pelicula inicializada, ancho para para pasar en la recursividad, altura para saber la altura maxima y profundida para saber la profundidad del servidor que servira la peticion

Postcondición

bool para indicar si a podido en una unidad de tiempo

Definición en la línea 85 del archivo Cjt_servidor.cpp.

```

{
//Pre: a = A l = L pel = Pelicula, ancho contine la suma de los anchos de banda de los nodos superiores
// alt indica en que altura estamos encada momento
// prof se utiliza para indicar al nodo padre la profundidad maxima del servidor que contiene la pelicula
bool b = false;
if (not a.es_buit() and ancho < pel.getTamano()){

```

```

    alt++;
    int valor = a.arrel();
    int n = pel.getId();
    //el servido visitado contiene la pelicula
    if(servidores[valor-1].contPelicula(n) and servidores[valor-1].getTiempo() == 0
    ) {
        ancho += servidores[valor-1].getAncho();
        l.insert(l.begin(),valor);
        prof = alt;
    }
    Arbre<int> a1,a2;
    a.fills(a1,a2);
    list<int> ret1,ret2;
    int ancho1,ancho2;
    ancho1 = ancho2 = ancho;
    int alt1,alt2,prof1,prof2;
    alt1 = alt2 = alt;
    prof1 = prof2 = prof;
    bool b1 = condiciol(a1,ret1,pel,ancho1,alt1,prof1);//ancho1 = ancho + (ancho
        de todo lo demas)
    bool b2 = condiciol(a2,ret2,pel,ancho2,alt2,prof2);//ancho2 = ancho + (ancho
        de todo lo demas)
    //HI: ancho1 indica el ancho de banda total de a1,ancho2 indica el ancho de b
        anda total de a2,
    // prof1 indica la profundidad maxima del servidor que contiene la pelicula d
        e a1,
    // prof2 indica la profundidad maxima del servidor que contiene la pelicula d
        e a2,
    a.plantar(valor,a1,a2);
    if(b1 and b2) {
        if(prof1 <= prof2) {
            ancho = ancho1;
            l.splice(l.end(),ret1);
            alt = alt1;
            prof = prof1;
        }
        else {
            ancho = ancho2;
            l.splice(l.end(),ret2);
            alt = alt2;
            prof = prof2;
        }
    }
    else if (b1){
        ancho = ancho1;
        l.splice(l.end(),ret1);
        alt = alt1;
        prof = prof1;
    }
    else if (b2){
        ancho = ancho2;
        l.splice(l.end(),ret2);
        alt = alt1;
        prof = prof1;
    }
    if(ancho >= pel.getTamano()) b = true;
    else b = false;
}
return b;
//Post: El resultado indica si la peticion se a podido servir en una unica unid
    ad de tiempo
}

```

4.3.3.5. void Cjt_servidor::condicio2 (Arbre< int > & a, list< int > & l, Pelicula & pel, int & ancho, int & alt, int & prof) [private]

metodo para calcular el tiempo final cojiendo el mejor ancho de banda segun los criterios

Precondición

Arbol inicializado, lista inicializada, pelicula inicializada, ancho para para pasar en la recursividad, altura para saber la altura maxima y profundida para saber la profundidad del servidor que servira la peticion

Postcondición

cierto

Definición en la línea 148 del archivo Cjt_servidor.cpp.

```

{
//Pre: a = A l = L pel = Pelicula, ancho contine la suma de los anchos de banda de los nodos superiores
// alt indica en que altura estamos encada momento
// prof se utiliza para indicar al nodo padre la profundidad maxima del servidor que contiene la pelicula
if (not a.es_buit()) {
    alt++;
    int valor = a.arrel();
    int n = pel.getId();
    //el servido visitado contiene la pelicula
    if(servidores[valor-1].contPelicula(n) and servidores[valor-1].getTiempo() == 0) {
        ancho += servidores[valor-1].getAncho();
        l.insert(l.begin(), valor);
        prof = alt;
    }
    Arbre<int> a1,a2;
    a.fill(a1,a2);
    list<int> ret1,ret2;
    int ancho1,ancho2;
    ancho1 = ancho2 = ancho;
    int alt1,alt2,prof1,prof2;
    alt1 = alt2 = alt;
    prof1 = prof2 = prof;
    condicio2(a1,ret1,pel,ancho1,alt1,prof1);//ancho1 = ancho + (ancho de todo l o demas)
    condicio2(a2,ret2,pel,ancho2,alt2,prof2);//ancho2 = ancho + (ancho de todo l o demas)
    //HI: ancho1 indica el ancho de banda total de a1, ancho2 indica el ancho de banda total de a2
    // prof1 indica la profundidad maxima del servidor que contiene la pelicula de a1
    // prof2 indica la profundidad maxima del servidor que contiene la pelicula de a2
    a.plantar(valor,a1,a2);
    if(ancho1 >= ancho2 ) {
        if(ancho1 == ancho2) {
            if(prof1 <= prof2) {
                ancho = ancho1;
                l.splice(l.end(),ret1);
            }
        }
    }
}

