

# **EnCuadro 2.0**

**GLOSARIO, APÉNDICE Y BIBLIOGRAFÍA**

Universidad de la República

UTU - CETP

Paysandú, 27 de Mayo de 2013

**Tutor:**

**Juan Cardelino**

**Estudiantes:**

**Mauricio Martínez**

**María Clara Fleitas**

**Martín Arévalo**

**Virginia Saldaña**

<b>1. Referencias.....</b>	<b>5</b>
<b>2. Apéndice.....</b>	<b>8</b>
2.1. Casos de uso .....	8
Requerimientos funcionales .....	8
Aplicación móvil .....	8
Modo manual .....	8
2.1.1. Listar salas.....	8
2.1.2. Listar obras .....	9
Modo automático.....	9
2.1.3. Identificar sala .....	9
2.1.4. Identificar obra .....	10
2.1.5. Obtener datos y contenidos .....	10
2.1.6. Realidad aumentada .....	11
Aplicación de escritorio .....	11
Modo administrador .....	12
2.1.7. Iniciar sesión .....	12
2.1.8. Agregar usuario .....	12
2.1.9. Eliminar usuario .....	13
2.1.10. Listar usuarios .....	13
2.1.11. Modificar usuario .....	14
2.1.12. Cerrar sesión.....	14
Modo empleado.....	15
2.1.13. Iniciar sesión .....	15
2.1.14. Agregar sala.....	16
2.1.15. Eliminar sala.....	17
2.1.16. Listar salas.....	17
2.1.17. Modificar sala.....	17
2.1.18. Agregar obra.....	18
2.1.19. Eliminar obra .....	19
2.1.20. Listar obra .....	19
2.1.21. Modificar obra.....	20
2.1.22. Agregar zona de interés .....	21
2.1.23. Eliminar zona de interés .....	21
2.1.24. Modificar zona de interés .....	22
2.1.25. Cerrar sesión.....	22
2.2. Diagramas de secuencia del sistema .....	23
Aplicación móvil .....	23
Modo manual .....	23
2.2.1. Listar salas.....	23
2.2.2. Listar obras .....	23
Modo automático.....	24
2.2.3. Identificar sala .....	24
2.2.4. Identificar obra .....	24
2.2.5. Obtener datos y contenidos .....	25
2.2.6. Realidad aumentada .....	26
Aplicación de escritorio .....	26
Modo administrador .....	26
2.2.7. Iniciar sesión .....	26

2.2.8. Agregar usuario .....	27
2.2.9. Eliminar usuario .....	27
2.2.10. Listar usuarios .....	28
2.2.11. Modificar usuario .....	29
2.2.12. Cerrar sesión.....	29
Modo empleado.....	30
2.2.13. Iniciar sesión .....	30
2.2.14. Agregar sala.....	31
2.2.15. Eliminar sala.....	32
2.2.16. Listar salas.....	33
2.2.17. Modificar sala.....	34
2.2.18. Agregar obra.....	35
2.2.19. Eliminar obra.....	36
2.2.20. Listar obras .....	37
2.2.21. Modificar obra.....	38
2.2.22. Agregar zona de interés.....	39
2.2.23. Eliminar zona de interés.....	40
2.2.24. Modificar zona de interés .....	41
2.2.25. Cerrar sesión.....	42
<b>2.3. Contratos .....</b>	<b>43</b>
Aplicación móvil .....	43
Modo manual .....	43
2.3.1. Listar salas.....	43
2.3.2. Listar obras .....	44
Modo automático.....	45
2.3.3. Identificar sala .....	45
2.3.4. Identificar obra .....	46
2.3.5. Obtener datos y contenidos .....	47
2.3.6. Realidad aumentada .....	48
Aplicación de escritorio .....	50
Modo administrador .....	50
2.3.7. Iniciar sesión .....	50
2.3.8. Agregar usuario .....	50
2.3.9. Modificar usuario .....	51
2.3.10. Listar usuarios .....	52
2.3.11. Eliminar usuario .....	53
2.3.12. Cerrar sesión.....	53
Modo empleado.....	54
2.3.14. Agregar sala.....	54
2.3.15. Eliminar sala.....	54
2.3.16. Listar salas.....	55
2.3.17. Modificar sala.....	56
2.3.18. Agregar obra.....	57
2.3.19. Eliminar obra .....	57
2.3.20. Listar obras .....	58
2.3.21. Modificar obra .....	59
2.3.22. Agregar zona de interés .....	60
2.3.23. Eliminar zona de interés .....	62

2.3.24. Modificar zona de interés .....	63
2.3.25. Cerrar sesión.....	63
<b>2.4. Vistas.....</b>	<b>64</b>
Aplicación de escritorio .....	64
2.4.1. Menú principal – Inicio sesión .....	64
Menú administrador .....	65
2.4.2. Menú principal .....	65
2.4.3. Menú usuarios .....	65
2.4.4. Menú sesión.....	66
2.4.5. Menú ayuda.....	66
2.4.6. Agregar usuarios .....	67
Menú empleado .....	67
2.4.7. Menú principal .....	67
2.4.8. Menú usuarios .....	68
2.4.9. Menú salas.....	68
2.4.10. Menú obras .....	69
2.4.11. Menú zona de interés.....	69
2.4.12. Menú sesión.....	70
2.4.13. Menú ayuda.....	70
2.4.14. Agregar salas .....	71
2.4.15. Agregar obras .....	72
2.4.16. Agregar zona de interés.....	74
Aplicación móvil .....	76
2.4.17. Menú principal .....	76
Modo manual .....	77
2.4.18. Lista salas .....	77
2.4.19. Lista obras .....	77
Modo automático.....	78
2.4.20. Identificar sala.....	78
2.4.21. Identificar obra .....	79
2.4.22. Obra completa .....	80
2.4.23. Realidad aumentada .....	81
<b>2.5. Código fuente .....</b>	<b>83</b>
2.5.1. Ejemplo de código para uso de SOAP .....	83
2.5.2. Ejemplo de código para uso de FTP.....	86
<b>3. Glosario .....</b>	<b>93</b>

## 1. Referencias.

- [1] Diccionario de informática, telecomunicaciones y ciencias afines – Mario León
- [2] Documentación del proyecto EnCuadro, Facultad de Ingeniería, Universidad de la República – Juan Braun, Martín Etchart, Pablo Flores y Mauricio González:  
<https://www.dropbox.com/sh/jky5oeef6mrwg4a/yj88zFr3OL/Documentacion%20enCuadro.pdf>
- [3] Universidad del Trabajo de Uruguay:  
<http://www.utu.edu.uy/>
- [4] Facultad de Ingeniería de la Universidad de la República:  
<http://www.fing.edu.uy/>
- [5] Museo Nacional de Artes Visuales:  
<http://mnav.gub.uy/>
- [6] Párrafo extraído del capítulo Conclusiones y trabajo a futuro de documentación de proyecto de fin de carrera de los estudiantes de Ingeniería Eléctrica, Encuadro.  
<https://www.dropbox.com/sh/jky5oeef6mrwg4a/yj88zFr3OL/Documentacion%20enCuadro.pdf>
- [7] Plan Ceibal:  
<http://www.ceibal.edu.uy/Paginas/Inicio.aspx>
- [8] Página oficial de Layar:  
<http://www.layar.com/what-is-layar/>
- [9] Interactividad y museos - Claudia Esther Calva Montiel:  
<http://www.gabinetecomunicacionyeducacion.com/files/adjuntos/Interactividad%20y%20museos%20la%20experiencia%20del%20museo%20interactivo%20de%20econom%C3%ADA%20la%28mide%29%20en%20la%20ciudad%20de%20M%C3%A9xico.pdf> (consultado 10/04/2013).
- [10] Página oficial de Java:  
<http://www.java.com/es/about/> (consultado 03/05/2013).
- [11] Página oficial de Netbeans:  
[https://netbeans.org/index\\_es.html](https://netbeans.org/index_es.html) (consultado 03/05/2013).

[12] Objective-C - Página oficial para desarrolladores de Apple:  
<http://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/Programmin gWithObjectiveC/Introduction/Introduction.html> (consultado 05/05/2013).

[13] Xcode - Página oficial para desarrolladores de Apple:  
[https://developer.apple.com/library/ios/#documentation/ToolsLanguages/Conceptual/Xc ode\\_User\\_Guide/000-About\\_Xcode/about.html](https://developer.apple.com/library/ios/#documentation/ToolsLanguages/Conceptual/Xc ode_User_Guide/000-About_Xcode/about.html) (consultado 05/05/2013).

[14] SOAP - Ing. Sandro Moscatelli:  
[www.fing.edu.uy/inco/grupos/lins/docsgen/soap/soap.doc](http://www.fing.edu.uy/inco/grupos/lins/docsgen/soap/soap.doc) (consultado 06/05/2013).

[15] FTP:  
[http://es.wikipedia.org/wiki/File\\_Transfer\\_Protocol](http://es.wikipedia.org/wiki/File_Transfer_Protocol) (consultado 06/05/2013).

[16] Synthetica Layout:  
<http://www.javasoft.de/synthetica/> (consultado 17/04/2013).

[17] Java Barcode:  
<http://www.barcodelib.com> (consultado 03/04/2013).

[18] Zebra-Crossing:  
<https://code.google.com/p/zxing/> (consultado 10/04/2013).

[19] VTK:  
<http://www.vtk.org/> (consultado 03/04/2013).

[20] Java Media Framework:  
[http://es.wikipedia.org/wiki/Java\\_Media\\_Framework](http://es.wikipedia.org/wiki/Java_Media_Framework) (consultado 08/05/2013).

[21] Plan Ceibal:  
<http://www.ceibal.edu.uy/Paginas/Inicio.aspx> (consultado 12/02/2013).

[22] Definición de comunicación:  
<http://es.wikipedia.org/wiki/Comunicaci%C3%B3n> (consultado 09/04/2013).

[23] New Scientist video – Transparent wall:  
[http://www.youtube.com/watch?feature=player\\_embedded&v=Q5O13bk7z2s](http://www.youtube.com/watch?feature=player_embedded&v=Q5O13bk7z2s)  
(consultado 06/05/2013).

[24] Información en el móvil - Natalia Arroyo Vázquez, Barcelona: Editorial UOC, Año 2011.

[25] Qué es la realidad aumentada y sus aplicaciones – Angel Vilchez:  
<http://www.configurarequipos.com/doc1214.html> (consultado 06/05/2013).

[26] Creación de marcadores para RA – Dirección operativa de incorporación de tecnologías (Plan integral de educación digital), Ministerio de Educación – Buenos Aires Ciudad:

[http://integrar.bue.edu.ar/wp-content/uploads/2011/09/Tutorial\\_Marcadores\\_Realidad\\_Aumentada.pdf](http://integrar.bue.edu.ar/wp-content/uploads/2011/09/Tutorial_Marcadores_Realidad_Aumentada.pdf) (consultado 10/04/2013).

[27] Definición de código QR:

<http://www.codigos-qr.com/> (consultado 07/05/2013).

[28] Información en el móvil – Natalia Arroyo Vázquez, Editorial UOC, 2011:

Usos de código QR, pág. 80.

[29] Análisis de vulnerabilidades de las mallas en los modelos 3D – Jovany Gil García, María Aurora Molina Vilchis:

<http://www.publicaciones.urbe.edu/index.php/telematique/article/viewArticle/1576/html> (consultado 11/04/2013).

[30] Estructuras y modelado 3D:

[http://megazine.co/estructuras-y-modelado-3d-bim-servicios\\_22812.html](http://megazine.co/estructuras-y-modelado-3d-bim-servicios_22812.html) (consultado 11/04/2013).

[31] Swing Absolute Layout – Java Developer Tools:

<https://developers.google.com/java-dev-tools/wbpro/layoutmanagers/swing/absolutelayout?hl=es> (consultado 07/03/2013).

[32] Java Media Framework:

[http://en.wikipedia.org/wiki/Java\\_Media\\_Framework](http://en.wikipedia.org/wiki/Java_Media_Framework) (consultado 01/03/2013).

[33] Documentación oficial de Barcode:

[http://www.barcodelib.com/java\\_barcode/main.html](http://www.barcodelib.com/java_barcode/main.html) (consultado 07/05/2013).

[34] QuickTime for Java:

[http://en.wikipedia.org/wiki/QuickTime\\_for\\_Java](http://en.wikipedia.org/wiki/QuickTime_for_Java) (consultado 19/05/2013).

[35] Página oficial de VTK:

<http://www.vtk.org/> (consultado 19/05/2013).

[36] Página oficial de Synthetica:

<http://www.javasoft.de/synthetica/> (consultado 01/03/2013).

## 2. Apéndice.

En este documento se describen todos los casos de uso, requerimientos funcionales, diagramas de secuencia del sistema, contratos, requerimientos de infraestructura informática de la aplicación de escritorio EnCuadro 2.0 y de la aplicación móvil que fueron modificados en nuestro proyecto.

### 2.1. Casos de uso.

#### Requerimientos funcionales.

##### Aplicación móvil.

Una vez que el visitante del museo ingresa a la red local del museo, podrá tener acceso a una aplicación en la que obtendrá de manera manual o automática a los datos (audio, texto, descripción, etc.) de salas y obras del museo y acceder a ciertas realidades aumentadas (video, modelo 3D, animación 3D), que se encuentran administrados en el servidor. También se le habilitarán funcionalidades para que pueda publicar desde la aplicación en su cuenta de red social ‘Twitter’.

##### Modo manual.

###### 2.1.1. Listar salas.

Caso de uso	<b>Listar salas</b>
Actores	Visitante del museo
Descripción	Este caso de uso lista todas las salas existentes en el sistema con su nombre, imagen y descripción para que el visitante seleccione la que desee, pudiendo acceder a todas las obras pertenecientes a la sala seleccionada.

## 2.1.2. Listar obras.

Caso de uso	<b>Listar obras</b>
Actores	Visitante del museo
Descripción	Se listan los datos (nombre, descripción e imagen) de todas las obras pertenecientes a la sala seleccionada en el caso de uso ‘Listar salas’, para que el visitante seleccione la deseada para ver sus datos y contenidos disponibles.

## Modo automático.

### 2.1.3. Identificar sala.

Caso de uso	<b>Identificar sala</b>
Actores	Visitante del museo
Descripción	Este caso de uso permite identificar la sala del museo en la que se encuentra el visitante. Se despliega una pantalla con la toma de la cámara y al enfocar el código QR se detecta el ID de la sala, con el cual se hace una petición al servidor para obtener los datos de la sala correspondientes, en caso de que el ID transmitido por el QR no sea una sala válida, se le notificará al usuario.

### **2.1.4. Identificar obra.**

Caso de uso	<b>Identificar obra</b>
Actores	Visitante del museo
Descripción	<p>Este caso de uso permite al visitante identificar la obra que está observando. La persona toma una fotografía con su dispositivo móvil de la obra en cuestión (de la sala previamente seleccionada en el caso de uso: ‘Identificar sala’) y la envía al servidor, junto con el ID de la sala que fue previamente admitido por el código QR. El servidor identifica mediante un algoritmo de comparación de imágenes la obra a la cual se le tomó la foto y devuelve el ID de la obra. En caso de que la obra no pertenezca a la sala o no exista se devolverá el ID de la obra que más parecida sea a la foto.</p>

### **2.1.5. Obtener datos y contenidos.**

Caso de uso	<b>Obtener datos y contenido</b>
Actores	Visitante del museo
Descripción	<p>Este caso de uso es usado en los dos modos de uso de la aplicación. La aplicación le solicita al servidor los datos y contenidos de la obra actual, para ello se envía el identificador de la obra (idObra, obtenido en el caso de uso ‘Identificar obra’ o en el caso de uso ‘Listar obras’). Al recibir los contenidos, se muestran los datos principales (nombre, autor, imagen, descripción, texto y audio).</p>

## 2.1.6. Realidad aumentada.

Caso de uso	<b>Obtener datos y contenido</b>
Actores	Visitante del museo
Descripción	Este caso de uso es usado en los dos modos de uso de la aplicación. Se presentan las opciones de los contenidos existentes para realidad aumentada de la obra (video, modelo 3D y/o animación 3D) y según la elección del usuario, la realidad aumentada que se reproduce.

## Aplicación de escritorio.

Una vez instalada la aplicación, el usuario podrá trabajar sin problemas sobre la misma. Primariamente las tareas que se podrán realizar son ABM<sup>1</sup> de los diferentes objetos con los cuales se trabajara en el sistema (usuarios, salas, obras y zonas de interés). El usuario podrá hacer uso del sistema visualizando, modificando y eliminando cada uno de estos objetos con sus respectivos datos asociados, como por ejemplo, cuando se listan los usuarios del sistema, se mostrarán sus datos personales (nombre, apellido, cédula de identidad, etc.), logrando que se pueda visualizar estos datos y modificarlos. A diferencia de los usuarios tanto las salas como las obras, cuentan con un nombre y una descripción (las obras también tienen autor), y tienen datos asociados llamados contenidos. Estos contenidos pueden ser (texto, video, audio, modelo 3D y animaciones 3D) que podrán ser visualizados y modificados en caso de ser necesario.

El sistema podrá ser manipulado por dos tipos de usuarios (Administrador o Empleado), que es detectado automáticamente cuando el usuario inicia sesión con su ‘usuario’ y ‘contraseña’. Dependiendo del tipo de usuario que inicio sesión se le habilitarán diferentes funcionalidades las cuales son exclusivas de cada tipo de usuario, es decir, un usuario Administrador no podrá utilizar las funcionalidades de un usuario tipo Empleado, y viceversa.

---

1. Alta, baja y modificación.

## Modo administrador.

### 2.1.7. Iniciar sesión.

Caso de uso	<b>Iniciar sesión</b>
Actores	Administrador
Descripción	Este caso de uso permite a un usuario Administrador iniciar sesión en el sistema mediante su nick de usuario y contraseña. El sistema verifica que los datos sean válidos. En caso positivo se le da acceso y en caso contrario el sistema informa del error. Sin realizar este caso de uso, el usuario no podrá acceder a ninguno de los otros casos de uso existentes.

### 2.1.8. Agregar usuario.

Caso de uso	<b>Agregar usuario</b>
Actores	Administrador
Descripción	Este caso de uso le permite al administrador ingresar nuevos usuarios al sistema. Los datos necesarios para ingresar el nuevo usuario son: Nombre, Apellido, CI, Email, Tipo Usuario (Administrador o Empleado), Nick y Contraseña (se le asignará una contraseña por defecto, “default”, que podrá ser modificada posteriormente). En caso de error de validación de datos el sistema informará del mismo.

### 2.1.9. Eliminar usuario.

Caso de uso	<b>Eliminar usuario</b>
Actores	Administrador
Descripción	Este caso de uso le permite al administrador eliminar un usuario administrador o empleado del sistema. El sistema lista los usuarios existentes y luego el administrador selecciona uno para ser eliminado (siempre y cuando no sea su propio usuario). Finalmente el administrador confirma la acción y el sistema procede a eliminarlo.

### 2.1.10. Listar usuarios.

Caso de uso	<b>Listar usuarios</b>
Actores	Administrador
Descripción	El sistema lista todos los usuarios ingresados en el sistema, luego el administrador selecciona uno y el sistema muestra todos los datos correspondientes al mismo. Están implementadas las funcionalidades de listar los usuarios discriminando entre Administradores y Empleados o todos los usuarios existentes en el sistema.

