Universidad Nacional de Educación a Distancia

Grado de Ingeniería Informática

# Sistema básico de almacenamiento en la nube usando Java RMI

Memoria presentada por Roberto Castellor Morant
para la asignatura Sistemas Distribuidos de Grado de Ingeniería Informática

DNI: 18041663A
Correo electrónico: roberto.castellor@gmail.com

2017

Departamento de Sistemas de Comunicación y Control

# Índice general

# Capítulo 1

# Introducción

Este documento contiene la memoria de la práctica obligatoria de laboratorio correspondiente a la asignatura de *Sistemas Distribuidos* del Grado de Ingeniería Informática. El propósito de la practica es el desarrollo de un software que implemente un sistema de almacenamiento de ficheros en la nube usando Java RMI.

El aplicativo consiste en tres ejecutables diferentes, el servidor levanta tres servicios, dos de ellos serán usados por los distintos usuarios del servicio distribuido mientras que el tercer servicio sera para uso interno de los servicios propios del servidor.

El repositorio publica dos objetos distribuidos para uso de los clientes o del servidor, estos objetos ofrecen los métodos necesarios para almacenar los ficheros en su sistema local de ficheros o enviarlos al sistema local de ficheros de los clientes.

El cliente sera la parte utilizada por el usuario para gestionar sus ficheros, publicara a su vez un objeto distribuido que se utilizara para almacenar ficheros en el disco local del cliente.

La interfaz de la aplicación se implementa mediante un menú en modo texto, los resultados de las operaciones se volcaran sobre la salida estándar. De cara a una depuración más eficiente del sistema se generará un fichero de log llamado "log.txt" en la carpeta donde se ejecuten los procesos donde se podrá encontrar información adicional para depurar la aplicación.

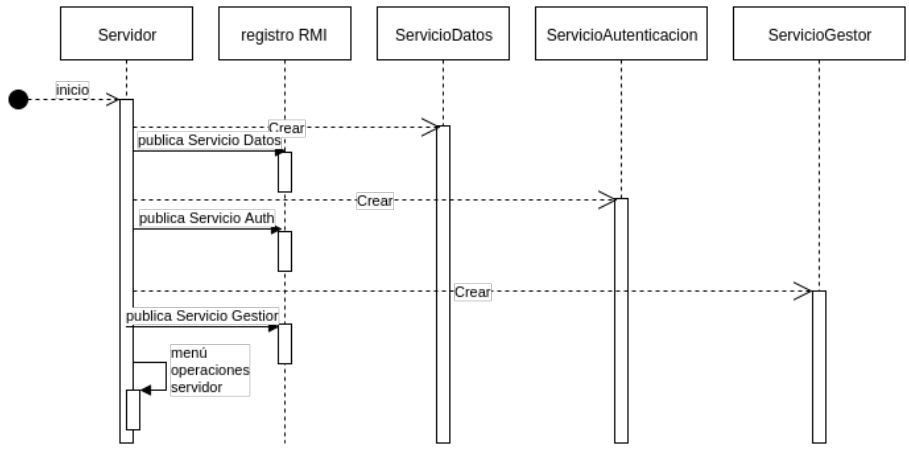
# Capítulo 2

# Casos de uso

En este capitulo repasamos los principales casos de uso de la aplicación, caben analizar muchos más escenarios y en sucesivas iteraciones de desarrollo se podría ampliar el numero de casos de uso así como logar un mayor nivel de detalle de los expuestos en el documento.