```

```

        alt = alt1;
        prof = prof1;
    }
    else {
        ancho = ancho2;
        l.splice(l.end(), ret2);
        alt = alt2;
        prof = prof2;
    }
    }
    else{
        ancho = ancho1;
        l.splice(l.end(), ret1);
        alt = alt1;
        prof = prof1;
    }
    }
    else {
        ancho = ancho2;
        l.splice(l.end(), ret2);
        alt = alt2;
        prof = prof2;
    }
    }
}

```

4.3.3.6. `int Cjt_servidor::roundUp (int a, int b)` [private]

calculo de la division redondeando hacia arriba

Precondición

Tamano de la pelicula, suma de anchos de banda

Postcondición

tiempo final de la peticion

Definición en la línea 208 del archivo Cjt_servidor.cpp.

```

{
    if (a%b == 0) return a/b;
    else return (a/b)+1;
}

```

4.3.3.7. `void Cjt_servidor::llegir_arbre_serv (int numpel)`

Lectura de la estructura del arbol y de los servidores.

Precondición

n = mida del [Cjt_pelicula](#)

Postcondición

cierto

Definición en la línea 30 del archivo Cjt_servidor.cpp.

```

{
    int numservidores = readint();
    servidores = vector<Servidor>(numservidores); //inicialitzem el vector de servidor
    s
    int numser = 0; //variable per contar el numero de servidor introduits en el ar
    bre
    llegir_estructura_arbre(estructura, 0, numser);
    llegir_serv(numser);
}

```

4.3.3.8. void Cjt_servidor::actualizar (int t)

actualizacion de los servidore si t > tiempo del servidor, tiempo = 0 y estado = false

Precondición

variable de tiempo

Postcondición

cierto

Definición en la línea 38 del archivo Cjt_servidor.cpp.

```

{
    for(int i = 0; i < servidores.size(); ++i) {
        servidores[i].actualizar(t);
    }
}

```

4.3.3.9. void Cjt_servidor::actualizar_servidor ()

actualizacion Alta/Baja de peliculas en el servidor

Precondición

cierto

Postcondición

cierto

Definición en la línea 58 del archivo Cjt_servidor.cpp.

```

{
    int x;
    x = readint();
    servidores[x-1].actualizar_pelicula();
}

```

4.3.3.10. void Cjt_servidor::calcular_tiempoFinal (Peticion & pet, Pelicula & pel)

calculo del tiempo final de la peticion(realiza los calculos apartir de las 3 condiciones)

Precondición

Peticion inicializada, Pelicula inicializada

Postcondición

cierto

Definición en la línea 213 del archivo Cjt_servidor.cpp.

```

list<int> ret;
int ancho,alt,tiempo,prof;
ancho = tiempo = 0;
alt = prof = -1;
bool b = condicio1(estructura,ret,pel,ancho,alt,prof);
if(not b) {
    ret.clear();
    ancho = tiempo = 0;
    alt = prof = -1;
    condicio2(estructura,ret,pel,ancho,alt,prof);
}
double d = 0;
cout << "Peticion procesada y servidores " << endl;
if (ancho != 0) {
    tiempo = roundUp(pel.getTamano(), ancho);
    p.setTf(tiempo);
    cout << p.getId() << " " << tiempo << endl;
    marcaje(ret,p.getT_f(),p.getId());
}
else cout << p.getId() << " " << tiempo << endl;
}
    
```

4.3.3.11. int Cjt_servidor::numeroServidores ()

consulta de numero de servidores

Precondición

cierto

Postcondición

cierto

Definición en la línea 64 del archivo Cjt_servidor.cpp.

```

    {
return servidores.size();
}
    
```

4.3.3.12. void Cjt_servidor::escriure_servidorOcupats (int t)

escritura de servidores ocupados en el tiempo t

Precondición

variable de tiempo

Postcondición

cierto

Definición en la línea 44 del archivo Cjt_servidor.cpp.

```

{
    bool b = false;
    cout << "Servidores ocupados" << endl;
    for(int i = 0; i < servidores.size(); ++i) {
        if (servidores[i].getTiempo() > t) {
            cout << servidores[i].getId() << " ";
            cout << servidores[i].getIdpet() << " ";
            cout << servidores[i].getTiempo() - t << endl;
            b = true;
        }
    }
    if (not b) cout << "0"<< endl;
}

```

4.3.4. Documentación de los datos miembro**4.3.4.1. Arbre<int> Cjt_servidor::estructura [private]**

Arbol que contiene la estructura de los servidores.

Definición en la línea 19 del archivo Cjt_servidor.hpp.

4.3.4.2. vector<Servidor> Cjt_servidor::servidores [private]

Vector de servidores.

Definición en la línea 21 del archivo Cjt_servidor.hpp.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- [Cjt_servidor.hpp](#)
- [Cjt_servidor.cpp](#)

4.4. Referencia de la Clase Pelicula

Representa una [Pelicula](#).