### 2.1.11. Modificar usuario.

Caso de uso	<b>Modificar usuario</b>
Actores	Administrador
Descripción	Este caso de uso le permite al administrador modificar y guardar los datos de un usuario (administrador o empleado) previamente ingresado en el sistema. En caso de error de validación de datos el sistema informará del mismo y los cambios no son almacenados hasta que dichos datos sean corregidos.

### 2.1.12. Cerrar sesión.

Caso de uso	<b>Cerrar sesión</b>
Actores	Administrador
Descripción	Este caso de uso le permite al administrador cerrar su sesión en el sistema y lo retorna a la pantalla inicial de inicio de sesión.

## Modo empleado.

### 2.1.13. Iniciar sesión.

Caso de uso	<b>Iniciar sesión</b>
Actores	Empleado de Museo
Descripción	Este caso de uso permite a un empleado del museo iniciar sesión en el sistema con un usuario y contraseña. El sistema verifica que los datos sean válidos. En caso positivo se le da acceso al sistema y en caso contrario el sistema informa del error. Sin realizar este caso de uso, el usuario no podrá acceder a ninguno de los otros casos de uso existentes.

## **2.1.14. Agregar sala.**

Caso de uso	<b>Agregar sala</b>
Actores	Empleado de Museo
Descripción	<p>Este caso de uso le permite agregar al sistema una nueva sala al usuario. El empleado ingresa los datos de la sala a dar de alta: nombre de sala y descripción, y se registra en el sistema generando el código QR que le corresponderá.</p> <p>Luego el sistema da opciones para elegir qué tipo de contenido desea el usuario agregarle a esta sala, y dependiendo de lo elegido (texto, modelo 3D, video, audio) los datos que el sistema pide.</p> <p><u>Texto</u>: El sistema pide un texto y después de ingresado se confirma, si desea cambiarlo, puede hacerlo hasta ser confirmados los contenidos de la sala.</p> <p><u>Modelo 3D</u>: El sistema pide ingresar un archivo de modelo 3D (en formato .ply). Una vez ingresado se muestra el modelo en una ventana de renderizado para su visualización. Si el usuario desea cambiar el archivo puede ingresar nuevamente un modelo que sustituirá el anteriormente ingresado, este modelo se muestra y así hasta ser confirmados los contenidos de la sala.</p> <p><u>Video</u>: El sistema pide ingresar un archivo de video (en formato .mov). Una vez ingresado el archivo es reproducido automáticamente. Si lo desea puede editar el mismo, siendo reproducido al igual que el original, y así hasta ser confirmados los contenidos de la sala.</p> <p><u>Audio</u>: El sistema pide ingresar un archivo (en formato .mp3). Una vez ingresado se reproduce, si lo desea puede editar el archivo ingresado, se reproduce el mismo, y así hasta ser confirmados los contenidos de la sala.</p> <p>Sólo se podrá ingresar un contenido de cada tipo. Si el usuario está conforme con los datos ingresados confirma los contenidos para la sala, y se guardan los datos correspondientes. En caso de error de validación de datos el sistema informará del mismo y la sala no es dada de alta hasta que dichos datos sean corregidos.</p>

### **2.1.15. Eliminar sala.**

Caso de uso	<b>Eliminar sala</b>
Actores	Empleado de Museo
Descripción	Este caso de uso le permite al usuario eliminar una sala del sistema. El usuario ingresa el identificador de la sala a eliminar y el sistema lista los datos correspondientes a la sala. Finalmente el empleado confirma la acción y el sistema procede a eliminar la sala.

### **2.1.16. Listar salas.**

Caso de uso	<b>Listar salas</b>
Actores	Empleado de Museo
Descripción	El sistema lista todas las salas ingresadas en el sistema, luego el usuario selecciona una de las salas y el sistema muestra todos los datos correspondientes a la misma, permitiendo visualizar los contenidos asignados e imprimir el código QR correspondiente.

### **2.1.17. Modificar sala.**

Caso de uso	<b>Modificar sala</b>
Actores	Empleado de Museo
Descripción	Este caso de uso le permite al empleado modificar una sala previamente creada en el sistema. El usuario busca la sala a modificar mediante su número de identificación (idSala) y el sistema retorna los datos referentes a la misma, permitiendo visualizarlos y/o modificarlos. Posteriormente el usuario confirma los cambios realizados y el sistema guarda las modificaciones efectuadas. En caso de error de validación de datos el sistema informará del mismo y la sala no es dada de alta hasta que dichos datos sean corregidos.

## 2.1.18. Agregar obra.

Caso de uso	<b>Agregar obra</b>
Actores	Empleado de Museo
Descripción	<p>El caso de uso comienza cuando el usuario desea agregar una nueva obra. Para esto se listan todas las salas existentes en el sistema y se selecciona la sala en la cual se desea crear la obra, posteriormente se ingresa autor de la obra, el nombre de la misma, su imagen y la descripción. Luego el sistema da opciones para elegir qué tipo de contenido desea el usuario agregarle a esta obra, y dependiendo de lo elegido (texto, modelo 3D, video, audio, animación) los datos que el sistema pide.</p> <p><u>Texto:</u> El sistema pide un texto y después de ingresado se confirma, si desea cambiarlo, puede hacerlo hasta que la obra sea confirmada.</p> <p><u>Modelo 3D:</u> El sistema pide ingresar un archivo de modelo 3D (en formato .pod). Si el usuario desea cambiar el archivo puede ingresar nuevamente un modelo que sustituirá el anteriormente ingresado, este modelo se muestra y así hasta ser confirmada la obra.</p> <p><u>Video:</u> El sistema pide ingresar un archivo de video (en formato .mov). Una vez ingresado el archivo es reproducido automáticamente. Si lo desea puede editar el mismo, siendo reproducido al igual que el original, y así hasta que sea confirmada la obra.</p> <p><u>Audio:</u> El sistema pide ingresar un archivo (en formato .mp3). Una vez ingresado se reproduce, si lo desea puede editar el archivo ingresado, se reproduce el mismo, y así hasta que sea confirmada la obra.</p> <p><u>Animación 3D:</u> El sistema pide ingresar cinco archivos (en formato .pod) que formarán la animación 3D. Si el usuario lo desea puede modificar los archivos, sustituyendo los cinco previamente seleccionados por otro quinteto, por lo que debe cargar los cinco archivos nuevamente (no es posible modificar sólo uno de ellos).</p> <p>Sólo se podrá ingresar un contenido de cada tipo. Si el usuario está conforme con los datos ingresados confirma la obra, y se guardan los datos correspondientes. En caso de error de validación de datos el sistema informará del mismo y la obra no es dada de alta hasta que dichos datos sean corregidos.</p>

## 2.1.19. Eliminar obra.

Caso de uso	<b>Eliminar obra</b>
Actores	Empleado de Museo
Descripción	El caso de uso comienza cuando el usuario desea eliminar una obra. Se listan todas las salas existentes en el sistema, y al seleccionar una se muestran todas las obras pertenecientes a la misma. El empleado selecciona de la lista la obra a eliminar, mostrándose sus datos. Si se confirma la eliminación el sistema nos preguntará si estamos seguros de querer eliminar la obra, si la respuesta es positiva se eliminará la misma del sistema.

## 2.1.20. Listar obra.

Caso de uso	<b>Listar obras</b>
Actores	Empleado de Museo
Descripción	El caso de uso comienza cuando el usuario desea ver todas las obras existentes. El sistema muestra una lista de todas las salas del sistema, luego se listan las obras de la sala seleccionada y el usuario elige la obra deseada para que el sistema despliegue todos los datos de la misma, haciendo que sea posible visualizar los contenidos de ella.

### **2.1.21. Modificar obra.**

Caso de uso	<b>Modificar obra</b>
Actores	Empleado de Museo
Descripción	<p>El caso de uso comienza cuando el usuario desea modificar una obra previamente dada de alta. Para esto se listan las salas existentes y luego de seleccionada una de ellas, se muestran las obras pertenecientes a la misma. Luego de seleccionada una de estas obras, se muestran sus datos y los contenidos pertenecientes a ella con su opciones en ambos para modificarlos y visualizarlos.</p> <p><u>Texto:</u> El sistema muestra el texto correspondiente a la obra, en caso de que exista. Si desea cambiarlo, puede hacerlo hasta que la modificación de la obra sea confirmada.</p> <p><u>Modelo 3D:</u> El sistema muestra el modelo 3D correspondiente a la obra, en caso de que exista. Se podrá sustituir el mismo seleccionando otro, así hasta ser confirmada la modificación de la obra.</p> <p><u>Video:</u> El sistema muestra el video correspondiente a la obra, en caso de que exista. Se podrá sustituir el mismo seleccionando otro. Una vez ingresado se reproduce, y así hasta que sea confirmada la modificación de la obra.</p> <p><u>Audio:</u> El sistema muestra el audio correspondiente a la obra, en caso de que lo hubiese. Se podrá sustituir el mismo seleccionando otro. Una vez ingresado se reproduce, y así hasta que sea confirmada la modificación de la obra.</p> <p><u>Animación 3D:</u> El sistema muestra los archivos que conforman la animación correspondiente a la obra, en caso que la hubiese. Se podrá modificar la misma seleccionando nuevamente los cinco archivos que la conforman.</p> <p>Si el usuario está conforme con los datos ingresados confirma la obra, y se guardan los nuevos datos. En caso de error de validación de datos el sistema informará del mismo y la obra no es modificada hasta que dichos datos sean corregidos.</p>

### **2.1.22. Agregar zona de interés.**

Caso de uso	<b>Agregar zona de interés</b>
Actores	Empleado de Museo
Descripción	<p>Este caso de uso le permite al usuario agregar una zona de interés en una obra existente. El sistema lista las salas existentes y al seleccionar una se listan las obras asociadas a la misma. El usuario selecciona una obra y se despliega la imagen de la obra para poder marcar una zona de interés. Luego de marcada la zona se podrán asignar los contenidos (debe tener por lo menos un contenido asociado y uno sólo de cada tipo). Finalmente el usuario confirma la zona y sus contenidos y se crea la zona de interés. En caso de error de validación de datos el sistema informará del mismo y la zona no es dada de alta hasta que dichos datos sean corregidos.</p>

### **2.1.23. Eliminar zona de interés.**

Caso de uso	<b>Eliminar zona de interés.</b>
Actores	Empleado de Museo
Descripción	<p>Este caso de uso le permite al usuario eliminar una zona de interés previamente ingresada. El sistema lista las salas existentes y al seleccionar una se listan las obras asociadas a dicha sala. El usuario selecciona una obra y se listan los identificadores (ID) de las zonas existentes en ella. Al seleccionar un ID se muestra la zona en la imagen de la obra y sus contenidos asignados. Si el usuario confirma que desea eliminar esa zona, se elimina la misma y todos sus contenidos asociados.</p>

### **2.1.24. Modificar zona de interés.**

Caso de uso	<b>Modificar zona de interés</b>
Actores	Empleado del museo.
Descripción	Este caso de uso le permite al empleado del museo modificar la zona de interés de la obra que desee. El sistema lista las salas existentes, y al seleccionarla se listan las obras pertenecientes a dicha sala. Al seleccionar una obra se listan las zonas de interés y se muestra la zona y los contenidos pertenecientes a la misma, permitiendo modificar cualquiera de estos o redibujar la zona actual. Si el usuario confirma la modificación se guardan los cambios. En caso de error de validación de datos el sistema informará del mismo y la zona no es modificada hasta que dichos datos sean corregidos.

### **2.1.25. Cerrar sesión.**

Caso de uso	<b>Cerrar sesión</b>
Actores	Empleado del museo.
Descripción	Este caso de uso le permite al Empleado del museo cerrar su sesión en el programa, y retornar a la pantalla inicial de inicio de sesión.

## 2.2. Diagramas de secuencia del sistema.

Aplicación móvil.

Modo manual.