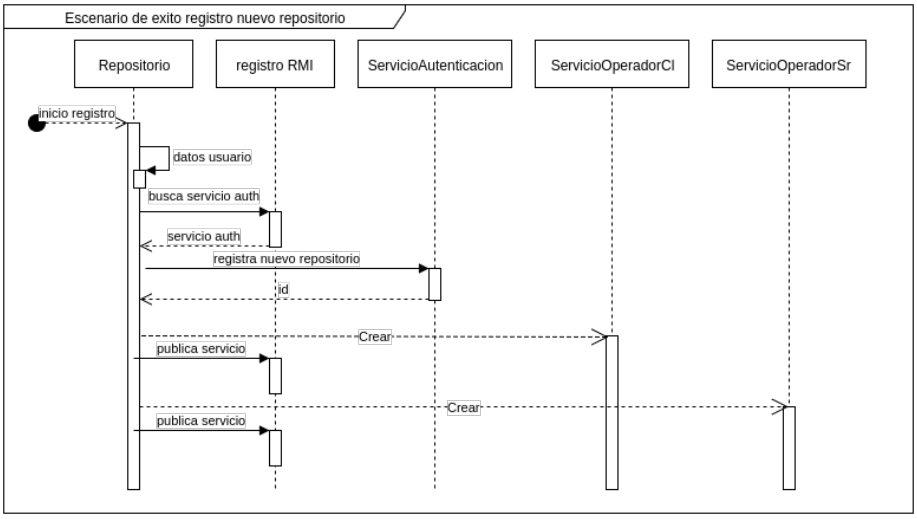
## 2.1. Inicio del servidor

1. El caso de uso comienza cuando el usuario lanza el proceso servidor.
2. El sistema levanta el registro para almacenar los objetos distribuidos
3. El sistema crea el objeto para el servicio de datos
4. El sistema publica en el registro RMI el servicio de datos
5. El sistema crea el objeto para el servicio de autenticación
6. El sistema publica en el registro RMI el servicio de autenticación
7. El sistema crea el objeto para el servicio gestor
8. El sistema publica en el registro RMI el servicio gestor
9. El caso de uso esta completo, el sistema entra en modo escucha de comandos



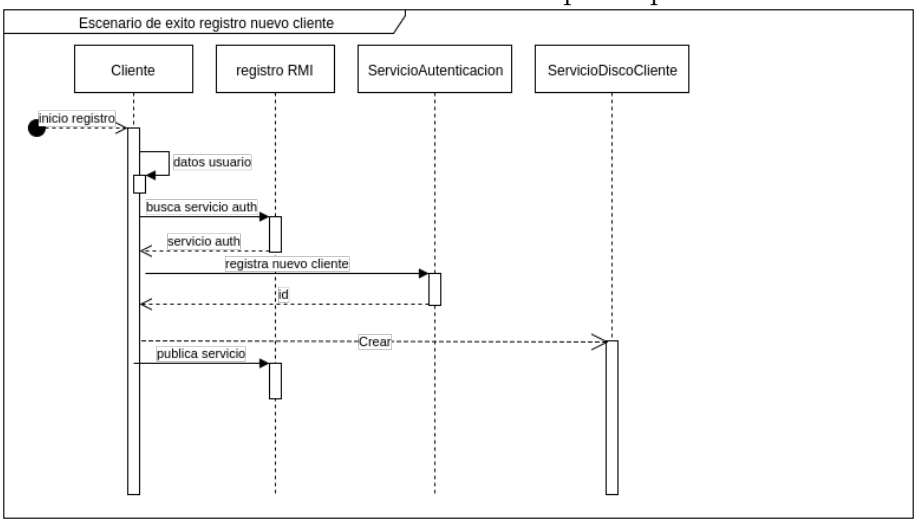

## 2.2. Registrar/autenticar repositorio

1. El caso de uso comienza cuando el usuario del repositorio selecciona la opción de registrar repositorio
2. El sistema solicita al usuario los datos de autenticación
3. El sistema busca el servicio de autenticación en el registro RMI
4. El sistema comprueba los parámetros de entrada con el objeto remoto
5. El sistema publica el servicio operador cliente
6. El sistema publica el servicio operador servidor
7. El sistema entra en modo consulta para operaciones con el repositorio activo



## 2.3. Registrar/autenticar cliente

1. El caso de uso comienza cuando el usuario del cliente selecciona la opción de registrar cliente
2. El sistema solicita al usuario los datos de autenticación
3. El sistema busca el servicio de autenticación en el registro RMI
4. El sistema comprueba los parámetros de entrada con el objeto remoto
5. El sistema publica los objetos remotos para operar con el servidor en el registro
6. El sistema entra en modo consulta para operaciones con el cliente activo