Métodos públicos

- `Pelicula ()`
Creadora por defecto. Se ejecuta automaticamente al declarar una `Pelicula`.
- `Pelicula (const Pelicula &p)`
Creadora copiadora.
- `~Pelicula ()`
Destructora.
- `void leer_pelicula (int &i)`
Lee los parametros de la pelicula.
- `int getId () const`
Consulta del ID de la pelicula.
- `int getTamano () const`
Consulta el tamano de la pelicula.

Atributos privados

- `int id`
Identificador de la pelicula.
- `int tam`
Tamano de la pelicula en mb.

4.4.1. Descripción detallada

Representa una `Pelicula`.

Definición en la línea 8 del archivo `Pelicula.hpp`.

4.4.2. Documentación del constructor y destructor

4.4.2.1. `Pelicula::Pelicula ()`

Creadora por defecto. Se ejecuta automaticamente al declarar una `Pelicula`.

Precondición

cierto

Postcondición

El resultado es una [Pelicula](#) sin parametros

Definición en la línea 3 del archivo Pelicula.cpp.

```
        {  
    id = -1;  
    tam = -1;  
}
```

4.4.2.2. Pelicula::Pelicula (const Pelicula & p)

Creadora copiadora.

Permite declarar una [Pelicula](#) nueva como copia de otra ya existente.

Precondición

cierto

Postcondición

El resultado es una peliucLa igual que p

Definición en la línea 8 del archivo Pelicula.cpp.

```
        {  
    id = p.getId();  
    tam = p.getTamano();  
}
```

4.4.2.3. Pelicula::~~Pelicula ()

Destructora.

Definición en la línea 13 del archivo Pelicula.cpp.

```
{ }
```

4.4.3. Documentación de las funciones miembro**4.4.3.1. void Pelicula::leer_pelicula (int & i)**

Lee los parametros de la pelicula.

Precondición

identificador asignado por el conjunto

Postcondición

cierto

Definición en la línea 15 del archivo Pelicula.cpp.

```
        {  
    id = i;  
    tam = readint();  
}
```

4.4.3.2. int Pelicula::getId () const

Consulta del ID de la pelicula.

Precondición

cierto

Postcondición

ID de la pelicular

Definición en la línea 20 del archivo Pelicula.cpp.

```
        {  
    return id;  
}
```

4.4.3.3. int Pelicula::getTamano () const

Consulta el tamano de la pelicula.

Precondición

cierto

Postcondición

tamano de la pelicula

Definición en la línea 24 del archivo Pelicula.cpp.

```
        {  
    return tam;  
}
```

4.4.4. Documentación de los datos miembro

4.4.4.1. `int Pelicula::id` [private]

Identificador de la pelicula.

Definición en la línea 13 del archivo Pelicula.hpp.

4.4.4.2. `int Pelicula::tam` [private]

Tamano de la pelicula en mb.

Definición en la línea 15 del archivo Pelicula.hpp.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- [Pelicula.hpp](#)
- [Pelicula.cpp](#)

4.5. Referencia de la Clase Peticion

Representa una [Peticion](#).

Métodos públicos

- [Peticion](#) ()
Creadora por defecto. Se ejecuta automaticamente al declarar una [Peticion](#).
- [Peticion](#) (int idPel, int tini)
Creadora con parametros.
- [Peticion](#) (const [Peticion](#) &p)
Creadora copiadora. Permite declarar una [Pelicula](#) nueva como copia de otra ya existente.
- [~Peticion](#) ()
Destructora.
- void [setTf](#) (int t)
Modifica el tiempo final de la [Peticion](#).
- void [setId](#) (int i)
Modifica el id de la [Peticion](#).
- int [getId](#) () const
Consulta del ID de la peticion.

- `int getT_i () const`
Consulta el tiempo de inicio de la peticion.
- `int getT_f () const`
Consulta el tiempo final de la peticion.
- `int getIdPelicula () const`
Consulta el ID de la pelicula de la peticion.

Atributos privados

- `int id`
Identificador de la peticion.
- `int t_i`
Tiempo inicial de la peticion.
- `int t_f`
Tiempo final de la peticion.
- `int idpelicula`
Identificador de la pelicula servida.

4.5.1. Descripción detallada

Representa una [Peticion](#).

Definición en la línea 9 del archivo Peticion.hpp.

4.5.2. Documentación del constructor y destructor

4.5.2.1. `Peticion::Peticion ()`

Creadora por defecto. Se ejecuta automaticamente al declarar una [Peticion](#).

Precondición

cierto

Postcondición

El resultado es una [Peticion](#) sin parametros

Definición en la línea 3 del archivo Peticion.cpp.

```
        {  
            id = -1;  
            t_i = -1;  
            t_f = -1;  
            idpelicula = -1;  
        }  
    }
```

4.5.2.2. Peticion::Peticion (int *idPel*, int *tini*)

Creadora con parametros.