### 2.2.1. Listar salas.

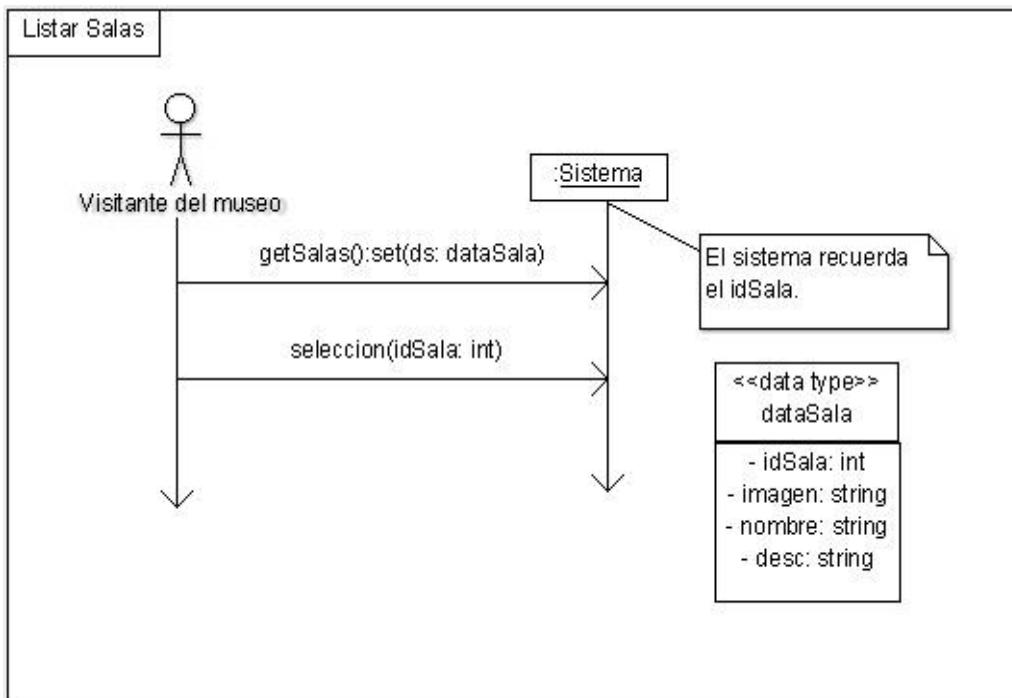


Figura 1

### 2.2.2. Listar obras.

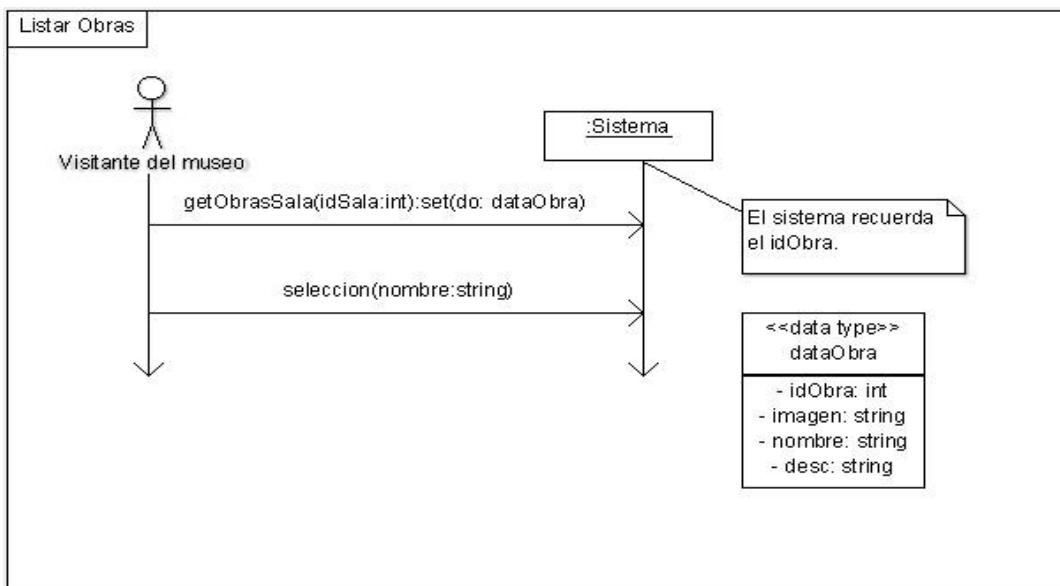


Figura 2

## Modo automático.

### 2.2.3. Identificar sala.

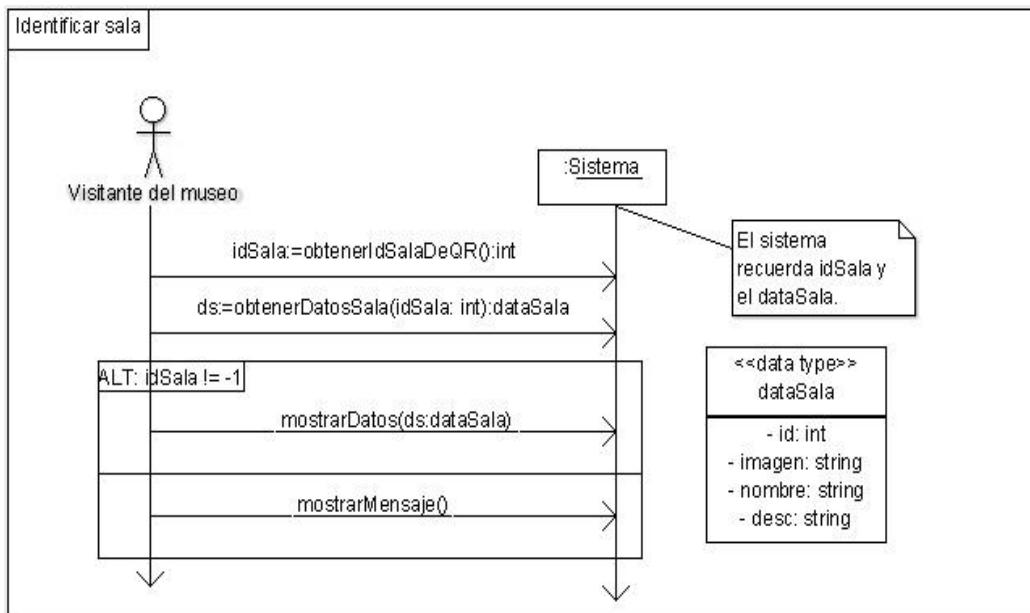


Figura 3

### 2.2.4. Identificar obra.

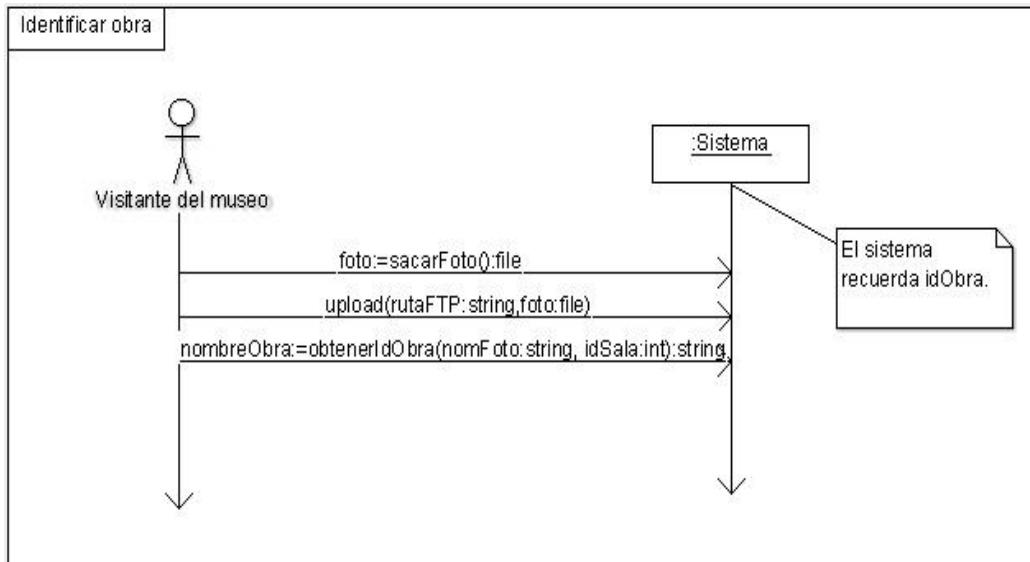


Figura 4

## 2.2.5. Obtener datos y contenidos.

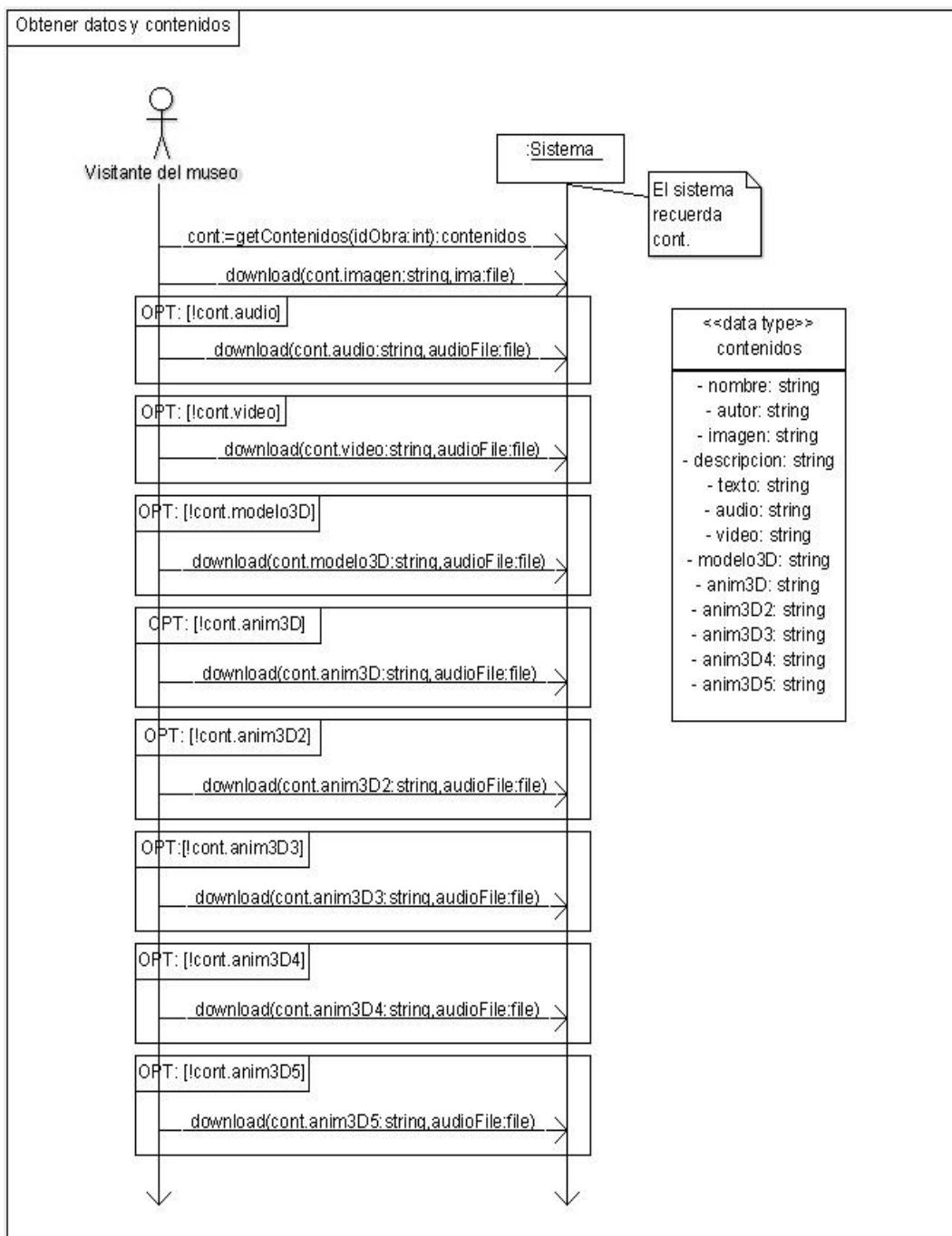


Figura 5

## 2.2.6. Realidad aumentada.

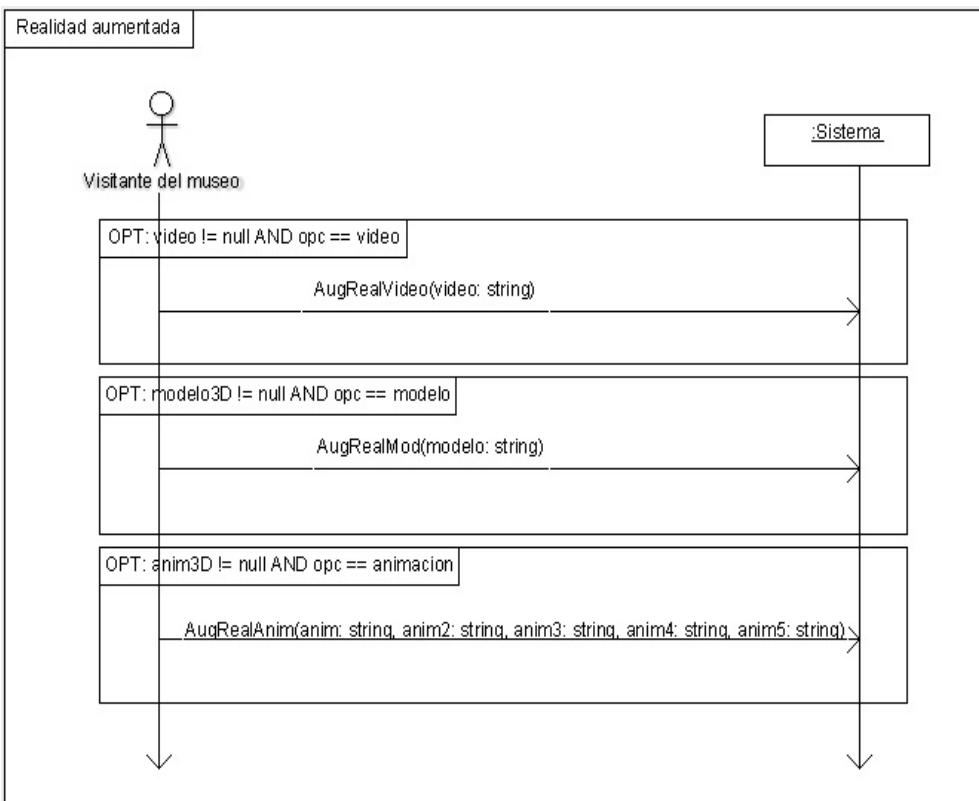


Figura 6

## Aplicación de escritorio.

### modo administrador.

#### 2.2.7. Iniciar sesión.

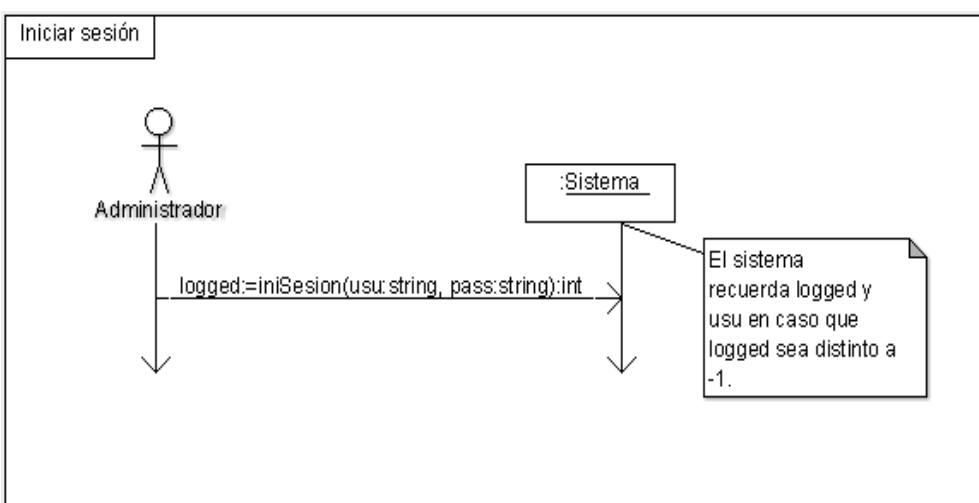


Figura 7

## 2.2.8. Agregar usuario.

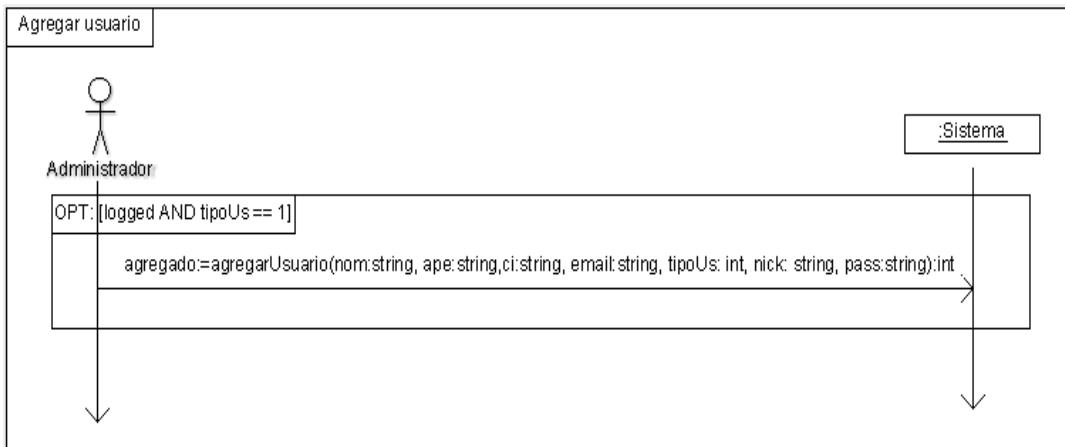


Figura 8

## 2.2.9. Eliminar usuario.

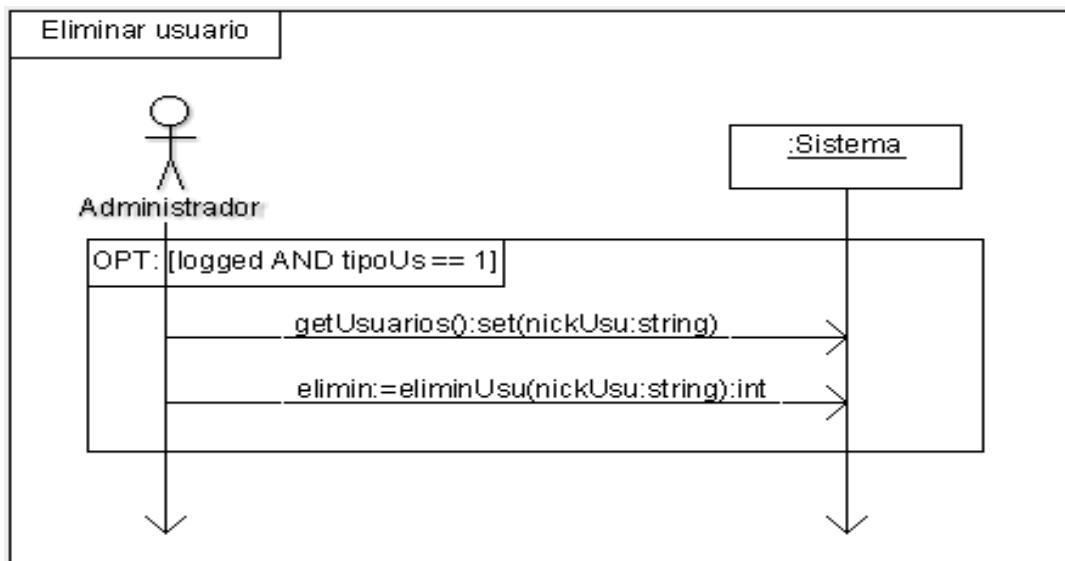
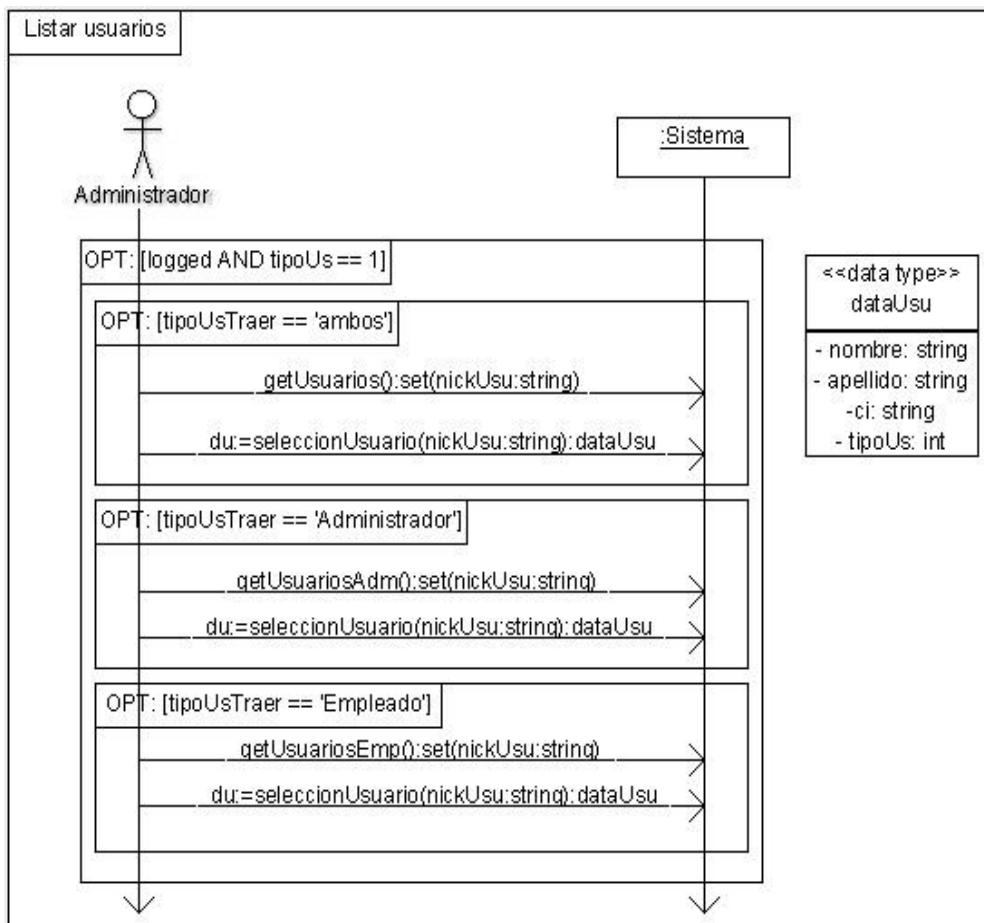