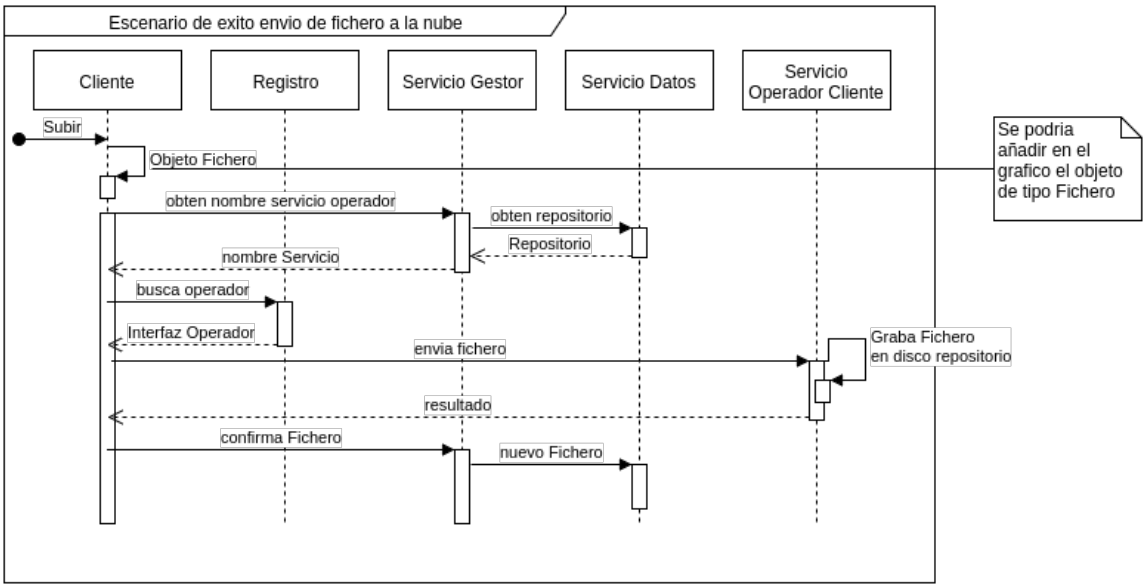



## 2.4. Subir fichero a la nube

1. El caso de uso comienza cuando el usuario selecciona la opción subir fichero
2. El sistema pregunta al usuario cual es el fichero que quiere subir a la nube
3. El usuario selecciona el nombre del fichero
4. El sistema crea un objeto serializable en memoria con el fichero a enviar
5. El sistema busca el objeto distribuido gestor
6. El sistema consulta al servicio gestor por la dirección del servicio operador cliente del repositorio
7. El sistema gestor consulta al servicio de datos por el id del repositorio
8. El sistema busca el objeto distribuido cliente operador del repositorio
9. El sistema envía el objeto de tipo Fichero Serializable al repositorio mediante

el servicio operador cliente

10. El servicio operador-cliente guarda el fichero en disco con el método escribirEn del objeto Fichero
11. El sistema notifica al sistema gestor el resultado del envío
12. El servicio gestor actualiza el servicio de datos con el nuevo fichero
13. El sistema notifica al usuario el resultado de la operación