Precondición

id de la pelicula, tiempo inicial

Postcondición

El resultado es una [Peticion](#)

Definición en la línea 10 del archivo Peticion.cpp.

```
        {  
            id = -1;  
            t_i = tini;  
            t_f = -1;  
            idpelicula = idPel;  
        }  
    }
```

4.5.2.3. Peticion::Peticion (const Peticion & *p*)

Creadora copiadora. Permite declarar una [Pelicula](#) nueva como copia de otra ya existente.

Precondición

cierto

Postcondición

El resultado es una [Peticion](#) igual que p

Definición en la línea 17 del archivo Peticion.cpp.

```
        {  
            id = p.getId();  
            t_i = p.getT_i();  
            t_f = p.getT_f();  
            idpelicula = p.getIdPelicula();  
        }  
    }
```

4.5.2.4. `Peticion::~~Peticion ()`

Destructora.

Definición en la línea 24 del archivo `Peticion.cpp`.

```
{ }
```

4.5.3. Documentación de las funciones miembro

4.5.3.1. `void Peticion::setTf (int t)`

Modifica el tiempo final de la [Peticion](#).

Precondición

peticion inicializada

Postcondición

cierto

Definición en la línea 26 del archivo `Peticion.cpp`.

```
    {  
        t_f = t + t_i;  
    }
```

4.5.3.2. `void Peticion::setId (int i)`

Modifica el id de la [Peticion](#).

Precondición

peticion inicializada

Postcondición

cierto

Definición en la línea 30 del archivo `Peticion.cpp`.

```
    {  
        id = i;  
    }
```

4.5.3.3. int Peticion::getId () const

Consulta del ID de la peticion.

Precondición

cierto

Postcondición

ID de la peticion

Definición en la línea 34 del archivo Peticion.cpp.

```
        {  
    return id;  
}
```

4.5.3.4. int Peticion::getTi () const

Consulta el tiempo de inicio de la peticion.

Precondición

cierto

Postcondición

tiempo de inicio de la peticion

Definición en la línea 38 del archivo Peticion.cpp.

```
        {  
    return t_i;  
}
```

4.5.3.5. int Peticion::getTf () const

Consulta el tiempo final de la peticion.

Precondición

cierto

Postcondición

tiempo de final de la peticion

Definición en la línea 42 del archivo Peticion.cpp.

```
        {  
    return t_f;  
}
```

4.5.3.6. `int Peticion::getIdPelicula () const`

Consulta el ID de la pelicula de la peticion.

Precondición

cierto

Postcondición

ID de la pelicula de la peticion

Definición en la línea 46 del archivo Peticion.cpp.

```
        {  
    return idpelicula;  
}
```

4.5.4. Documentación de los datos miembro

4.5.4.1. `int Peticion::id [private]`

Identificador de la peticion.

Definición en la línea 14 del archivo Peticion.hpp.

4.5.4.2. `int Peticion::t_i [private]`

Tiempo inicial de la peticion.

Definición en la línea 16 del archivo Peticion.hpp.

4.5.4.3. `int Peticion::t_f [private]`

Tiempo final de la peticion.

Definición en la línea 18 del archivo Peticion.hpp.

4.5.4.4. `int Peticion::idpelicula [private]`

Identificador de la pelicula servida.

Definición en la línea 20 del archivo Peticion.hpp.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

- [Peticion.hpp](#)
- [Peticion.cpp](#)

4.6. Referencia de la Clase Servidor

Representa un [Servidor](#).

Métodos públicos

- [Servidor](#) ()
Creadora por defecto. Se ejecuta automaticamente al declarar un servidor.
- [Servidor](#) (const [Servidor](#) &s)
Creadora copiadora.
- [~Servidor](#) ()
Destructora.
- void [leer_servidor](#) (int i, int n)
Lee los parametros del servidor.
- void [actualizar](#) (int t)
Actualiza el servidor.
- void [actualizar_pelicula](#) ()
Actualiza el Pelicules de servidor.
- void [setTiempo](#) (int tf)
Modifica el tiempo del servidor.
- void [setIdpet](#) (int idp)
Modifica el id de la peticion que esta sirviendo.
- void [addPeliculas](#) (int n)
Anadir peliculas al servidor.
- void [deletePeliculas](#) (int n)
Eliminar una pelicula del servidor.
- int [getId](#) () const
Consulta del ID del servidor.
- int [getTiempo](#) () const
Consulta el tiempo del servidor.
- int [getAncho](#) () const
Consulta el ancho de banda del servidor.
- int [getIdpet](#) () const

Consulta el id de la petición del servidor, si esta sirviendo una si no esta sirviendo una petición Idpet = -1.

- `vector< bool > getPeliculas () const`

Consulta las películas del servidor(para hacer la copia del servidor)

- `bool contPelicula (int &id) const`

Consulta la película con el id = id.

Atributos privados

- `int id`

Identificador del servidor.

- `int ancho`

Ancho de banda del servidor en mb.

- `int tiempo`

tiempo en el que el servidor finaliza de servir una petición

- `int idpet`

Identificador de la petición sirviendo en el servidor.

- `vector< bool > lpel`

Vector de películas que contiene el servidor.