Figura 9

## 2.2.10. Listar usuarios.



**Figura 10**

## 2.2.11. Modificar usuario.

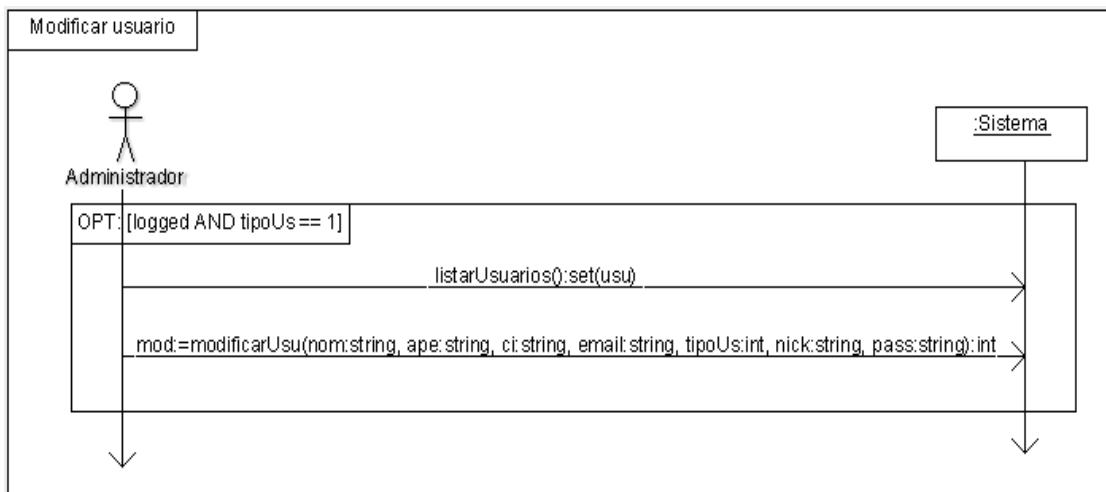


Figura 11

## 2.2.12. Cerrar sesión.

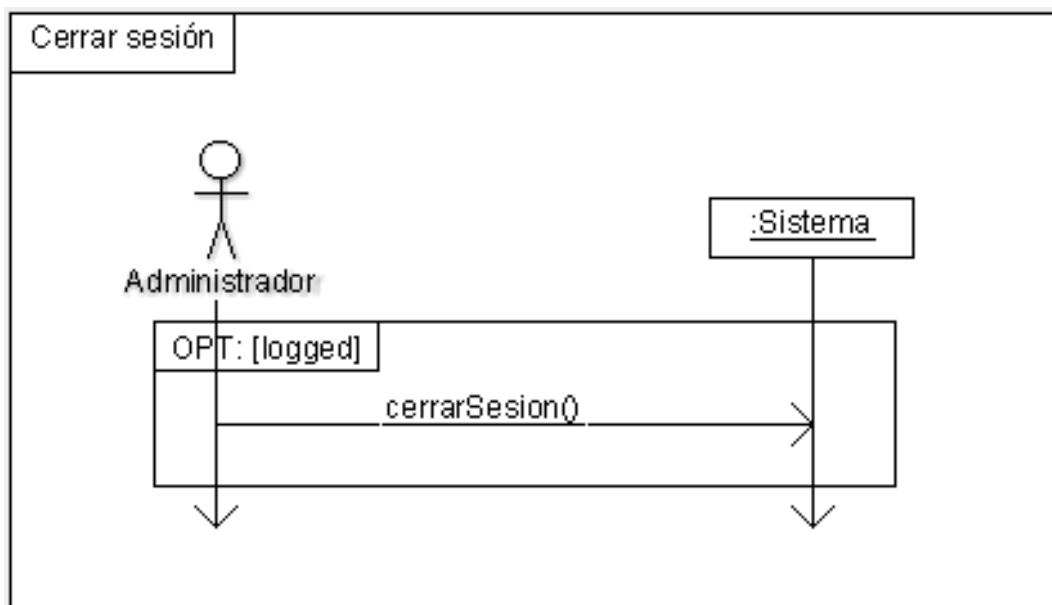


Figura 12

## Modo empleado.

### 2.2.13. Iniciar sesión.

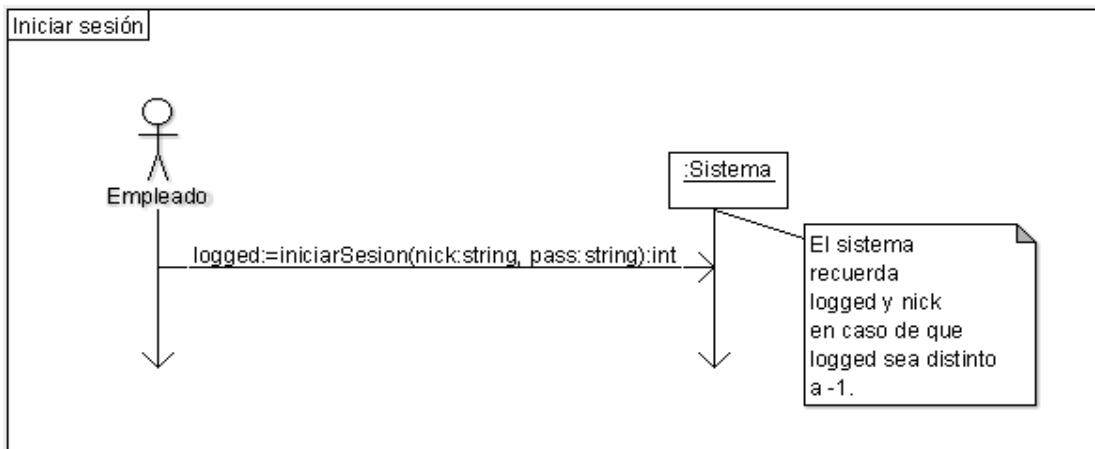


Figura 13

## 2.2.14. Agregar sala.

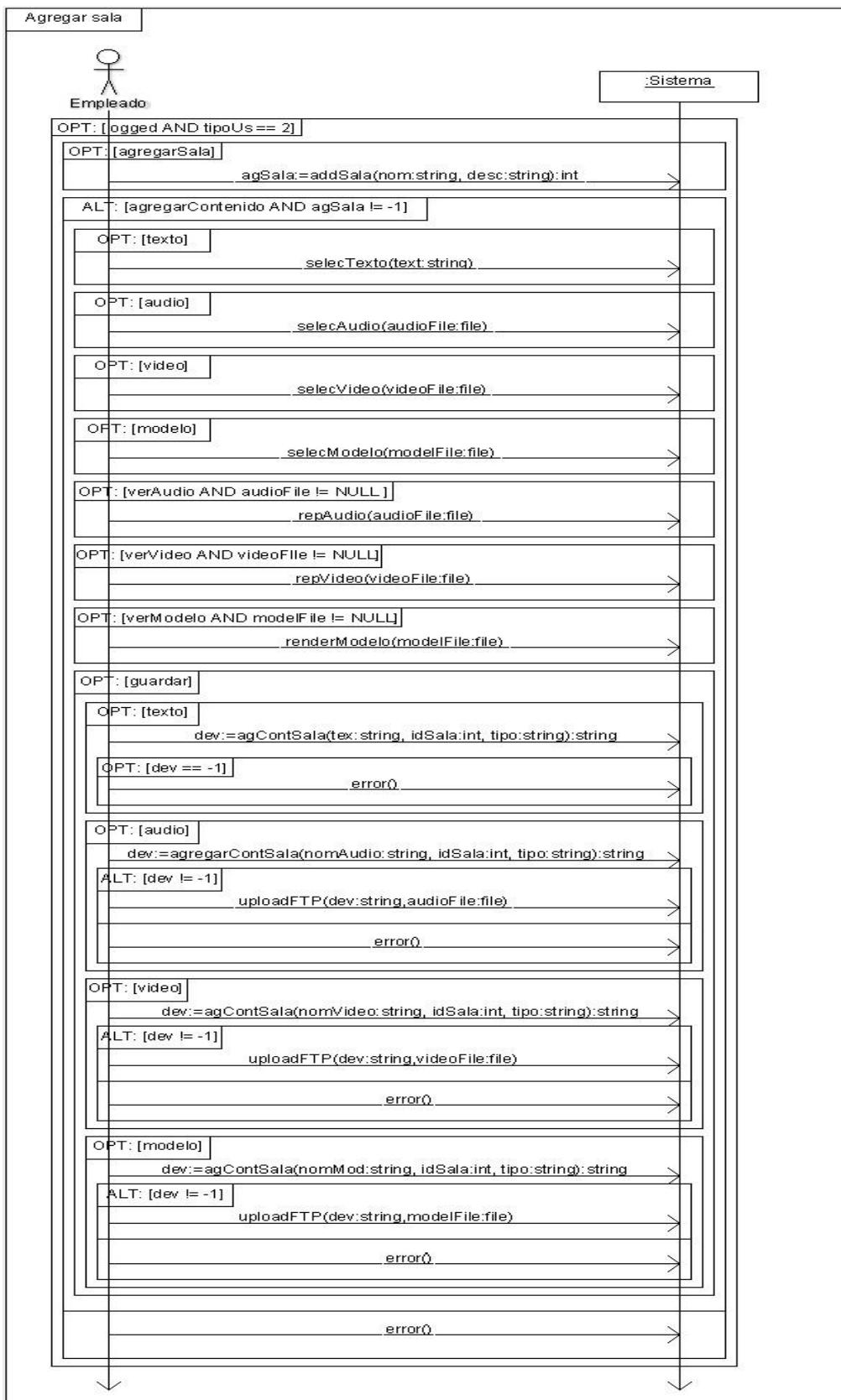
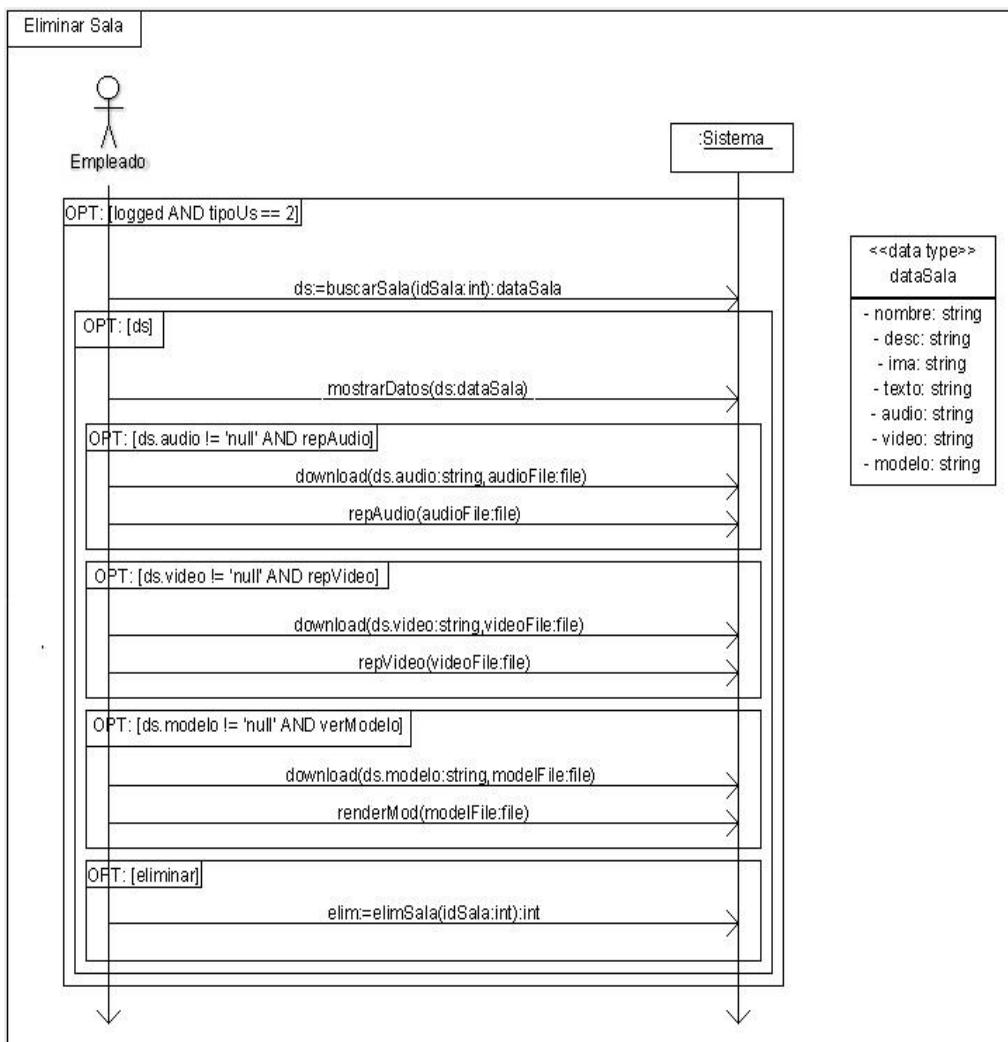