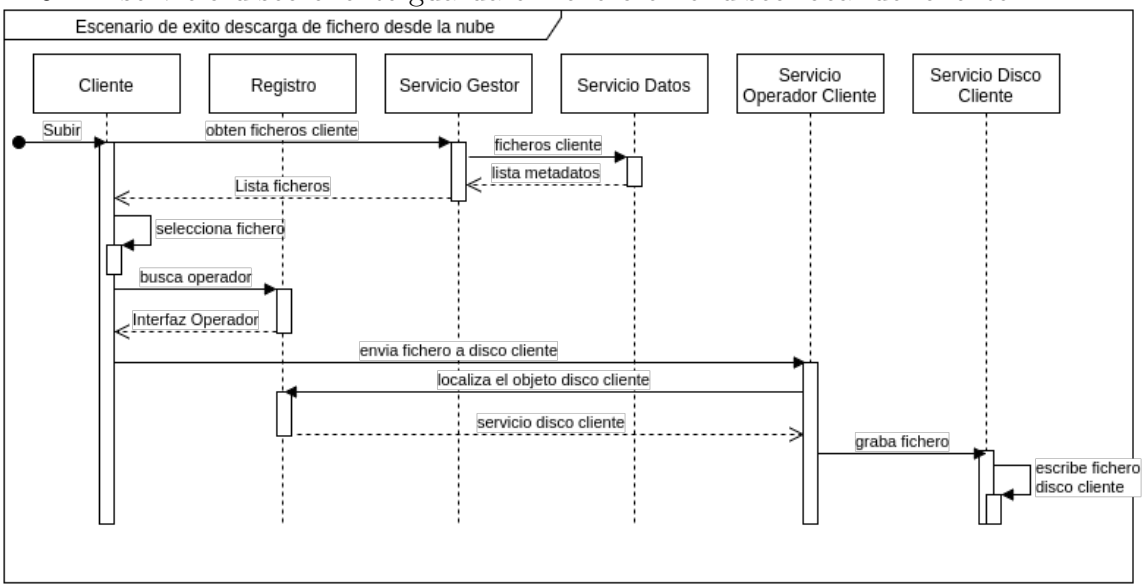



## 2.5. Descargar fichero desde la nube

1. El caso de uso comienza cuando el usuario selecciona la opción de descargar fichero
2. El sistema pregunta al servicio gestor por los ficheros a los que tiene acceso el cliente
3. El sistema pregunta al usuario cual de los ficheros quiere descargar
4. El usuario selecciona un fichero de la lista
5. El sistema busca en el registro el objeto operador cliente del repositorio
6. El sistema pide al objeto operador cliente que envíe el fichero mediante el servicio disco cliente
7. El servicio operador busca el servicio disco cliente
8. El servicio operador cliente envía el fichero a través del método del servicio

disco cliente

9. El servicio disco cliente guarda el fichero en el disco local del cliente




## 2.6. Borrar fichero de la nube

1. El caso de uso comienza cuando el usuario selección la opción de borrar fichero
2. El sistema pregunta al servicio gestor por los ficheros del usuario
3. El sistema pregunta al usuario cual es el fichero que quiere borrar
4. El sistema pide al servicio gestor la url del objeto distribuido del servicio operador-cliente de su repositorio
5. El sistema busca en el registro RMI el servicio operador-cliente del repositorio donde esta el fichero
6. El sistema pide al servicio operador-cliente que borre el fichero
7. El sistema notifica al servicio gestor del resultado de la operación de borrado
8. El servicio gestor notifica al servicio de datos que el fichero se ha borrado fisícamente

# Capítulo 3

# Consideraciones de diseño

## 3.1. Estructura de directorios

Las Carpeta comun contiene las clases e interfaces comunes al resto del proyecto.

- **Cliente**, clase con la información de los clientes, el usuario, el id y lista con los ficheros del cliente
- **Fichero**, clase proporcionada por el equipo docente para enviar los ficheros
- **GUI**, clase con métodos genéricos para mostrar por pantalla información y recoger las entradas del usuario
- **Logger**, clase con métodos para volcar a un fichero log de la aplicación y de esta forma no mezclarse con la interfaz del usuario
- **Metadato**, clase con información referente a los ficheros, tendrá el repositorio, el id del cliente propietario del fichero, etc.
- **Repositorio**, clase con datos de los repositorios registrados en el sistema, los id, el usuario y las url de los objetos remotos que publica el repositorio.
- *ServicioAutenticacionInterface*, interfaz del servicio de autenticación.
- *ServicioDatosInterface*, interfaz del servicio de datos
- *ServicioGestorInterface*, interfaz del servicio gestor
- *ServicioClOperadorInterface*, interfaz del servicio cliente-operador
- *ServicioSrOperadorInterface*, interfaz del servicio servidor-operador
- *ServicioDiscoClienteInterface*, interfaz del servicio de disco del cliente
- **Utils**, clase con métodos genéricos para el uso por parte de la aplicación

La carpeta servidor, contiene el código necesario para lanzar el servidor y la implementación del servicio de datos, el servicio de autenticación y el servicio gestor.

- **Servidor**, clase que lanza el servidor, crea los objetos remotos, los publica en el registro RMI y genera el menu para operar con el servidor
- **ServicioAutenticacionImpl**, clase que implementa el servicio de autenticación
- **ServicioDatosImpl**, clase que implementa el servicio de datos
- **ServicioGestorImpl**, clase que implementa el servicio gestor