4.6.1. Descripción detallada

Representa un [Servidor](#).

Definición en la línea 9 del archivo Servidor.hpp.

4.6.2. Documentación del constructor y destructor

4.6.2.1. `Servidor::Servidor ()`

Creadora por defecto. Se ejecuta automáticamente al declarar un servidor.

Precondición

cierto

Postcondición

El resultado es un servidor sin valor

Definición en la línea 3 del archivo Servidor.cpp.

```
    {  
        tiempo = 0;  
        ancho = 0;  
        idpet = -1;  
    }
```

4.6.2.2. Servidor::Servidor (const Servidor & s)

Creadora copiadora.

Precondición

cierto

Postcondición

El resultado es servidor igual que c

Definición en la línea 8 del archivo Servidor.cpp.

```
    {  
        id = s.getId();  
        ancho = s.getAncho();  
        tiempo = s.getTiempo();  
        lpel = s.getPeliculas();  
        idpet = s.getIdpet();  
    }
```

4.6.2.3. Servidor::~~Servidor ()

Destructora.

Definición en la línea 15 del archivo Servidor.cpp.

```
{ }
```

4.6.3. Documentación de las funciones miembro

4.6.3.1. void Servidor::leer_servidor (int i, int n)

Lee los parametros del servidor.

Precondición

id del servidor, n = mida del Cjt de peliculas

Postcondición

Definición en la línea 16 del archivo Servidor.cpp.

```

{
    id = i+1;
    lpel = vector<bool>(n, false);
    int x, y;
    x = readint();
    ancho = x;
    x = readint();
    addPeliculas(x);
}
```

4.6.3.2. void Servidor::actualizar (int t)

Actualiza el servidor.

Precondición

tiempo a comparar

Postcondición

cierto

Definición en la línea 25 del archivo Servidor.cpp.

```

{
    if(t>= tiempo) {
        tiempo = 0;
        idpet = -1;
    }
}
```

4.6.3.3. void Servidor::actualizar_pelicula ()

Actualiza el Pelicules de servidor.

Precondición

cierto

Postcondición

cierto

Definición en la línea 31 del archivo Servidor.cpp.

```

{
    int x;
    x = readint();
    addPeliculas(x);
    x = readint();
    deletePeliculas(x);
}
```

4.6.3.4. void Servidor::setTiempo (int *tf*)

Modifica el tiempo del servidor.

Precondición

tiempo del servidor

Postcondición

cierto

Definición en la línea 39 del archivo Servidor.cpp.

```
        {  
    tiempo = t;  
}
```

4.6.3.5. void Servidor::setIdpet (int *idp*)

Modifica el id de la peticion que esta sirviendo.

Precondición

id de la peticion, -1 si no esta sirviendo ninguna

Postcondición

cierto

Definición en la línea 43 del archivo Servidor.cpp.

```
        {  
    idpet = idp;  
}
```

4.6.3.6. void Servidor::addPeliculas (int *n*)

Añadir peliculas al servidor.

Precondición

n es el numero de pelicules a añadir

Postcondición

cierto

Definición en la línea 69 del archivo Servidor.cpp.

```

{
    int x;
    for(int i = 0; i < n; ++i) {
        x = readint();
        lpel[x-1] = true;
    }
}
```

4.6.3.7. void Servidor::deletePeliculas (int n)

Eliminar una pelicula del servidor.

Precondición

n es el numero de pelicules a eliminar

Postcondición

cierto

Definición en la línea 76 del archivo Servidor.cpp.

```

{
    int x;
    for(int i = 0; i < n; ++i) {
        x = readint();
        lpel[x-1] = false;
    }
}
```

4.6.3.8. int Servidor::getId () const

Consulta del ID del servidor.

Precondición

cierto

Postcondición

Id del servidor

Definición en la línea 47 del archivo Servidor.cpp.

```

{
    return id;
}
```

4.6.3.9. int Servidor::getTiempo () const

Consulta el tiempo del servidor.

Precondición

cierto

Postcondición

tiempo del servidor

Definición en la línea 50 del archivo Servidor.cpp.

```
        {  
    return tiempo;  
}
```

4.6.3.10. int Servidor::getAncho () const

Consulta el ancho de banda del servidor.

Precondición

cierto

Postcondición

ancho de banda del servidor

Definición en la línea 53 del archivo Servidor.cpp.

```
        {  
    return ancho;  
}
```

4.6.3.11. int Servidor::getIdpet () const

Consulta el id de la peticion del servidor, si esta sirviendo una si no esta sirviendo una peticion Idpet = -1.

Precondición

cierto

Postcondición

id de la peticion

Definición en la línea 57 del archivo Servidor.cpp.

```
        {  
    return idpet;  
}
```

4.6.3.12. vector< bool > Servidor::getPelículas () const

Consulta las películas del servidor(para hacer la copia del servidor)

Precondición

cierto

Postcondición

ancho de banda del servidor

Definición en la línea 61 del archivo Servidor.cpp.

```
        {  
    return lpel;  
}
```

4.6.3.13. bool Servidor::contPelícula (int & id) const

Consulta la película con el id = id.