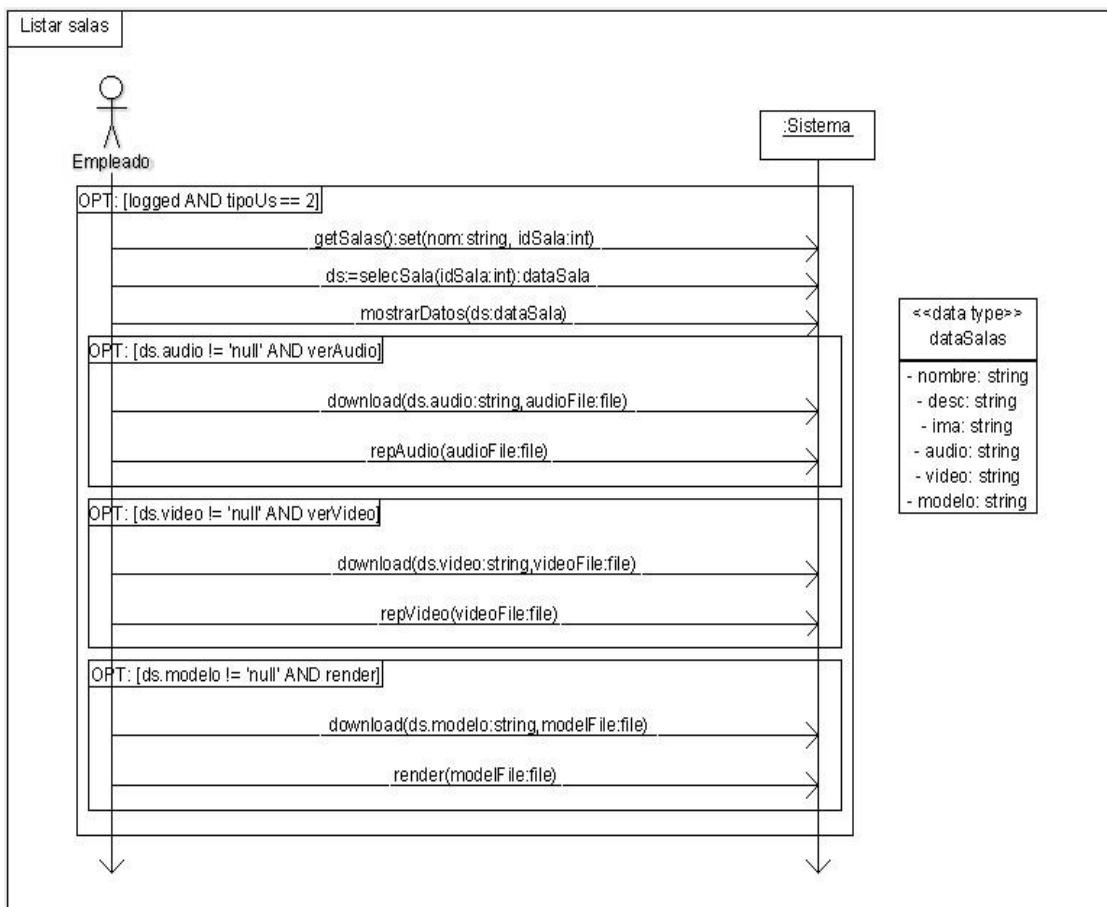
Figura 14

## 2.2.15. Eliminar sala.



**Figura 15**

## 2.2.16. Listar salas.



**Figura 16**

## 2.2.17. Modificar sala.

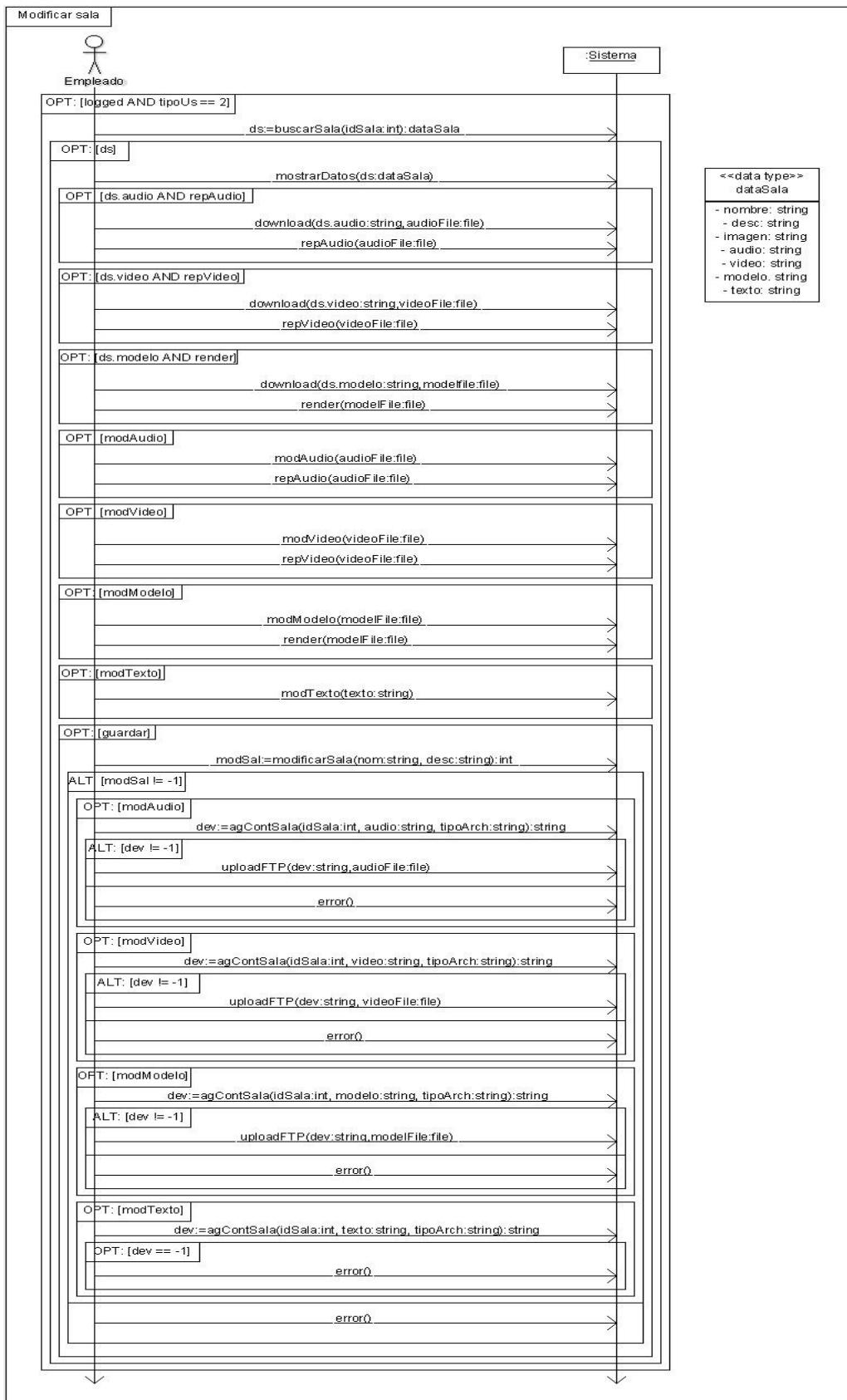


Figura 17

## 2.2.18. Agregar obra.

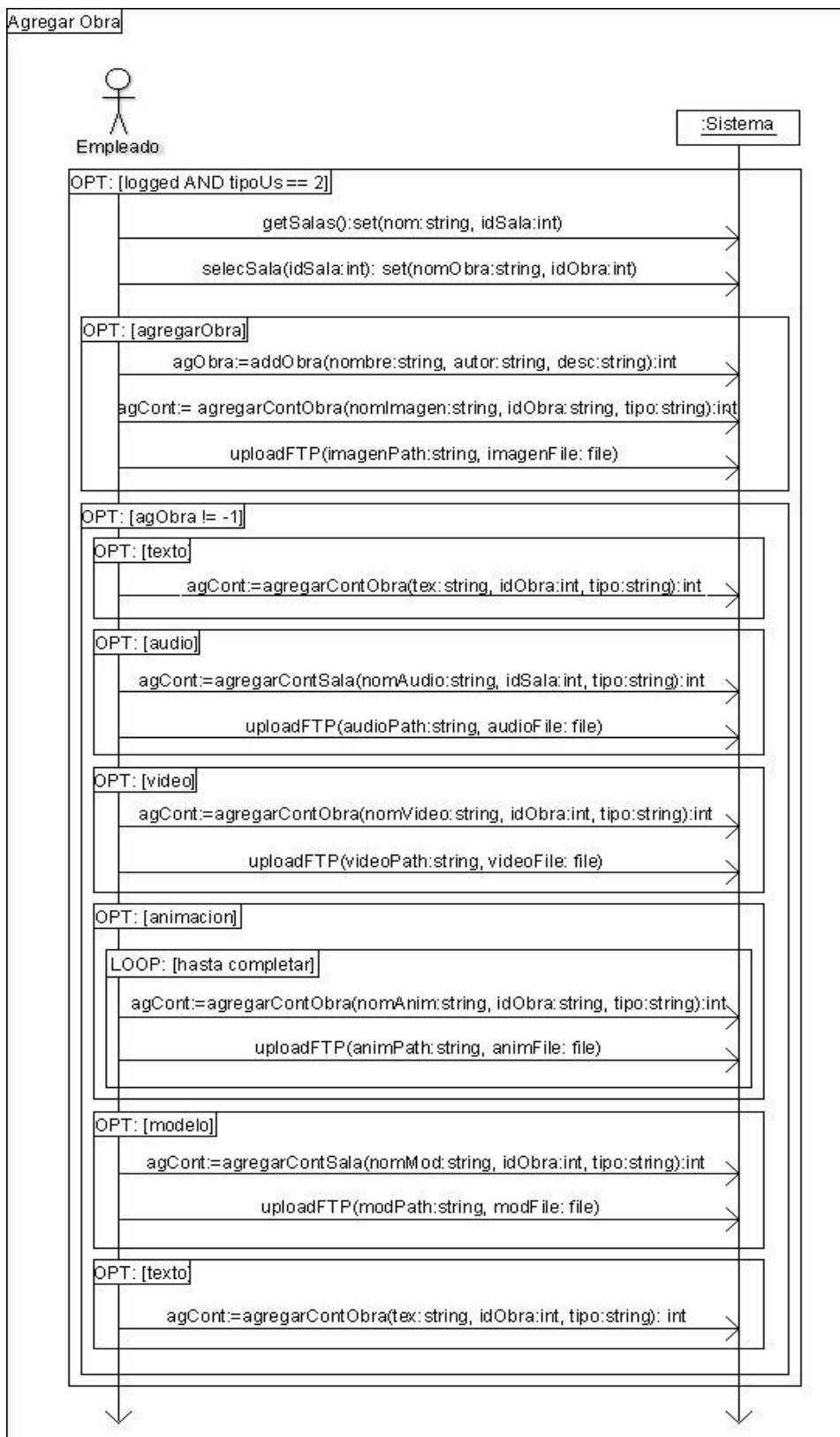


Figura 18

## 2.2.19. Eliminar obra.

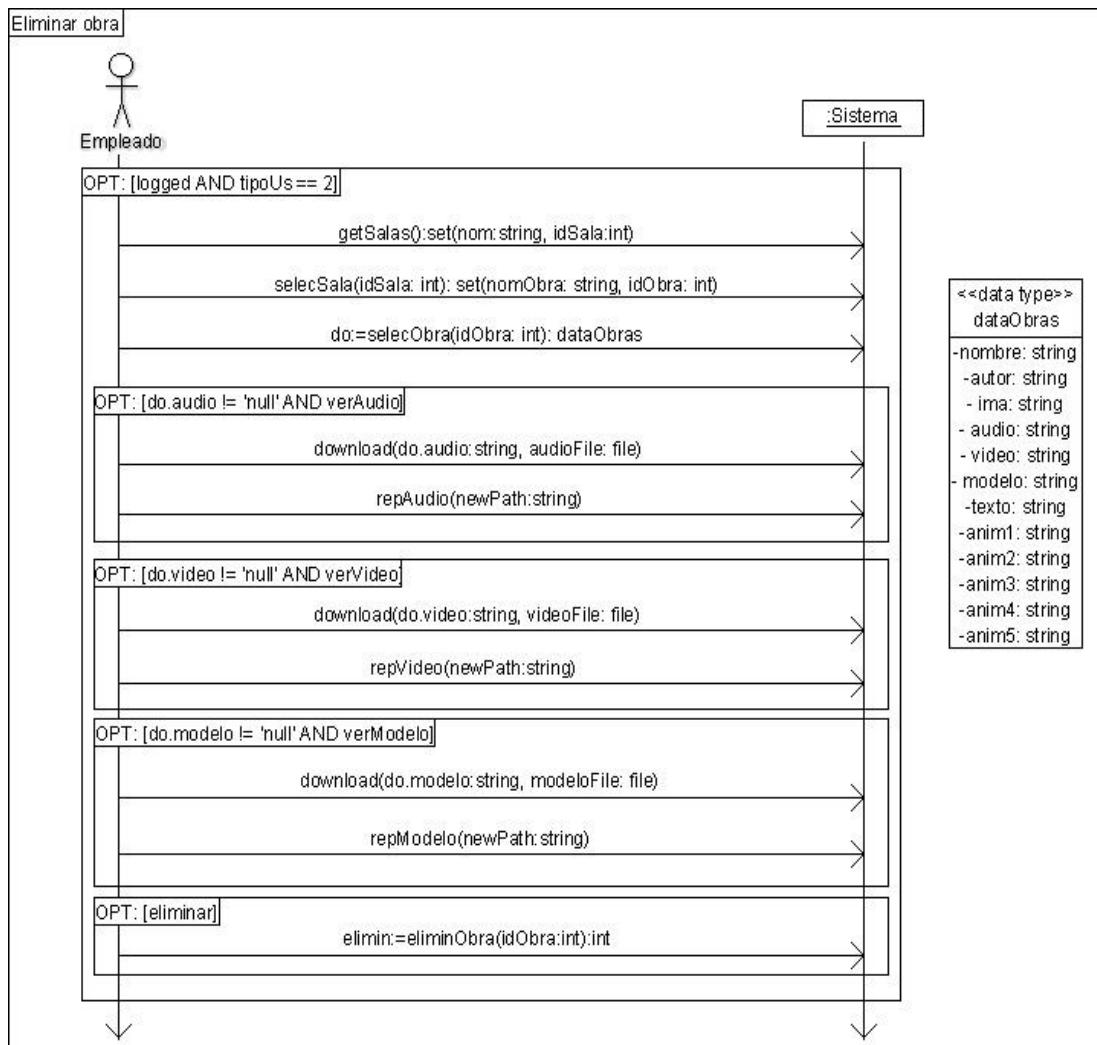


Figura 19

## 2.2.20. Listar obras.

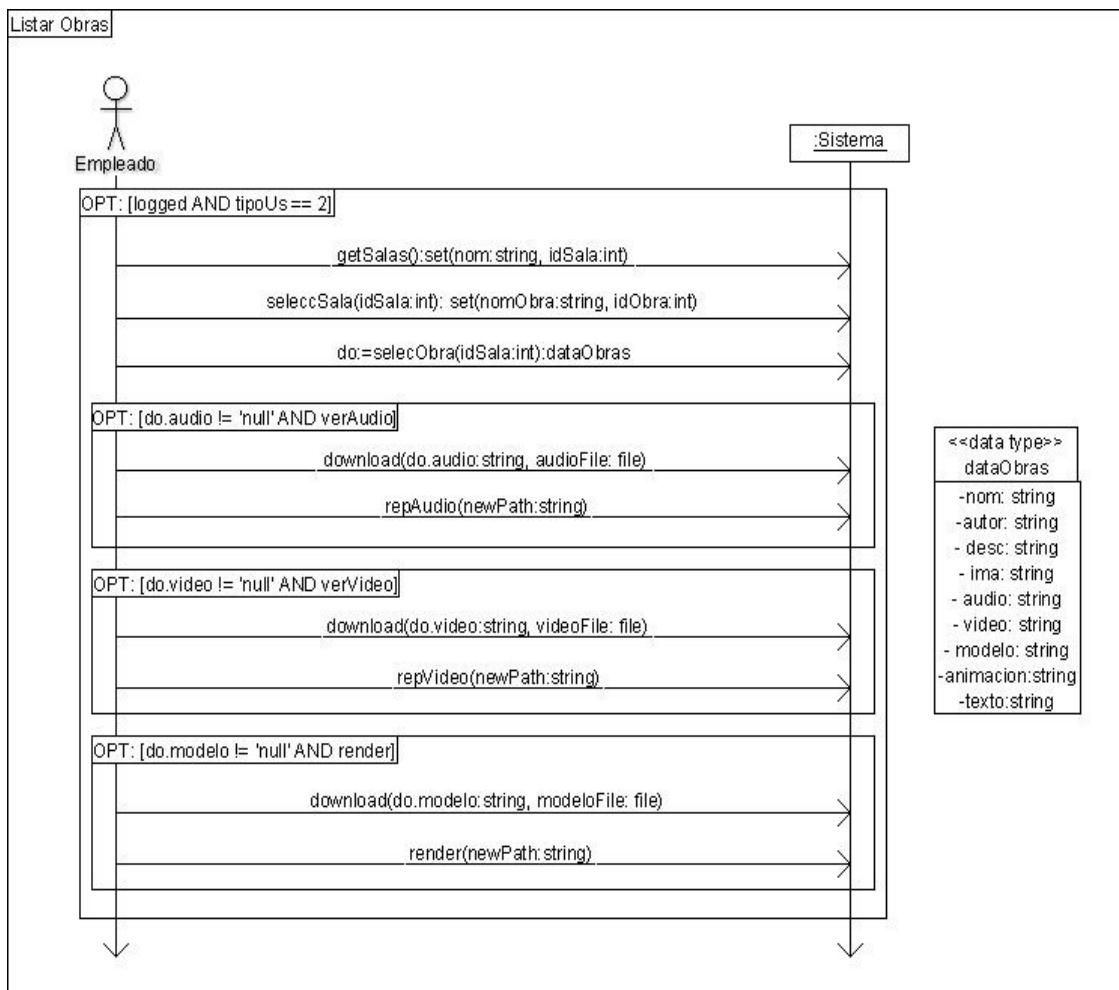


Figura 20

## 2.2.21. Modificar obra.

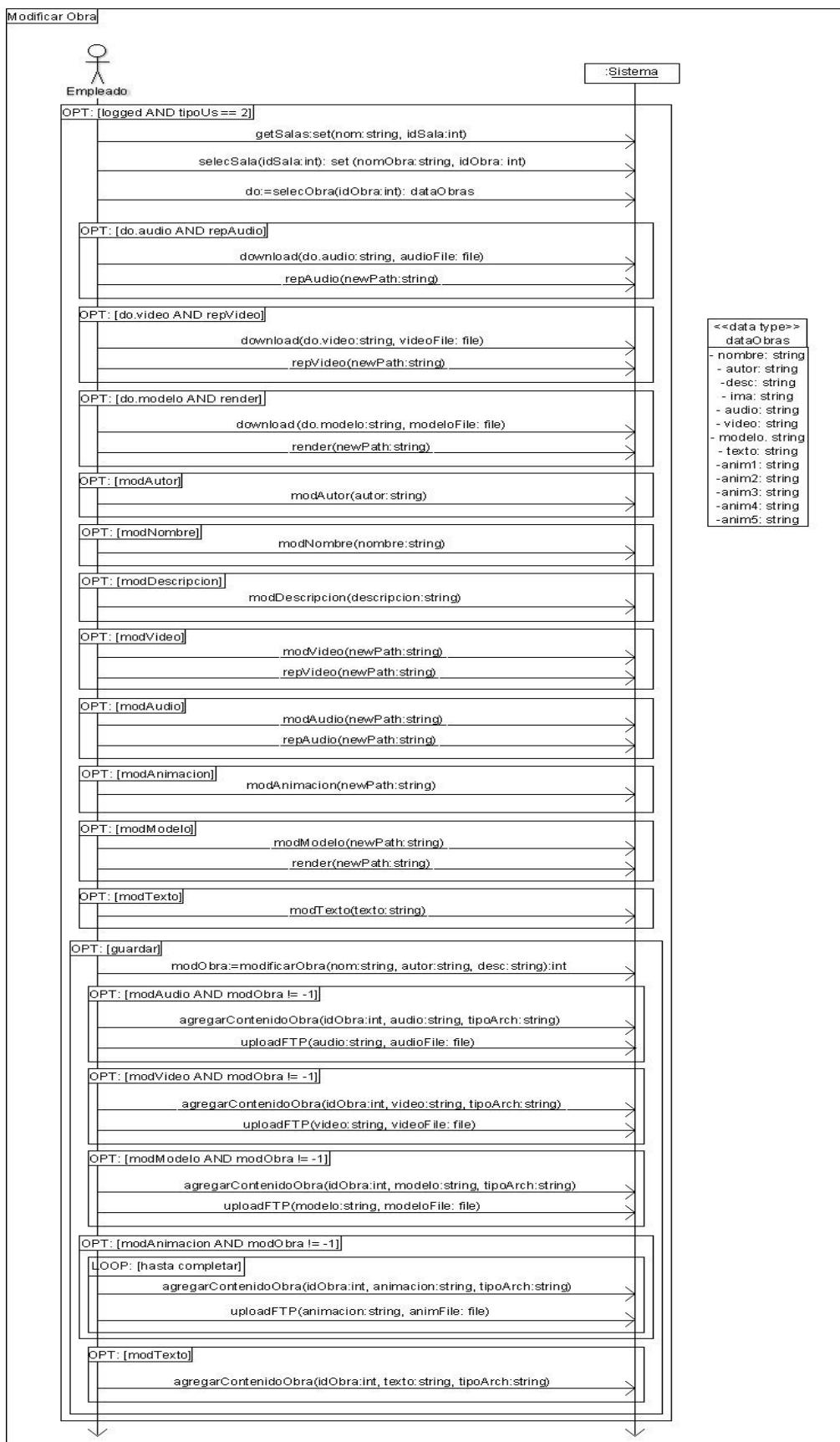


Figura 21

## 2.2.22. Agregar zona de interés.

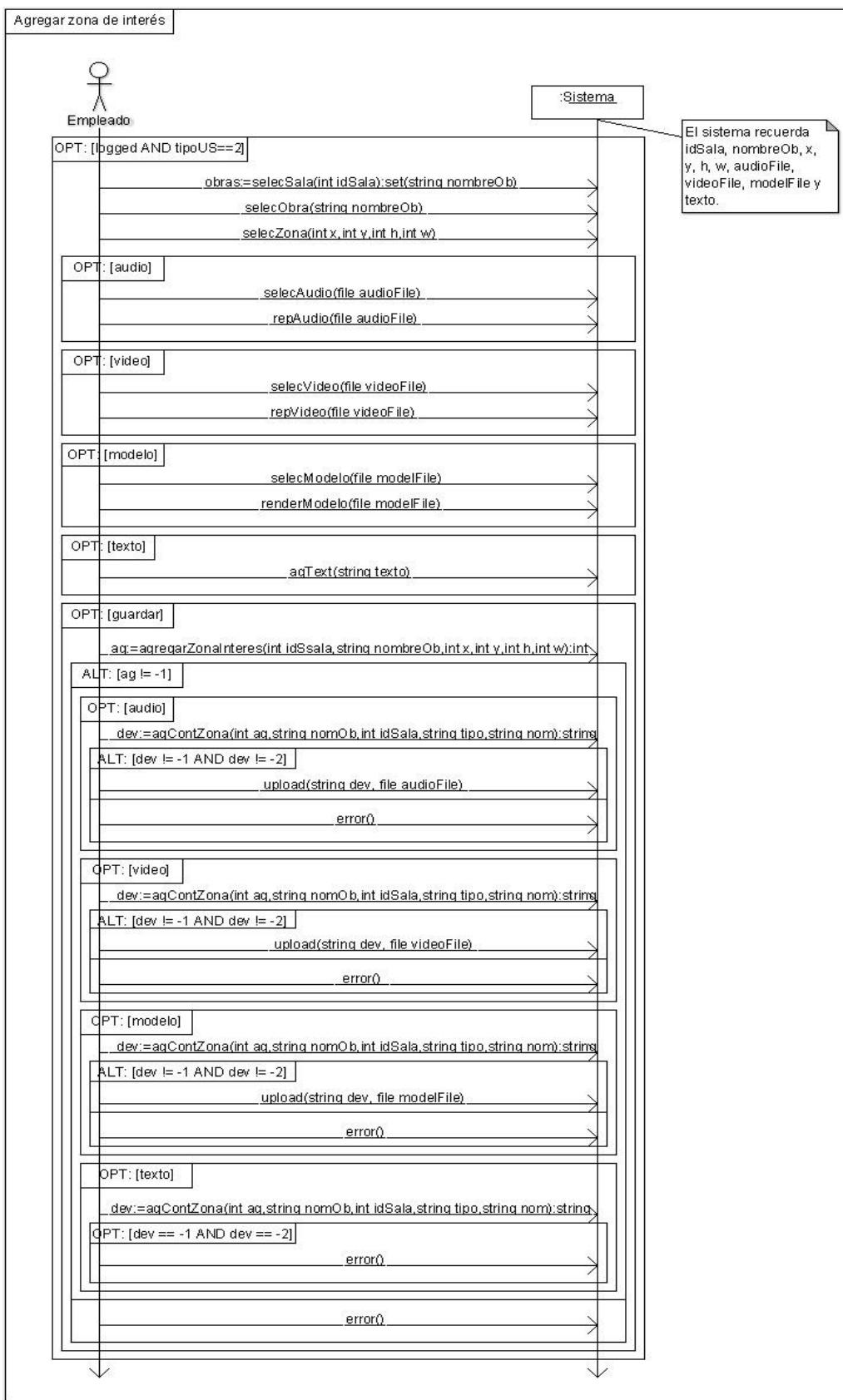


Figura 22

### 2.2.23. Eliminar zona de interés.

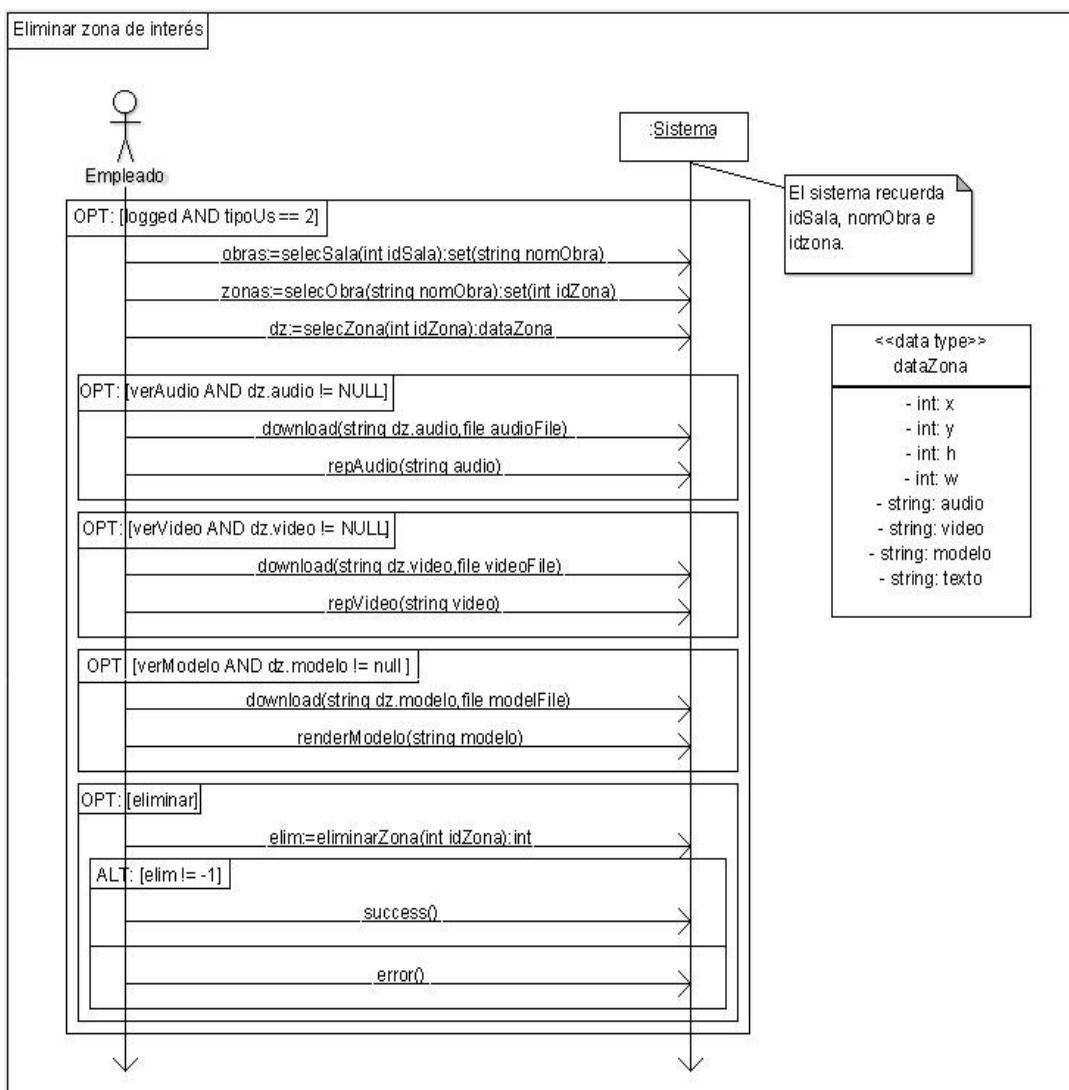


Figura 23

## 2.2.24. Modificar zona de interés.

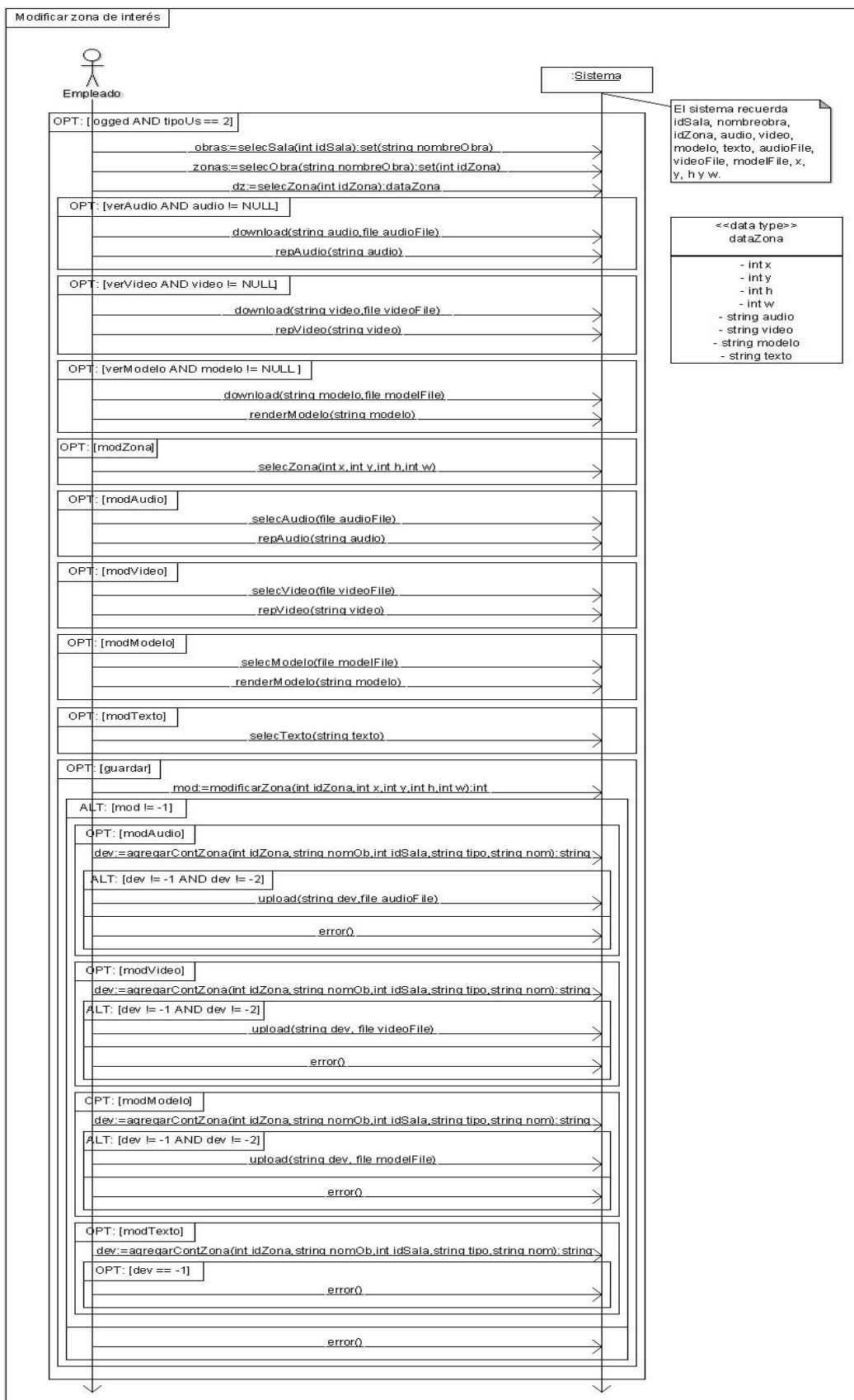


Figura 24

## 2.2.25. Cerrar sesión.

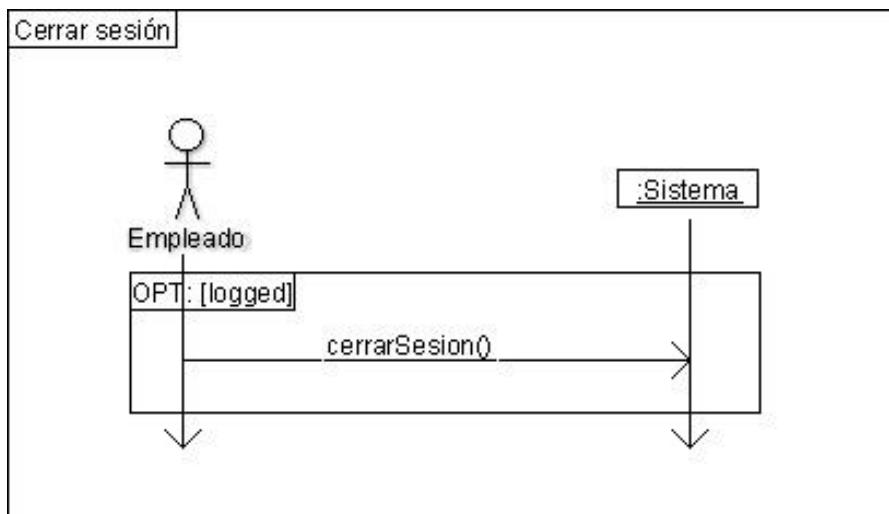


Figura 25

## **2.3. Contratos.**

**Aplicación móvil.**

**Modo manual.**

### **2.3.1. Listar salas.**

<b>Operación</b>	<b>getSalas():set(ds:dataSalas)</b>
<b>Entrada</b>	
<b>Salida</b>	Set del data type dataSala.
<b>Descripción</b>	Se obtienen el identificador, la imagen, el nombre y la descripción de todas las salas existentes.
<b>Precondiciones</b>	Pre1: Existe en el sistema una sala con id = idSala.
<b>Postcondiciones</b>	

<b>Operación</b>	<b>selección(idSala:int)</b>
<b>Entrada</b>	Identificador de una sala.
<b>Salida</b>	
<b>Descripción</b>	Se selecciona una sala para que el sistema recuerde el idSala de la sala seleccionada.
<b>Precondiciones</b>	Pre1: Existe una lista de salas. Pre2: Existe una sala con id = idSala.
<b>Postcondiciones</b>	Post1: El sistema recuerda el idSala de la sala seleccionada.

### **2.3.2. Listar obras.**

<b>Operación</b>	getObrasSala(idSala:int):set(do:dataObra)
<b>Entrada</b>	Identificador de una sala.
<b>Salida</b>	Set del data type dataObra.
<b>Descripción</b>	Se obtienen el identificador, la imagen, el nombre y la descripción de todas las obras existentes en la sala con id = idSala.
<b>Precondiciones</b>	Pre1: Existe en el sistema una sala con id = idSala.
<b>Postcondiciones</b>	

<b>Operación</b>	seleccion(nombre:string)
<b>Entrada</b>	Identificador de una sala.
<b>Salida</b>	
<b>Descripción</b>	Se selecciona una obra para que el sistema recuerde el nombre de la obra seleccionada.
<b>Precondiciones</b>	Pre1: Existe una lista de obras. Pre2: Existe en el sistema una obra con nombre = nombre.
<b>Postcondiciones</b>	Post1: El sistema recuerda el nombre de la obra seleccionada.

## Modo automático.

### 2.3.3. Identificar sala.

<b>Operación</b>	<b>idSala:=obtenerIdSalaDeQR():int</b>
<b>Entrada</b>	
<b>Salida</b>	Integer.
<b>Descripción</b>	Se obtiene el identificador de la sala a través del dispositivo móvil enfocando el código QR de dicha sala.
<b>Precondiciones</b>	Pre1: Existe en el sistema una sala con id = idSala.
<b>Postcondiciones</b>	

<b>Operación</b>	<b>ds:=obtenerDatosSala(idSala:int):dataSala</b>
<b>Entrada</b>	Identificador de Sala.
<b>Salida</b>	Data type dataSala, que tiene imagen, nombre y descripción.
<b>Descripción</b>	Se obtienen los datos de la sala con id = idSala, en un data type llamado dataSala con los datos imagen, nombre y descripcion correspondientes y -1 si la sala no existe.
<b>Precondiciones</b>	Pre1: Existe en el sistema una sala con id = idSala.
<b>Postcondiciones</b>	Post1: Se identificó la sala a la que se ingresa.

<b>Operación</b>	<b>mostrarDatos(ds:dataSala)</b>
<b>Entrada</b>	Data type dataSala.
<b>Salida</b>	
<b>Descripción</b>	Se muestran los datos de dataSala (idSala, imagen, nombre y descripción).
<b>Precondiciones</b>	Pre1: Existe en el sistema una sala con id = idSala.
<b>Postcondiciones</b>	

<b>Operación</b>	<b>mostrarMensaje()</b>
<b>Entrada</b>	
<b>Salida</b>	
<b>Descripción</b>	Se muestra un mensaje de error.
<b>Precondiciones</b>	
<b>Postcondiciones</b>	

### 2.3.4. Identificar obra.

<b>Operación</b>	<b>foto:=sacarFoto():file</b>
<b>Entrada</b>	
<b>Salida</b>	File.
<b>Descripción</b>	Se saca una foto y se obtiene el archivo foto.
<b>Precondiciones</b>	
<b>Postcondiciones</b>	Post1: Existe ahora una foto nueva.

<b>Operación</b>	<b>upload(rutaFTP:string, foto:file)</b>
<b>Entrada</b>	La ruta (string) para subir archivos al servidor y el archivo a subir (file).
<b>Salida</b>	
<b>Descripción</b>	Se sube la foto al servidor.
<b>Precondiciones</b>	Pre1: Existe una rutaFTP a donde enviar la foto. Pre2: Existe un archivo foto para subir.
<b>Postcondiciones</b>	Post1: Existe ahora una foto nueva en el servidor.

<b>Operación</b>	<b>nombreObra:=obtenerIdObra(nombreFoto:string,idSala:int):string</b>
<b>Entrada</b>	El nombre (string) de la foto subida y el idSala (int) en la que se encuentra.
<b>Salida</b>	String, el nombre de la obra identificada.
<b>Descripción</b>	Se envían el nombre de la foto subida al servidor previamente y el id de la sala en la que se encuentra para identificar la obra mediante comparación de imágenes de la sala.
<b>Precondiciones</b>	Pre1: Existe en el sistema una foto con nombre = nombreFoto. Pre2: Existe en el sistema una sala con id = idSala.
<b>Postcondiciones</b>	Post1: Se identificó una obra.

### 2.3.5. Obtener datos y contenidos.

<b>Operación</b>	<b>cont:=getContenidos(idObra:int):contenidos</b>
<b>Entrada</b>	Identificador de Obra.
<b>Salida</b>	Data type contenidos.
<b>Descripción</b>	Se obtienen todos los datos de la obra con id = idObra. Los datos obtenidos en el data type, son: nombre, autor, imagen, descripción, texto, audio, video, modelo y los modelos 3D que formarán la animación.
<b>Precondiciones</b>	Pre1: Existe en el sistema una obra con id = IdObra. Pre2: La obra debe estar asociada a una sala del museo. Pre3: La obra de id = IdObra debe tener al menos 1 tipo de contenido.
<b>Postcondiciones</b>	Post1: Se obtienen los contenidos de la obra de id = IdObra

<b>Operación</b>	<b>download(rutaFTP:string, archivo:file)</b>
<b>Entrada</b>	String de la ruta en la que se encuentra el archivo a bajar, y el file donde se ubicará el archivo a bajar.
<b>Descripción</b>	Se baja por FTP el archivo que se encuentra en la rutaFTP al file archivo.
<b>Precondiciones</b>	Pre1: Existe en el sistema un archivo con ruta = rutaFTP. Pre2: Existe un archivo = archivo.
<b>Postcondiciones</b>	Post1: Se obtiene el archivo.

### 2.3.6. Realidad aumentada.

<b>Operación</b>	<b>AugRealVideo(video:string)</b>
<b>Entrada</b>	String de la ruta en la que se encuentra el archivo que realizará la realidad aumentada.
<b>Salida</b>	
<b>Descripción</b>	Se reproduce el video que se encuentra en la ruta = video.
<b>Precondiciones</b>	Pre1: Existe en el sistema un video con ruta = video.