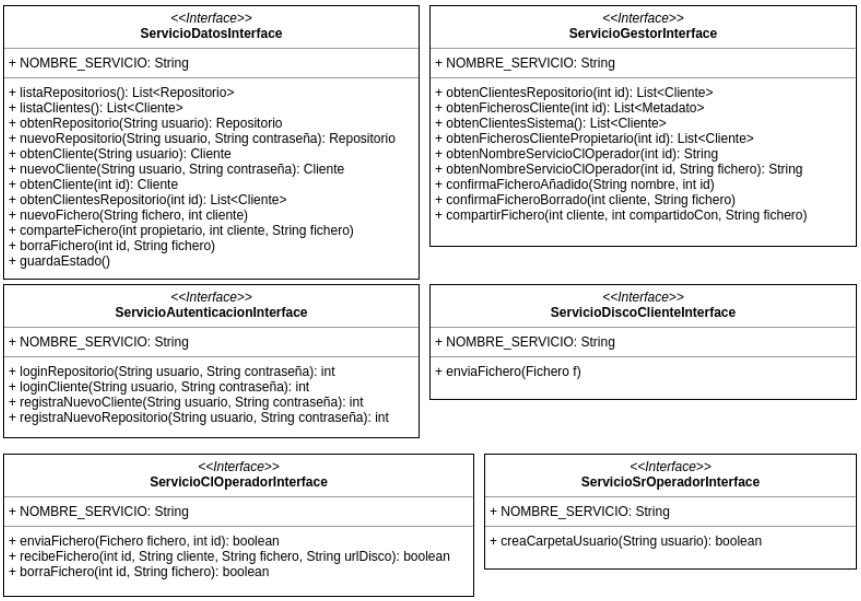
La carpeta repositorio contiene el código necesario para lanzar los repositorios y la implementación de los servicios operador-cliente y operador-servidor.

- **Repositorio**, clase que lanza el repositorio y genera el menú para operar con el y publica los objetos remotos.
- **ServicioClOperadorImpl**, clase que implementa el servicio cliente-operador
- **ServicioSrOperadorImpl**, clase que implementa el servicio servidor-operador

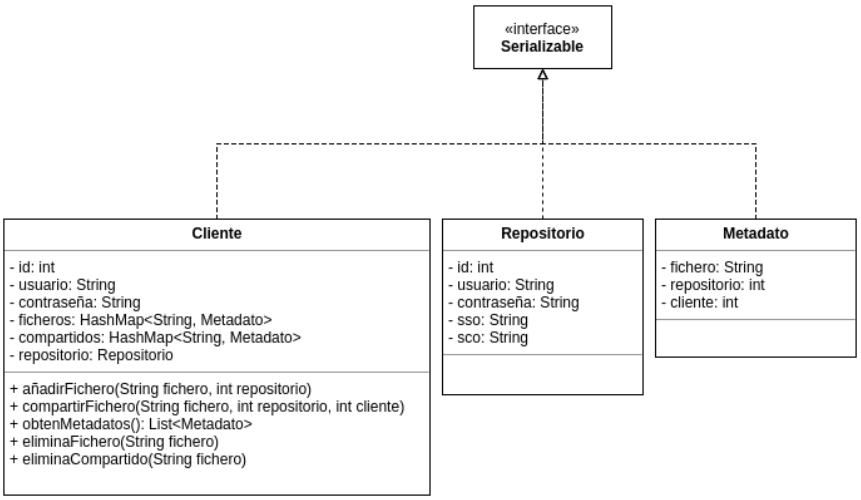
La carpeta cliente contiene el código necesario para lanzar los clientes y la implementación de la interfaz remota del servicio de disco del cliente

- **Cliente**, clase que lanza el cliente, genera el menú para poder operar con el y publica los objetos remotos
- **ServicioDiscoClienteImpl**, clase que implementa el servicio de disco del cliente a través del cual se recibirán ficheros de la nube

## 3.2. Interfaces remotas




## 3.3. Objetos modelo de datos




El modelo de datos esta basado en la api Collections de Java.

Cada elemento del sistema sera identificado por un id numérico que formara parte del objeto como atributo.

En el servicio de datos se almacenara la siguiente información:

- Lista con todos los clientes registrados en la aplicación.