Precondición

$1 < id \leq lpel.size()$

Postcondición

true si contiene la película, false si no la contiene

Definición en la línea 65 del archivo Servidor.cpp.

```
        {  
    return lpel[id];  
}
```

4.6.4. Documentación de los datos miembro**4.6.4.1. int Servidor::id [private]**

Identificador del servidor.

Definición en la línea 14 del archivo Servidor.hpp.

4.6.4.2. int Servidor::ancho [private]

Ancho de banda del servidor en mb.

Definición en la línea 16 del archivo Servidor.hpp.

4.6.4.3. int Servidor::tiempo [private]

tiempo en el que el servidor finaliza de servir una peticion

Definición en la línea 18 del archivo Servidor.hpp.

4.6.4.4. int Servidor::idpet [private]

Identificador de la peticion sirviendo en el servidor.

Definición en la línea 20 del archivo Servidor.hpp.

4.6.4.5. vector<bool> Servidor::lpel [private]

Vector de peliculas que contiene el servidor.

Definición en la línea 22 del archivo Servidor.hpp.

La documentación para esta clase fue generada a partir de los siguientes ficheros:

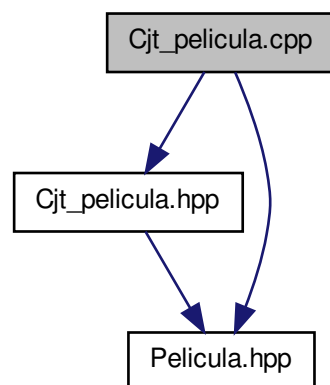
- [Servidor.hpp](#)
- [Servidor.cpp](#)

Capítulo 5

Documentación de archivos

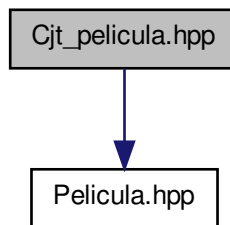
5.1. Referencia del Archivo Cjt_película.cpp

Dependencia gráfica adjunta para Cjt_película.cpp:



5.2. Referencia del Archivo Cjt_película.hpp

Dependencia gráfica adjunta para Cjt_película.hpp:



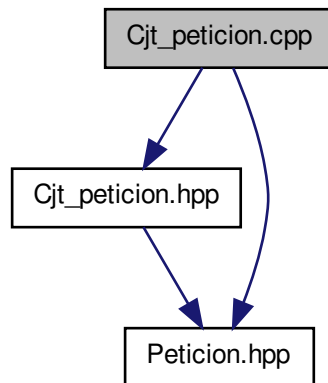
Clases

- class [Cjt_película](#)

Representa un conjunto de [Película](#).

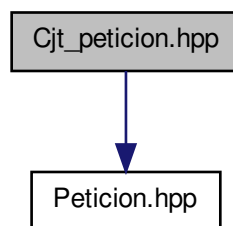
5.3. Referencia del Archivo Cjt_peticion.cpp

Dependencia gráfica adjunta para Cjt_peticion.cpp:



5.4. Referencia del Archivo Cjt_peticion.hpp

Dependencia gráfica adjunta para Cjt_peticion.hpp:



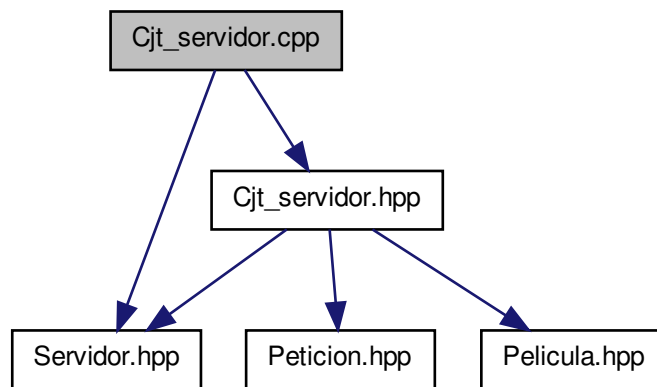
Clases

- class [Cjt_peticion](#)

Representa un conjunto de [Petición](#).

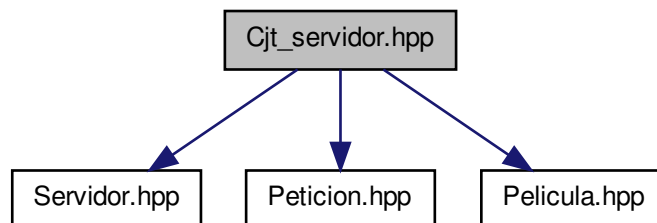
5.5. Referencia del Archivo Cjt_servidor.cpp

Dependencia gráfica adjunta para Cjt_servidor.cpp:



5.6. Referencia del Archivo Cjt_servidor.hpp

Dependencia gráfica adjunta para Cjt_servidor.hpp:



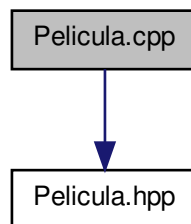
Clases

- class [Cjt_servidor](#)

Representa un conjunto de [Servidor](#).