<b>Postcondiciones</b>	Post1: Se reproduce la realidad aumentada.

<b>Operación</b>	<b>AugRealMod(modelo:string)</b>
<b>Entrada</b>	String de la ruta en la que se encuentra el archivo que realizará la realidad aumentada.
<b>Salida</b>	
<b>Descripción</b>	Se reproduce el render que se encuentra en la ruta = modelo.
<b>Precondiciones</b>	Pre1: Existe en el sistema un modelo 3D con ruta = modelo.
<b>Postcondiciones</b>	Post1: Se reproduce la realidad aumentada.

<b>Operación</b>	<b>AugRealAnim(anim:string,anim2:string,anim3:string,anim4:string,anim5:string)</b>
<b>Entrada</b>	String de las rutas en las que se encuentran los archivos que realizaran la animación en la realidad aumentada.
<b>Salida</b>	
<b>Descripción</b>	Se reproduce la animación con los archivos que se encuentran en sus respectivas rutas = anim3D.
<b>Precondiciones</b>	Pre1: Existe en el sistema un modelo 3D con ruta = anim. Pre2: Existe en el sistema un modelo 3D con ruta = anim2. Pre3: Existe en el sistema un modelo 3D con ruta = anim3. Pre4: Existe en el sistema un modelo 3D con ruta = anim4. Pre5: Existe en el sistema un modelo 3D con ruta = anim5.
<b>Postcondiciones</b>	Post1: Se reproduce la realidad aumentada.

## **Aplicación de escritorio.**

### **Modo administrador.**

#### **2.3.7. Iniciar sesión.**

<b>Operación</b>	<code>logged:= iniSesion(usu:string, pass:string): integer</code>
<b>Entrada</b>	Nombre de usuario “usu”, contraseña de usuario “pass”
<b>Salida</b>	Integer: 1 en caso que se haya logueado correctamente, -1 en caso contrario
<b>Descripción</b>	Permite loguearse en el sistema a través del nombre de usuario y contraseña
<b>Precondiciones</b>	Pre1: Existe en el sistema el usuario con nick “usu” y contraseña “pass”.
<b>Postcondiciones</b>	Post1: Se logueó el usuario en el sistema.

#### **2.3.8. Agregar usuario.**

<b>Operación</b>	<code>agregado:= agregarUsuario(nom:string, ape:string, ci:string, email:string, tipoUs:int, nick:string, pass:string): int</code>
<b>Entrada</b>	Datos del usuario a dar de alta en el sistema.
<b>Salida</b>	Integer: 1 en caso que se haya agregado correctamente, -1 en caso contrario
<b>Descripción</b>	Permite agregar un nuevo usuario en el sistema, proporcionando los datos necesarios.
<b>Precondiciones</b>	Pre1: No existe en el sistema el usuario con nick “nick”.
<b>Postcondiciones</b>	Post1: Se dio de alta el nuevo usuario.

### **2.3.9. Modificar usuario.**

<b>Operación</b>	<b>listarUsuarios(): set(Usu)</b>
<b>Salida</b>	Lista de datos de usuarios registrados.
<b>Descripción</b>	Permite obtener una lista con los datos de los usuarios registrados en el sistema.
<b>Precondiciones</b>	Pre 1: Existe al menos un usuario registrado.
<b>Postcondiciones</b>	Post 1: Se obtuvo la lista de datos de usuarios registrados.

<b>Operación</b>	<b>mod:= modificarUsuario(nom:string, ap:string, ci:string, email:string, tipos:int, nick:string, pass:string): int</b>
<b>Entrada</b>	Datos a modificar en el usuario.
<b>Salida</b>	Integer: 1 en caso que se haya modificado correctamente, -1 en caso contrario.
<b>Descripción</b>	Permite modificar los datos del usuario con nick “nick”, registrado en el sistema.
<b>Precondiciones</b>	Pre 1: Existe el usuario de nick “nick” registrado en el sistema.
<b>Postcondiciones</b>	Post 1: Se modificaron los datos suministrados para el usuario.

### 2.3.10. Listar usuarios.

<b>Operación</b>	<b>du:= seleccionUsuario(nickUsu:string): DataUsu</b>
<b>Entrada</b>	Nick del usuario a consultar los datos.
<b>Salida</b>	Datos del usuario seleccionado.
<b>Descripción</b>	Permite obtener los datos pertenecientes a un usuario seleccionado.
<b>Precondiciones</b>	Pre 1: Existe en el sistema el usuario con Nick “nickUsu”.
<b>Postcondiciones</b>	Post 1: Se obtuvo los datos pertenecientes al usuario seleccionado.

<b>Operación</b>	<b>getUsuariosAdm(): set(nickUsu:string)</b>
<b>Salida</b>	Lista de nicks de usuarios de tipo Administrador.
<b>Descripción</b>	Permite obtener una lista de los nicks de los usuarios Administradores registrados en el sistema.
<b>Precondiciones</b>	Pre 1: Existe al menos un usuario de tipo Administrador registrado.
<b>Postcondiciones</b>	Post 1: Se obtuvo la lista de nicks de usuarios Administradores registrados.

<b>Operación</b>	<b>getUsuariosEmp(): set(nickUsu:string)</b>
<b>Salida</b>	Lista de nicks de usuarios de tipo Empleado.
<b>Descripción</b>	Permite obtener una lista de los nicks de los usuarios Empleado registrados en el sistema.
<b>Precondiciones</b>	Pre 1: Existe al menos un usuario de tipo Empleado registrado.
<b>Postcondiciones</b>	Post 1: Se obtuvo la lista de nicks de usuarios Empleado registrados.

### 2.3.11. Eliminar usuario.

<b>Operación</b>	<code>getUsuarios(): set(nickUsu: string)</code>
<b>Salida</b>	Lista de nicks de usuarios.
<b>Descripción</b>	Permite obtener una lista de los nicks de los usuarios registrados en el sistema.
<b>Precondiciones</b>	Pre1: Existe al menos un usuario registrado.
<b>Postcondiciones</b>	Post1: Se obtuvo la lista de nicks de los usuarios registrados.

<b>Operación</b>	<code>elimin:= eliminUsu(nickUsu: string): integer</code>
<b>Entrada</b>	Nick del usuario a eliminar.
<b>Salida</b>	Integer: 1 en caso que se haya eliminado correctamente, -1 en caso contrario
<b>Descripción</b>	Permite eliminar un usuario en el sistema, proporcionando el nick del mismo.
<b>Precondiciones</b>	Pre1: Existe en el sistema el usuario con nick “nick”.
<b>Postcondiciones</b>	Post1: Se eliminó el usuario del sistema.

### 2.3.12. Cerrar sesión.

<b>Operación</b>	<code>CerrarSesion()</code>
<b>Descripción</b>	Permite cerrar la sesión del usuario actualmente logueado.
<b>Precondiciones</b>	Pre 1: Existe el usuario registrado en el sistema. Pre 2: El usuario está actualmente logueado.
<b>Postcondiciones</b>	Post 1: Se cerró la sesión de usuario previamente iniciada.

## **Modo empleado.**

### **2.3.14. Agregar sala.**

<b>Operación</b>	<b>ingresarSala(id: String, nombre: String, descr: String)</b>
<b>Entrada</b>	Datos de la sala a dar de alta
<b>Salida</b>	Boolean
<b>Descripción</b>	Agrega al sistema una nueva sala.
<b>Precondiciones</b>	Pre1: No existe una sala con idSala=id que la identifica.
<b>Postcondiciones</b>	Post1: Existe una sala con idSala=id que la identifica.

### **2.3.15. Eliminar sala.**

<b>Operación</b>	<b>ListarSalas(): set(DataSalas)</b>
<b>Entrada</b>	
<b>Salida</b>	Lista con datos de salas.
<b>Descripción</b>	Lista todas las salas existentes que no tienen ninguna obra asociadas en el sistema.
<b>Precondiciones</b>	Pre1: Existe al menos una sala en el sistema.
<b>Postcondiciones</b>	Post1: Existe una lista de salas con todas las obras existentes sin obras, con un id=idSala que identifica a cada una.