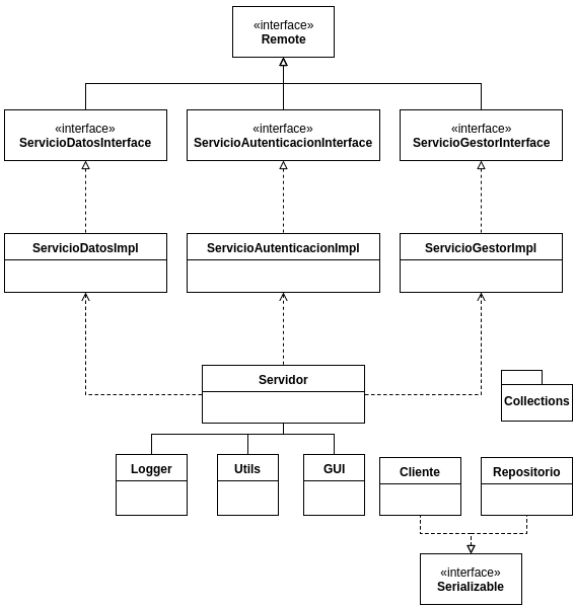
- Relación de nombre de usuario con id de clientes en una HashMap<String, Integer>
- Relación de id de usuario con el objeto de tipo Cliente que tiene toda la información referente a cada cliente.
- Variable sesioncli con el ultimo identificador de cliente asignado, a cada cliente nuevo se le asignara como identificador el valor de esta variable y esta se incrementara en espera del próximo cliente.
- Lista con todos los repositorios registrados en la aplicación.
- Relación de nombre de usuario con id de repositorio en una HashMap<String, Integer>
- Relación de id de usuario con el objeto de tipo Repositorio que tiene toda la información referente a cada cliente.
- Variable sesionrep con el ultimo identificador de repositorio asignado, a cada repositorio nuevo se le asignara como identificador el valor de esta variable y esta se incrementara en espera del próximo cliente.
- Variable numerica que se utilizara para decidir a que repositorio asignar cada nuevo cliente

La clase Cliente del paquete comun tiene la información propia del cliente y un mapa de los metadatos indexado por el nombre del fichero. Las operaciones con los datos de los ficheros se realizaran mediante llamadas a metodos de esta clase y en ningún momento accediendo directamente a las collection que albergan los datos, (ver apartado B.1)

Esta información se persistirá a disco cuando se cierre el sistema para guardar el estado del servidor.

## 3.4. Entorno del servidor

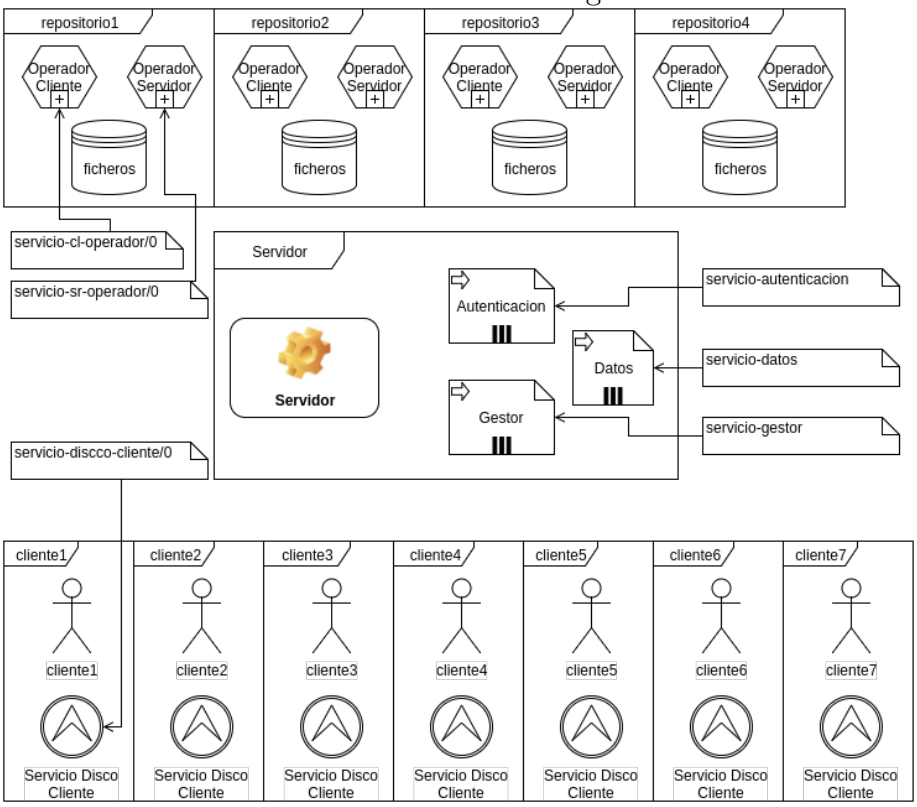
El entorno del servidor usara las siguientes clases