5.7. Referencia del Archivo Pelicula.cpp

Dependencia gráfica adjunta para Pelicula.cpp:



5.8. Referencia del Archivo Pelicula.hpp

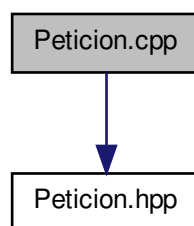
Clases

- class [Pelicula](#)

Representa una [Pelicula](#).

5.9. Referencia del Archivo Peticion.cpp

Dependencia gráfica adjunta para Peticion.cpp:



5.10. Referencia del Archivo Peticion.hpp

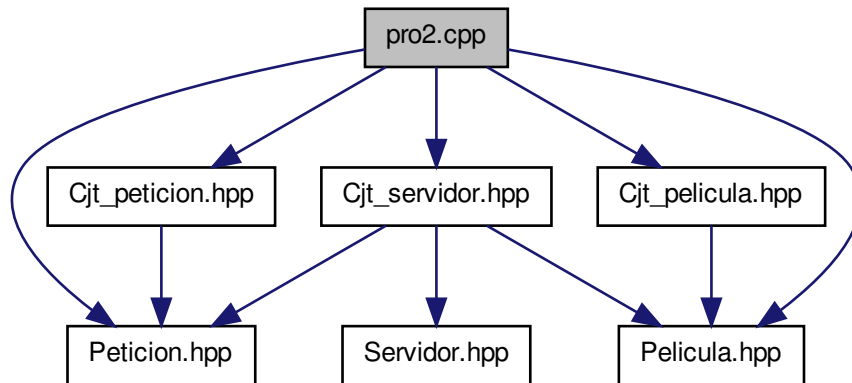
Clases

- class [Peticion](#)

Representa una [Peticion](#).

5.11. Referencia del Archivo pro2.cpp

Dependencia gráfica adjunta para pro2.cpp:



Funciones

- int `main()`

5.11.1. Documentación de las funciones

5.11.1.1. int main ()

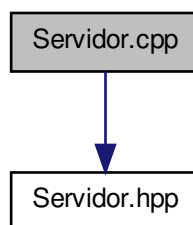
Definición en la línea 13 del archivo pro2.cpp.

```
{
    int n;
    Cjt_pelicula peliculas;
    peliculas.leer_peliculas();
    Cjt_servidor servers;
    servers.llegir_arbre_serv(peliculas.mida());
    Cjt_peticion peticiones;
    n = readint();
    while(n != -6 ) {
        if(n == -1) {
            int t, peli;
            peli = readint();
            t = readint();
            servers.actualizar(t);
            Peticion p(peli, t);
            peticiones.incrementId(p);
        }
    }
}
```

```
Pelicula pel = peliculas.getPeliculaId(peli);
servers.calcular_tiempoFinal(p,pel);
if (p.getT_f() != -1) peticiones.anadir_peticion(p);
}
if(n == -2) {
    int t;
    t = readint();
    servers.actualizar(t);
    peticiones.peticion_sinFinalizar(t);
}
if(n == -3) {
    int t;
    t = readint();
    servers.actualizar(t);
    servers.escriure_servidorOcupats(t);
}
if(n == -4) {
    servers.actualizar_servidor();
}
if(n == -5) {
    int t1,t2;
    t1 = readint();
    t2 = readint();
    peticiones.pelicula_solicitada(t1,t2,peliculas.mida());
}
n = readint();
}
}
```

5.12. Referencia del Archivo Servidor.cpp

Dependencia gráfica adjunta para Servidor.cpp:



5.13. Referencia del Archivo Servidor.hpp

Clases

- class [Servidor](#)
Representa un [Servidor](#).

Índice alfabético

- ~Cjt_pelicula
 - Cjt_pelicula, 8
- ~Cjt_peticion
 - Cjt_peticion, 12
- ~Cjt_servidor
 - Cjt_servidor, 17
- ~Pelicula
 - Pelicula, 27
- ~Petition
 - Petition, 31
- ~Servidor
 - Servidor, 37
- actualizar
 - Cjt_servidor, 23
 - Servidor, 38
- actualizar_pelicula
 - Servidor, 38
- actualizar_servidor
 - Cjt_servidor, 23
- addPeliculas
 - Servidor, 39
- anadir_peticion
 - Cjt_peticion, 12
- ancho
 - Servidor, 42
- calcular_tiempoFinal
 - Cjt_servidor, 23
- Cjt_pelicula, 7
 - ~Cjt_pelicula, 8
 - Cjt_pelicula, 8
 - Cjt_pelicula, 8
 - escribir_peliculas, 9
 - getPeliculaId, 9
 - leer_peliculas, 8
 - mida, 9
 - vecpel, 10
- Cjt_pelicula.cpp, 45
- Cjt_pelicula.hpp, 46
- Cjt_peticion, 10
 - ~Cjt_peticion, 12
 - anadir_peticion, 12
 - Cjt_peticion, 11
 - Cjt_peticion, 11
 - cont, 15
 - incrementId, 13
 - listpet, 15
 - mida, 13
 - pelicula_solicitada, 14
 - peticion_sinFinalizar, 13
 - pushList, 12
- Cjt_peticion.cpp, 47
- Cjt_peticion.hpp, 47
- Cjt_servidor, 15
 - ~Cjt_servidor, 17
 - actualizar, 23
 - actualizar_servidor, 23
 - calcular_tiempoFinal, 23
 - Cjt_servidor, 17
 - Cjt_servidor, 17
 - condicio1, 19
 - condicio2, 20
 - escriure_servidorOcupats, 24
 - estructura, 25
 - llegir_arbre_serv, 22
 - llegir_estructura_arbre, 18
 - llegir_serv, 18
 - marcaje, 18
 - numeroServidores, 24
 - roundUp, 22
 - servidores, 25
- Cjt_servidor.cpp, 48
- Cjt_servidor.hpp, 48
- condicio1
 - Cjt_servidor, 19
- condicio2
 - Cjt_servidor, 20
- cont
 - Cjt_peticion, 15
- contPelicula

- Servidor, 42
- deletePelículas
 - Servidor, 40
- escribir_peliculas
 - Cjt_pelicula, 9
- escriure_servidorOcupats
 - Cjt_servidor, 24
- estructura
 - Cjt_servidor, 25
- getAncho
 - Servidor, 41
- getId
 - Pelicula, 28
 - Peticion, 32
 - Servidor, 40
- getIdPelicula
 - Peticion, 33
- getIdpet
 - Servidor, 41
- getPeliculaId
 - Cjt_pelicula, 9
- getPelículas
 - Servidor, 41
- getT_f
 - Peticion, 33
- getT_i
 - Peticion, 33
- getTamano
 - Pelicula, 28
- getTiempo
 - Servidor, 40
- id
 - Pelicula, 29
 - Peticion, 34
 - Servidor, 42
- idpelicula
 - Peticion, 34
- idpet
 - Servidor, 43
- incrementId
 - Cjt_peticion, 13
- leer_pelicula
 - Pelicula, 27
- leer_peliculas
 - Cjt_pelicula, 8
- leer_servidor
 - Servidor, 37
- listpet
 - Cjt_peticion, 15
- llegir_arbre_serv
 - Cjt_servidor, 22
- llegir_estructura_arbre
 - Cjt_servidor, 18
- llegir_serv
 - Cjt_servidor, 18
- lpel
 - Servidor, 43
- main
 - pro2.cpp, 51
- marcaje
 - Cjt_servidor, 18
- mida
 - Cjt_pelicula, 9
 - Cjt_peticion, 13
- numeroServidores
 - Cjt_servidor, 24
- Pelicula, 25
 - ~Pelicula, 27
 - getId, 28
 - getTamano, 28
 - id, 29
 - leer_pelicula, 27
 - Pelicula, 26, 27
 - tam, 29
- Pelicula.cpp, 49
- Pelicula.hpp, 49
- pelicula_solicitada
 - Cjt_peticion, 14
- Peticion, 29
 - ~Peticion, 31
 - getId, 32
 - getIdPelicula, 33
 - getT_f, 33
 - getT_i, 33
 - id, 34
 - idpelicula, 34
 - Peticion, 30, 31
 - setId, 32
 - setTf, 32
 - t_f, 34
 - t_i, 34
- Peticion.cpp, 50
- Peticion.hpp, 50

peticion_sinFinalizar
 Cjt_peticion, [13](#)

pro2.cpp, [51](#)
 main, [51](#)

pushList
 Cjt_peticion, [12](#)

roundUp
 Cjt_servidor, [22](#)

Servidor, [35](#)
 ~Servidor, [37](#)
 actualizar, [38](#)
 actualizar_pelicula, [38](#)
 addPeliculas, [39](#)
 ancho, [42](#)
 contPelicula, [42](#)
 deletePeliculas, [40](#)
 getAncho, [41](#)
 getId, [40](#)
 getIdpet, [41](#)
 getPeliculas, [41](#)
 getTiempo, [40](#)
 id, [42](#)
 idpet, [43](#)
 leer_servidor, [37](#)
 lpel, [43](#)
 Servidor, [36](#), [37](#)
 setIdpet, [39](#)
 setTiempo, [38](#)
 tiempo, [42](#)

Servidor.cpp, [52](#)

Servidor.hpp, [53](#)

servidores
 Cjt_servidor, [25](#)

setId
 Peticion, [32](#)

setIdpet
 Servidor, [39](#)

setTf
 Peticion, [32](#)

setTiempo
 Servidor, [38](#)

t_f
 Peticion, [34](#)

t_i
 Peticion, [34](#)

tam
 Pelicula, [29](#)

tiempo
 Servidor, [42](#)

vecpel
 Cjt_pelicula, [10](#)