«interface»
Remote
«interface»
ServicioDatosInterface
«interface»
ServicioAutenticacionInterface
«interface»
ServicioGestorInterface
ServicioDatosImpl
ServicioAutenticacionImpl
ServicioGestorImpl
Servidor
Collections
Logger
Utils
GUI
Cliente
Repositorio
«interface»
Serializable

# Capítulo 4

# Ejemplos de uso

En este capitulo mostraremos ejemplos de uso de la aplicación, para los ejemplos he registrado en el servidor cuatro repositorios y siete clientes, cada cliente tendra asociados varios ficheros. Este sera el diagrama de los elementos en las pruebas.



## 4.1. Operaciones del servidor

```
==========Menu Servidor==========
1-Listar Clientes
2-Listar Repositorios
3-Listar Parejas Repositorio-Cliente
4-Salir
==========Menu Servidor==========
Seleccione una opcion: 1
-- Clientes registrados en el sistema --
Cliente: 0-cliente1
Cliente: 1-cliente2
Cliente: 2-cliente3
Cliente: 3-cliente4
Cliente: 4-cliente5
Cliente: 5-cliente6
Cliente: 6-cliente7
==========Menu Servidor==========
1-Listar Clientes
2-Listar Repositorios
3-Listar Parejas Repositorio-Cliente
4-Salir
==========Menu Servidor==========
Seleccione una opcion: 2
-- Repositorios registrados en el sistema --
Repositorio: 0-repositorio1
Repositorio: 1-repositorio2
Repositorio: 2-repositorio3
Repositorio: 3-repositorio4
==========Menu Servidor==========
1-Listar Clientes
2-Listar Repositorios
3-Listar Parejas Repositorio-Cliente
4-Salir
==========Menu Servidor==========
```

```
Seleccione una opcion: 3
-- Clientes - repositorio registrados en el sistema --
Cliente: 0-cliente1 -> 0-repositorio1
Cliente: 1-cliente2 -> 1-repositorio2
Cliente: 2-cliente3 -> 2-repositorio3
Cliente: 3-cliente4 -> 3-repositorio4
Cliente: 4-cliente5 -> 0-repositorio1
Cliente: 5-cliente6 -> 1-repositorio2
Cliente: 6-cliente7 -> 2-repositorio3
==========Menu Servidor==========
1-Listar Clientes
2-Listar Repositorios
3-Listar Parejas Repositorio-Cliente
4-Salir
==========Menu Servidor==========
Seleccione una opcion:
```

## 4.2. Operaciones del repositorio

Este es un ejemplo de las operaciones disponibles en los repositorios.

### 4.2.1. Registro nuevo repositorio

```
==========Menu Repositorio==========
1-Registrar nuevo repositorio
2-Autenticarse en el sistema
3-Salir
==========Menu Repositorio==========
Seleccione una opcion: 1
Introduzca el usuario: repositorio1
Introduzca la contraseña: rep1
Introduzca la contraseña de nuevo: rep1
========== repositorio0 ==========
1-Listar Clientes
```

```
2-Listar ficheros del Cliente
3-Salir
========== repositorio0 ==========
Seleccione una opcion:
```

### 4.2.2. Operaciones repositorio

```
========== repositorio0 ==========
1-Listar Clientes
2-Listar ficheros del Cliente
3-Salir
========== repositorio0 ==========
Seleccione una opcion: 1
Listado de clientes en el repositorio
Cliente: 0-cliente1
Cliente: 4-cliente5
========== repositorio0 ==========
1-Listar Clientes
2-Listar ficheros del Cliente
3-Salir
========== repositorio0 ==========
Seleccione una opcion: 2
Clientes validos:
0 4
Seleccione un cliente: 0
Fichero: test1.txt
Fichero: test2.txt
========== repositorio0 ==========
1-Listar Clientes
2-Listar ficheros del Cliente
3-Salir
========== repositorio0 ==========
Seleccione una opcion:
```

## 4.3. Cliente

Ejemplo de uso del cliente

### 4.3.1. Registro nuevo cliente

```
==========Menu Cliente==========
1-Registrar nuevo cliente
2-Autenticarse en el sistema
3-Salir
==========Menu Cliente==========
Seleccione una opcion: 1
Introduzca usuario: cliente3
Introduzca contraseña: cli3
Introduzca la contraseña de nuevo: cli3
==========-cliente3-==========
1-Subir fichero
2-Bajar fichero
3-Borrar fichero
4-Compartir ficheros
5-Listar ficheros
6-Listar clientes del sistema
7-Salir
==========-cliente3-==========
Seleccione una opcion:
```

### 4.3.2. Operación subir fichero

```
==========-cliente3-==========
1-Subir fichero
2-Bajar fichero
3-Borrar fichero
4-Compartir ficheros
5-Listar ficheros
```

```
6-Listar clientes del sistema
7-Salir
==========-cliente3-==========
Seleccione una opcion: 1
Introduzca fichero a enviar: test1.txt
Fichero enviado correctamente a la nube
```

### 4.3.3. Operación descargar fichero

```
==========-cliente3-==========
1-Subir fichero
2-Bajar fichero
3-Borrar fichero
4-Compartir ficheros
5-Listar ficheros
6-Listar clientes del sistema
7-Salir
==========-cliente3-==========
Seleccione una opcion: 2
test1.txt
Seleccione el fichero que quiere recibir: test1.txt
Escribiendo fichero en test1.txt
```

### 4.3.4. Operación borrar un fichero

```
==========-cliente3-==========
1-Subir fichero
2-Bajar fichero
3-Borrar fichero
4-Compartir ficheros
5-Listar ficheros
6-Listar clientes del sistema
7-Salir
```

```
==========cliente3==========
Seleccione una opcion: 3
test1.txt
Seleccione el fichero que quiere eliminar: test1.txt
```

### 4.3.5. Otras operaciones

Listado de ficheros del cliente y de los clientes del sistema.

```
==========cliente1==========
1-Subir fichero
2-Bajar fichero
3-Borrar fichero
4-Compartir ficheros
5-Listar ficheros
6-Listar clientes del sistema
7-Salir
==========cliente1==========
Seleccione una opcion: 5
Fichero: test1.txt
Fichero: test2.txt
==========cliente1==========
1-Subir fichero
2-Bajar fichero
3-Borrar fichero
4-Compartir ficheros
5-Listar ficheros
6-Listar clientes del sistema
7-Salir
==========cliente1==========
Seleccione una opcion: 6
Cliente: 0-cliente1
Cliente: 1-cliente2
Cliente: 2-cliente3
Cliente: 3-cliente4
```

```
Cliente: 4-cliente5
Cliente: 5-cliente6
Cliente: 6-cliente7
```

# Apéndice A

# Herramientas

Para realizar esta practica se han utilizado estas herramientas:

OpenJDK Java 1.8 http://openjdk.java.net/ Ant para construir los ejecutables del proyecto http://ant.apache.org/ Eclipse para edición del código http://www.eclipse.org/ git como sistema para mantener el control de la evolución del código y permitir desarrollar desde varios PC https://git-scm.com/ draw.io, herramienta en la nube para generar graficos https://www.draw.io/ Latex, entorno tex para realizar la memoria

# Apéndice B

# Riesgos, consideraciones y conclusiones

El proceso para desarrollo de la practica ha sido un proceso incremental por lo que algunas decisiones de diseño pueden no ser todo lo correctas, en el código fuente entregado esta también el histórico y la evolución del código, el histórico se puede consultar usando la herramienta de control de versiones **git**.

## B.1. Revisión de decisiones de diseño

1. La clase cliente que mantiene la información de los clientes en el servicio de datos necesita una refactorización, en muchos métodos remotos estamos enviando toda la información del cliente, eso no esta optimizado y en sucesiones iteraciones del desarrollo habría que optimizarlo para enviar solo la información necesaria

2. En la aplicación no se comprueba si los servicios están publicados correctamente y se da por supuesto que todos los objetos remotos están operativos. En siguientes iteraciones del proceso de desarrollo habría que ver que implicaciones tendría tener un nivel mas fino de control de errores.

3. El repositorio crea una carpeta con su id para almacenar los ficheros, simplemente es para facilitar las pruebas con varios repositorios lanzados desde la

misma carpeta.