<b>Operación</b>	<b>EliminarSalaElegida(idSala: int): boolean</b>
<b>Entrada</b>	Id de sala a eliminar.
<b>Salida</b>	Boolean, indicando <i>true</i> si la sala fue eliminada del sistema.
<b>Descripción</b>	Elimina del sistema la sala con id=idSala, siempre y cuando la misma no tenga ninguna obra asociada.
<b>Precondiciones</b>	Pre1: Existe sala con id=idSala que identifica la sala y que no tiene ninguna obra asociada.
<b>Postcondiciones</b>	Post1: No existe sala con id=idSala, que identificaba la sala.

### **2.3.16. Listar salas.**

<b>Operación</b>	<b>ListarSalas(): salas: Set(DataSa)</b>
<b>Entrada</b>	
<b>Salida</b>	Lista de salas
<b>Descripción</b>	Lista las salas existentes en el sistema.
<b>Precondiciones</b>	Pre1: Existe una sala en el sistema.
<b>Postcondiciones</b>	Post1: El sistema lista todas las salas existentes en él.

<b>Operación</b>	<b>sal: IdentificarSala(IdSala: int): Sala</b>
<b>Entrada</b>	Identificador de sala.
<b>Salida</b>	Datos de la sala seleccionada.
<b>Descripción</b>	Retorna los datos correspondientes a la sala de id=IdSala.
<b>Precondiciones</b>	Pre1: Existe en el sistema una sala de id=IdSala.
<b>Postcondiciones</b>	Post1: El sistema retorna todos los datos de la sala con id=IdSala.

### **2.3.17. Modificar sala.**

<b>Operación</b>	<b>BuscarSala(IdSala: int)</b>
<b>Entrada</b>	Identificador de sala
<b>Salida</b>	Datos de la sala
<b>Descripción</b>	Busca la sala identificada con el id=idSala
<b>Precondiciones</b>	Pre1: Existe una sala con id=IdSala
<b>Postcondiciones</b>	Post1: La sala de id=IdSala es encontrada por el sistema y sus datos son devueltos.

<b>Operación</b>	<b>ModSala(idSala: int, nombre: string, descripcion: string)</b>
<b>Entrada</b>	Datos modificados de la sala
<b>Salida</b>	
<b>Descripción</b>	Modifica los datos de una sala
<b>Precondiciones</b>	Pre1: Existe la sala con id=IdSala que la identifica.
<b>Postcondiciones</b>	Post1: La sala de id=IdSala es modificada reemplazando los datos existentes por los datos nuevos.

### **2.3.18. Agregar obra.**

<b>Operación</b>	<b>agObra=addObra(nombreOb: string, autor: string, desc: string): int</b>
<b>Entrada</b>	Nombre de la Obra, autor y descripción de la obra.
<b>Salida</b>	Integer: 1 en caso que se haya agregado correctamente, -1 en caso contrario
<b>Descripción</b>	Permite agregar una obra en el sistema.
<b>Precondiciones</b>	Pre 1: Debe existir en el sistema al menos 1 sala. Pre 2: No debe existir en el sistema una obra con nombre = nombreOb.
<b>Postcondiciones</b>	Post 1: La obra de nombre “nombreOb” fue dada de alta.

### **2.3.19. Eliminar obra.**

<b>Operación</b>	<b>eliminarObraElegida(O: DataObra)</b>
<b>Entrada</b>	DataObra
<b>Salida</b>	Boolean
<b>Descripción</b>	Elimina del sistema la obra O.
<b>Precondiciones</b>	Pre1: Existe obra con id=idObra que identifica a O.
<b>Postcondiciones</b>	Post1: No existe obra con id=idObra, que identificaba a O.

### **2.3.20. Listar obras.**

<b>Operación</b>	<b>l: ListarObras(): set(DataListaObra)</b>
<b>Entrada</b>	
<b>Salida</b>	Lista de obras
<b>Descripción</b>	Lista todas las obras existentes en el sistema.
<b>Precondiciones</b>	Pre1: Existe al menos una obra en el sistema.
<b>Postcondiciones</b>	Post1: Se muestra una lista con todas las obras existentes en el sistema.

<b>Operación</b>	<b>SeleccionarObra(idObra: int): DataObra</b>
<b>Entrada</b>	Identificador de obra
<b>Salida</b>	Datos de la obra seleccionada
<b>Descripción</b>	Devuelve los datos correspondientes a la obra con id=idObra.
<b>Precondiciones</b>	Pre1: Existe la obra con id=idObra que la identifica.
<b>Postcondiciones</b>	Post1: Se muestran los datos de la obra identificada con id=idObra.

### 2.3.21. Modificar obra.

<b>Operación</b>	<b>BuscarObra(idObra: int): DataObra</b>
<b>Entrada</b>	Identificador de obra
<b>Salida</b>	Datos de la obra
<b>Descripción</b>	Devuelve los datos correspondientes a la obra con id=idObra.
<b>Precondiciones</b>	Pre1: Existe la obra con id=idObra que la identifica.
<b>Postcondiciones</b>	Post1: Se muestran los datos de la obra identificada con id=idObra.

<b>Operación</b>	<b>ModificarObra(nombre: string, descripcion: string, descriptor: string, contenido: contenido, idSala: int): boolean</b>
<b>Entrada</b>	Datos modificados de la obra
<b>Salida</b>	Boolean, devolviendo <i>true</i> si los datos son válidos
<b>Descripción</b>	Modifica los datos de la obra previamente seleccionada.
<b>Precondiciones</b>	Pre1: Existe la obra en el sistema.
<b>Postcondiciones</b>	Post1: Se modifican los datos de la obra.

### 2.3.22. Agregar zona de interés.

<b>Operación</b>	<code>selecZona(x:int,y:int,h:int,w:int)</code>
<b>Entrada</b>	X (x) e y (y) de la posición donde comienza el rectángulo. H (alto) y w (ancho) del rectángulo.
<b>Salida</b>	
<b>Descripción</b>	Guarda en memoria x, y, ancho y largo del rectángulo seleccionado por el usuario.
<b>Precondiciones</b>	
<b>Postcondiciones</b>	Post1: Existe un x, y, h y w que representan un rectángulo (zona de interés).

<b>Operación</b>	<code>ag:=agregarZonaInteres(idSala:int,nombreOb:string,x:int,y:int,h:int,w:int):int</code>
<b>Entrada</b>	Int idSala, a donde pertenece la obra a agregarle zona de interés. String del nombre de la obra a agregarle zona de interés. Int x, coordenada del rectángulo que representa la zona de interés. Int y, coordenada del rectángulo que representa la zona de interés. Int h, alto del rectángulo que representa la zona de interés. Int w, ancho del rectángulo que representa la zona de interés.
<b>Salida</b>	Int ag, devolviendo el id de la zona creada si se agregó la zona de interés correctamente y -1 si no.
<b>Descripción</b>	Agrega una nueva zona de interés a la tabla zonas de la base de datos, con su sala y obra correspondientes, y los datos consecuentes a la misma (x, y, alto y ancho).
<b>Precondiciones</b>	Pre1: Existe una sala con id = idSala. Pre2: Existe una obra con nombre = nombreOb.
<b>Postcondiciones</b>	Post1: Existe una zona de interés con id = ag.

<b>Operación</b>	<b>dev:=agContZona(idZ:int,nomOb:string,idSala:int,tipArch :string,nomArch:string):string</b>
<b>Entrada</b>	Int idZ, siendo el id de la zona a agregarle contenido. String nomOb, de la obra a la que pertenece la zona de interés. Int idSala, de la sala a la que pertenece la obra. String tipoArch, el tipo del archivo a agregar a la zona. String nomArch, el nombre del archivo o texto a subir.
<b>Salida</b>	String dev, siendo -1 o -2 en caso de fallas, y la ruta FTP a donde subir el archivo en caso de que no falle.
<b>Descripción</b>	Agrega un contenido a la zona de interés indicada.
<b>Precondiciones</b>	Pre1: Existe una sala con id = idSala. Pre2: Existe una obra con nombre = nomOb. Pre3: Existe una zona de interés con id = idZ.
<b>Postcondiciones</b>	Post1: Existe un contenido de tipo = tipoArch en la zona de interés con id = idZ.

### 2.3.23. Eliminar zona de interés.

<b>Operación</b>	<b>dz:=selecZona(idZona:int):dataZona</b>
<b>Entrada</b>	Int idZona, siendo idZona el id de la zona seleccionada.
<b>Salida</b>	dataZona dz, siendo dataZona un data type con los datos x, y, alto, ancho, audio, video, modelo y texto.
<b>Descripción</b>	Guarda en memoria una zona de interés y muestra sus datos.
<b>Precondiciones</b>	Pre1: Existe una zona de interés con id = idZona.
<b>Postcondiciones</b>	

<b>Operación</b>	<b>elim:=eliminarZona(idZona:int):int</b>
<b>Entrada</b>	Int idZona, siendo idZona el id de la zona a eliminar.
<b>Salida</b>	Int elim, si la eliminación tuvo éxito devuelve 1 y si no -1.
<b>Descripción</b>	Elimina la zona de interés con id = idZona y sus contenidos.
<b>Precondiciones</b>	Pre1: Existe una zona de interés con id = idZona.
<b>Postcondiciones</b>	Post1: No existe una zona de interés con id = idZona.

### 2.3.24. Modificar zona de interés.

<b>Operación</b>	<code>mod:=modificarZona(idZona:int,x:int,y:int,h:int,w:int):int</code>
<b>Entrada</b>	Int idZona, siendo idZona el id de la zona a modificar. Int x, el nuevo valor de x. Int y, el nuevo valor de y. Int h, el nuevo valor de h. Int w, el nuevo valor de w.
<b>Salida</b>	Int mod, si la modificación tuvo éxito devuelve 1 y si no -1.
<b>Descripción</b>	Modifica la zona de interés con id = idZona.
<b>Precondiciones</b>	Pre1: Existe una zona de interés con id = idZona.
<b>Postcondiciones</b>	Post1: La zona de interés con id = idZona tiene ahora los valores x, y, h y w.

### 2.3.25. Cerrar sesión.

<b>Operación</b>	<code>CerrarSesion()</code>
<b>Descripción</b>	Cierra la sesión iniciada por un usuario.
<b>Precondiciones</b>	Pre1: Usuario debe estar logueado en el sistema.
<b>Postcondiciones</b>	Post1: El sistema queda libre para un nuevo inicio de sesión.

## 2.4. Vistas.

### Aplicación de escritorio.

#### 2.4.1. Menú principal – Inicio sesión.



Figura 26

## Menú administrador.

### 2.4.2. Menú principal.



Figura 27

### 2.4.3. Menú usuarios.



Figura 28

#### 2.4.4. Menú sesión.



Figura 29

#### 2.4.5. Menú ayuda.



Figura 30

## 2.4.6. Agregar usuarios.

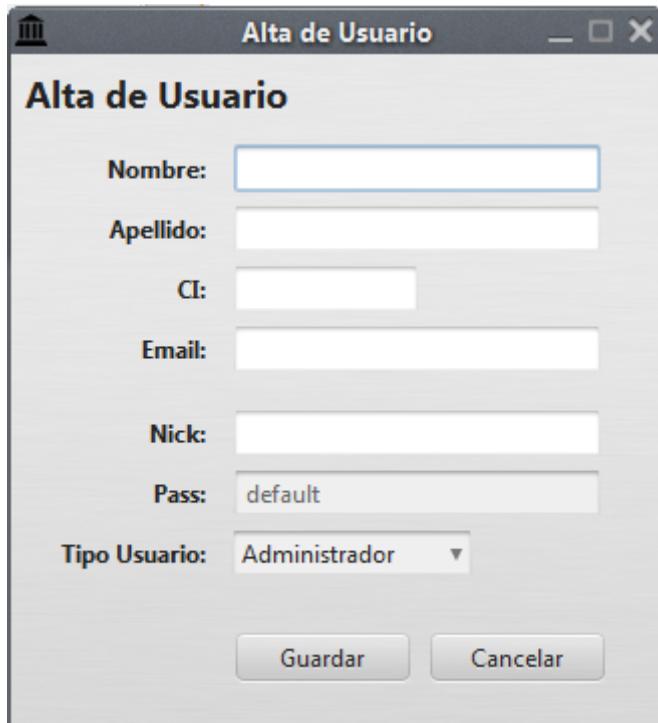


Figura 31

## Menú empleado.

### 2.4.7. Menú principal.



Figura 32

#### 2.4.8. Menú usuarios.



Figura 33

#### 2.4.9. Menú salas.



Figura 34

## 2.4.10. Menú obras.



Figura 35

## 2.4.11. Menú zona de interés.



Figura 36

## 2.4.12. Menú sesión.



Figura 37

## 2.4.13. Menú ayuda.



Figura 38

## 2.4.14. Agregar salas.

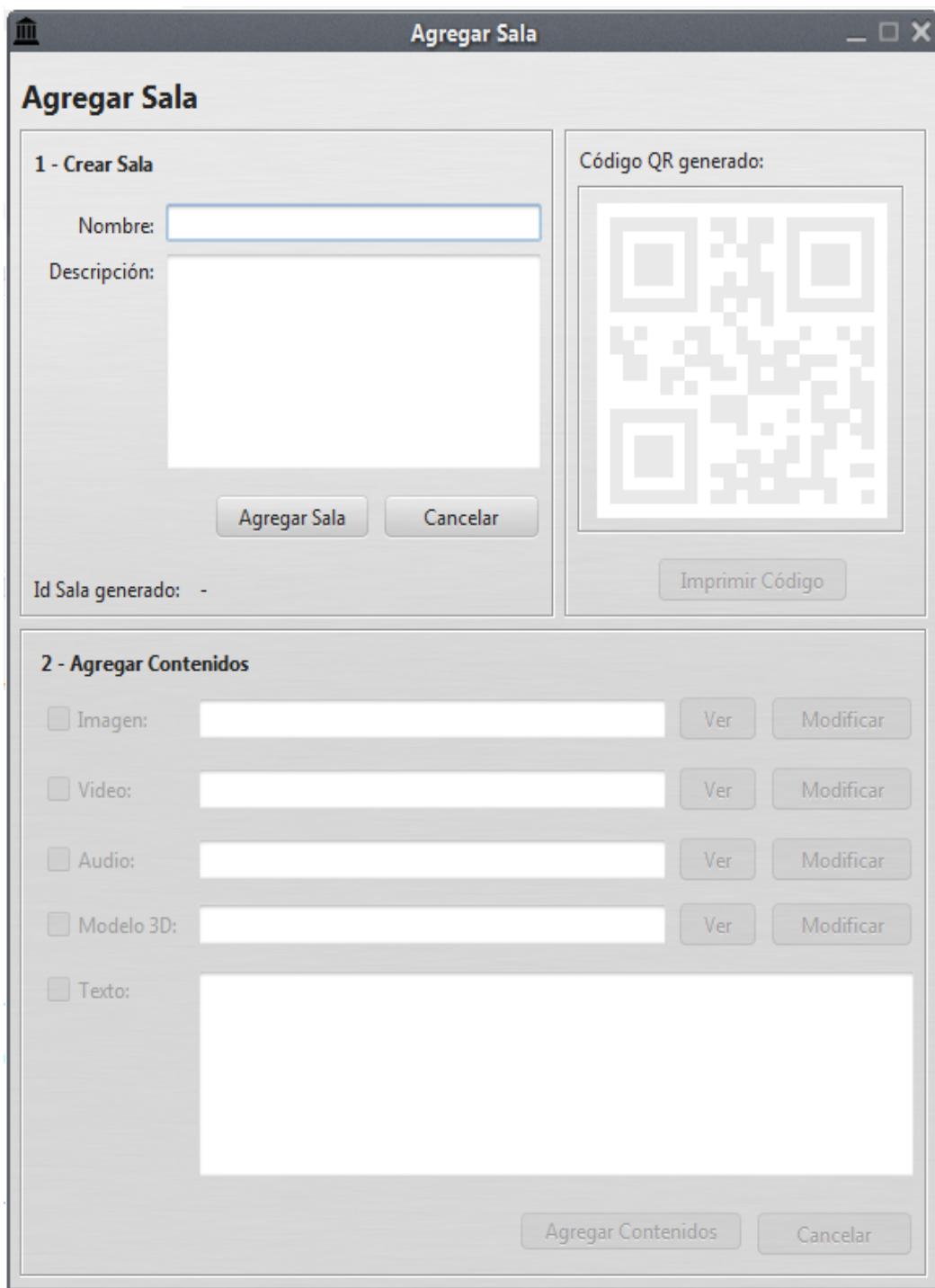


Figura 39

## 2.4.15. Agregar obras.



Figura 40

**Agregar Obra**

### Agregar Obra (2)

2- Ingresá los datos requeridos sobre la Obra:

Autor:

Nombre:

Imagen:  Ver Modificar

Descripción:

3- Seleccioná el tipo de Contenido a agregar:

Video:  Ver Modificar

Audio:  Ver Modificar

Animación:  Modificar

Modelo 3D:  Ver Modificar

Texto:  Modificar

Guardar Cancelar

Figura 41

## 2.4.16. Agregar zona de interés.

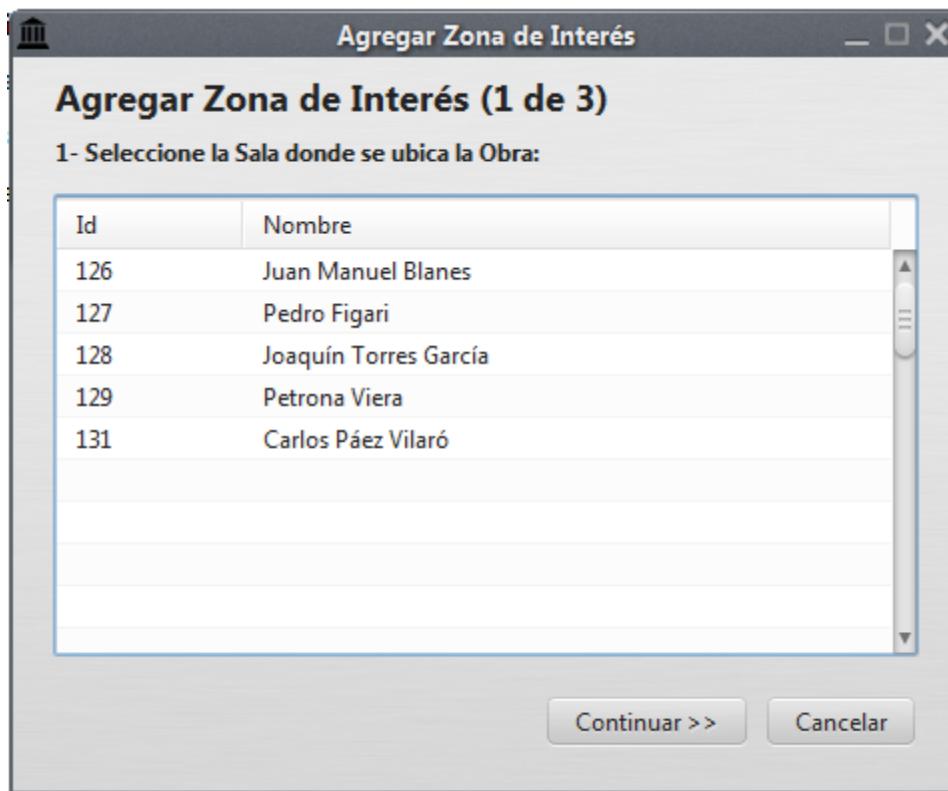


Figura 42

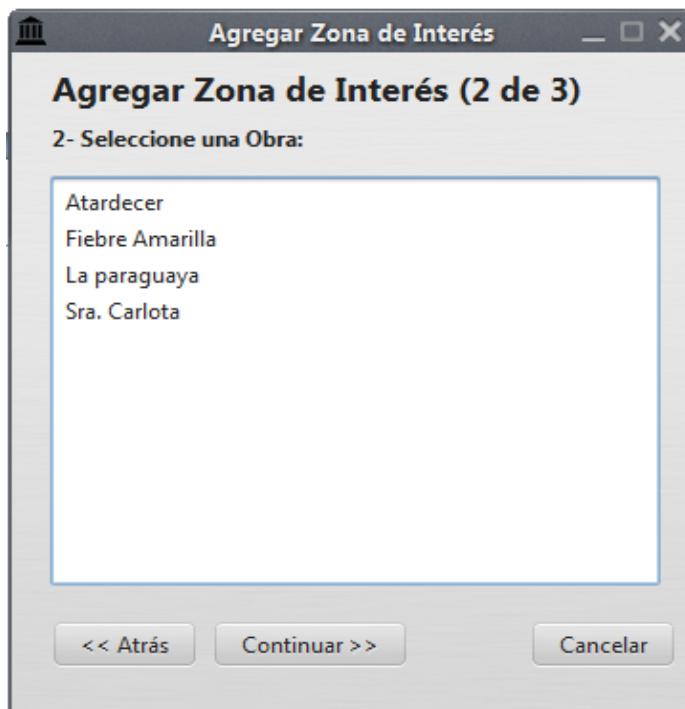


Figura 43

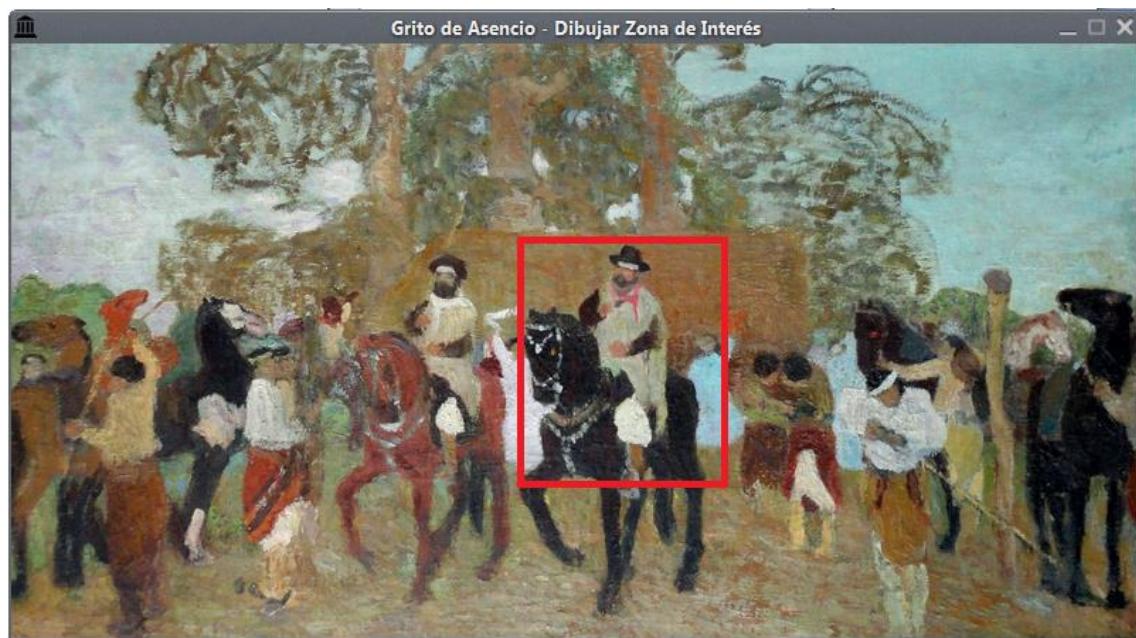


Figura 44

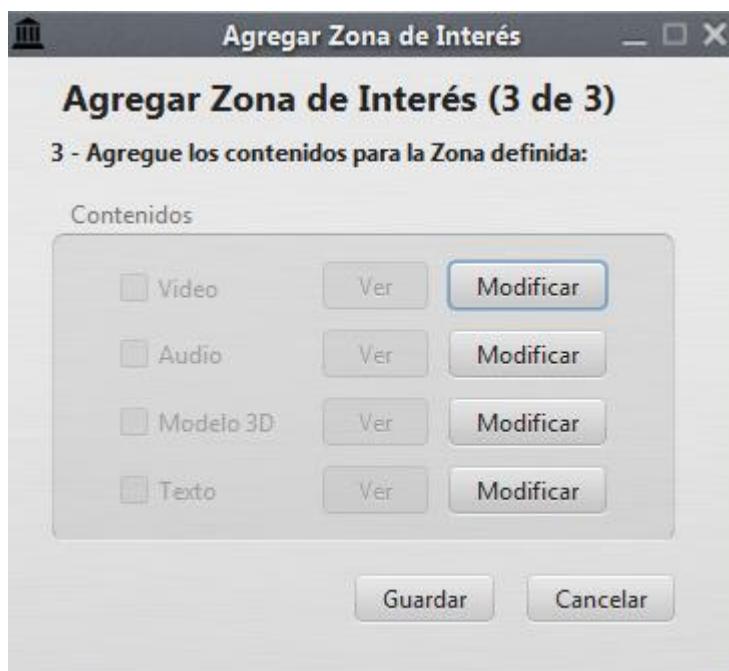


Figura 45

## Aplicación móvil.

### 2.4.17. Menú principal.



Figura 46

## Modo manual.

### 2.4.18. Lista salas.

The screenshot shows a mobile application interface titled 'Salas'. At the top left is a back arrow labeled 'enCuadro App'. The title 'Salas' is at the top right. Below the title is a list of five entries, each with a small profile picture on the left:

- Juan Manuel Blanes  
Juan Manuel Blanes (Montevideo, 8 de Junio de 1830 ? Pisa, Italia, 15 de abril de 1901) fue un pintor uruguayo, el más reconocido del Río de la Plata por sus interpreta... >
- Pedro Figari  
Pedro Figari Solari (n. Montevideo, 29 de junio de 1861 - ibidem, 24 de julio de 1938) pintor, abogado, político, escritor y periodista uruguayo. Una de las figuras más de... >
- Joaquín Torres García  
Joaquín Torres García (Montevideo, 28 de julio de 1874 - Montevideo, 8 de agosto de 1949) fue un pintor constructivista, profesor y escritor uruguayo. >
- Petrona Viera  
NACE EN MONTEVIDEO, EL 24 DE MARZO DE 1895. DE PEQUEÑA PADECIÓ UNA TERRIBLE ENFERMEDAD LLAMADA MENINGITIS QUE LA DEJÓ SORDOMUDA. E... >
- Carlos Páez Vilaró  
Nació en Montevideo, el 1 de noviembre de 1923. Es un pintor, ceramista, escultor, muralista, escritor, compositor y constructor uruguayo. >

Figura 47

### 2.4.19. Lista obras.

The screenshot shows a mobile application interface titled 'Obras'. At the top left is a back arrow labeled 'Salas'. The title 'Obras' is at the top right. Below the title is a list of six entries, each with a small thumbnail image on the left:

- Pedro Figari  
Cambacuá >
- Pedro Figari  
Candombe >
- Pedro Figari  
Grito de Asencio >
- Pedro Figari  
Pericón en el patio de la estancia >
- Pedro Figari  
Pique nique >
- Pedro Figari  
Toque de oración >

Figura 48

## Modo automático.

### 2.4.20. Identificar sala.



Figura 50

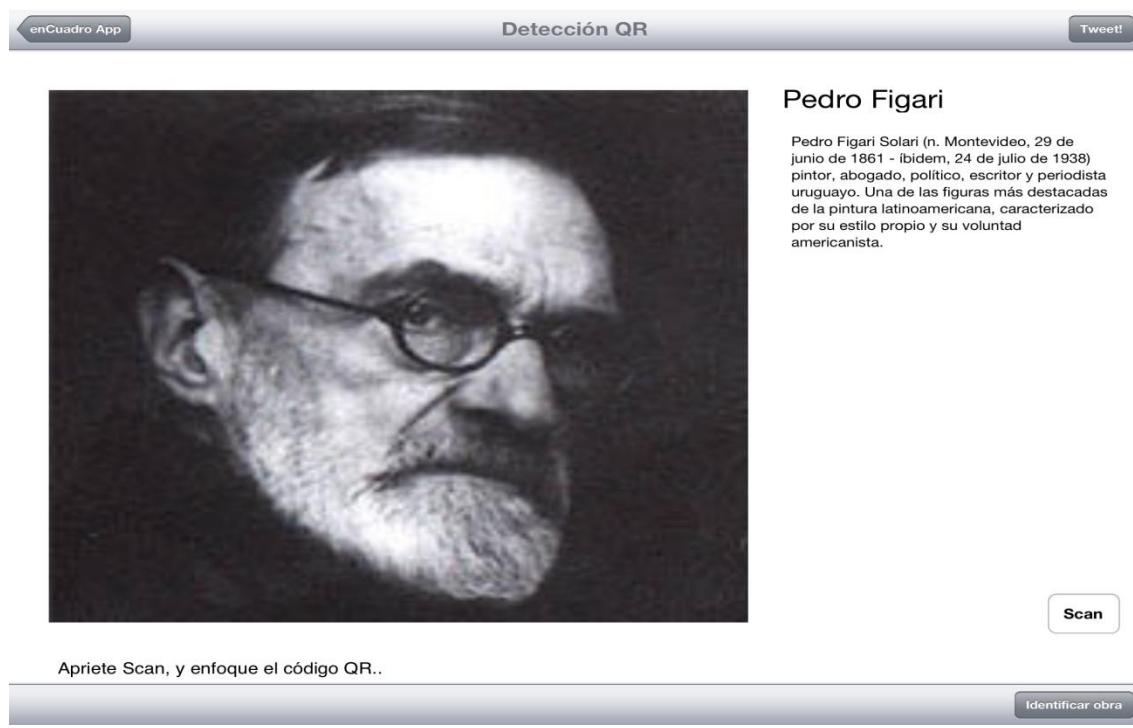


Figura 49

#### 2.4.21. Identificar obra.



Figura 51

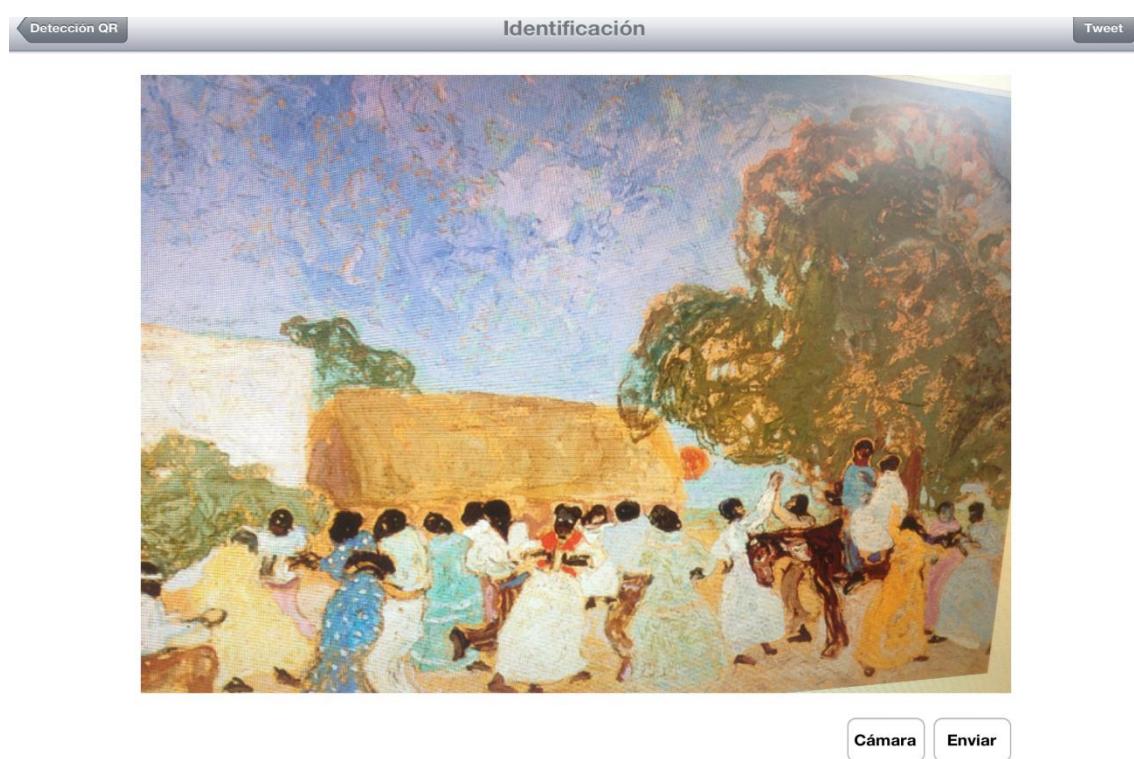


Figura 52

## 2.4.22. Obra completa.



Realizado: c.1925  
Técnica: Óleo sobre cartón  
Medidas: 62 x 82 cm



**Figura 53**

### 2.4.23. Realidad aumentada.



Figura 54

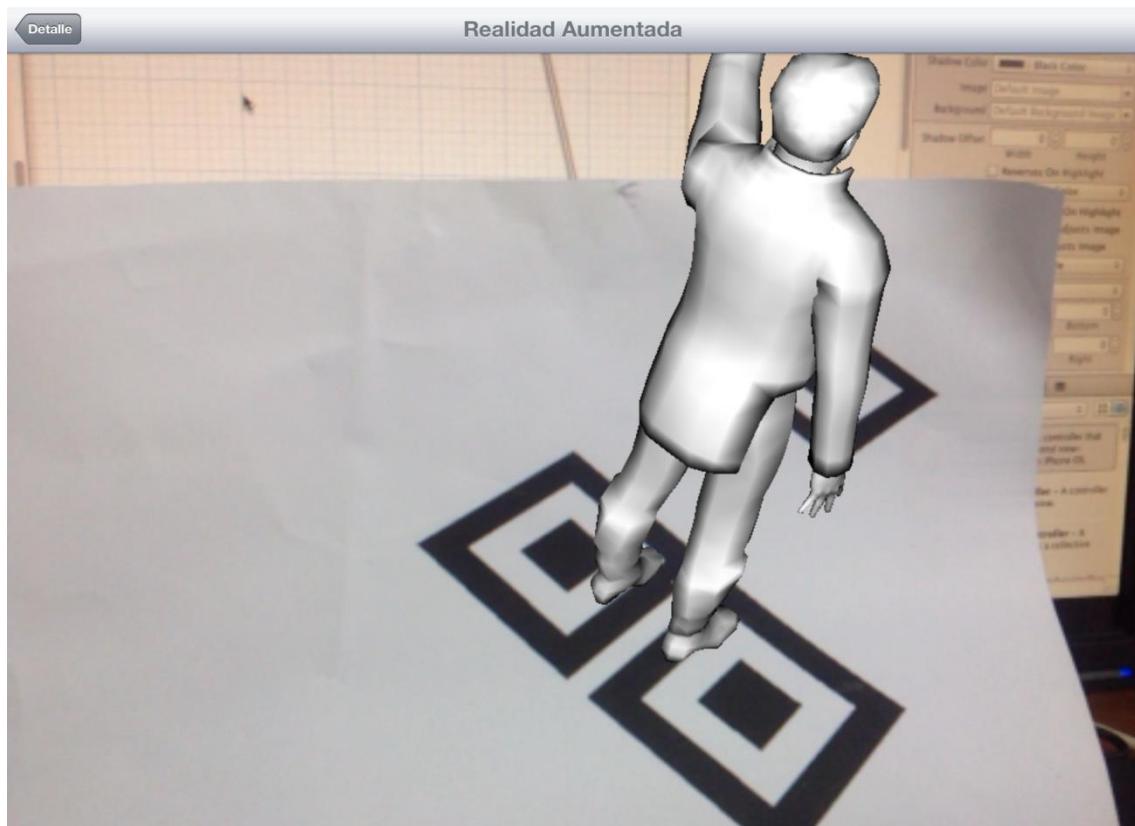
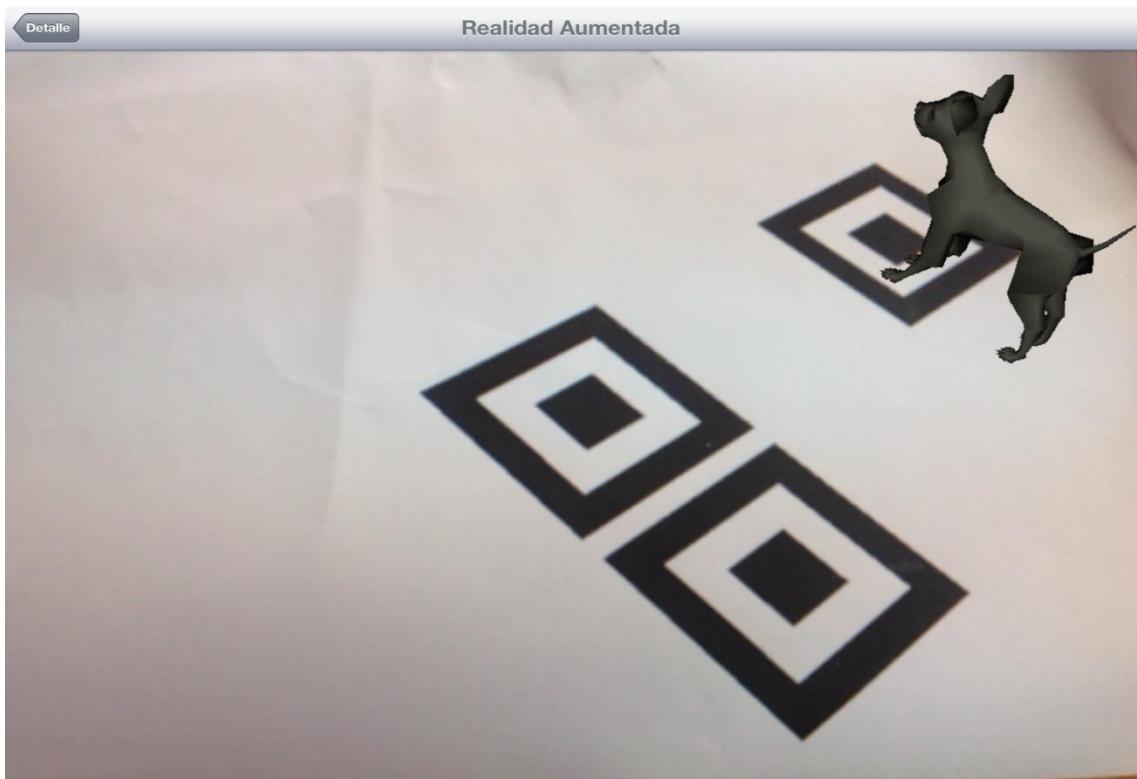


Figura 55



**Figura 56**



**Figura 57**

## 2.5. Código fuente.

### 2.5.1. Ejemplo de código para uso de SOAP.

#### Código en Java:

- Método que ejecuta el llamado a la función del servidor obteniendo una devolución inmediata.

```
private static String nombrefuncion(int param1, java.lang.String param2,  
java.lang.String param3) {  
    _109._2._0._10.server_php.Comision service = new  
_109._2._0._10.server_php.Comision();  
    _109._2._0._10.server_php.ComisionPortType port =  
service.getComisionPort();  
    return port.agregarContenidoSala(idSala, tipo, nombre);  
}
```

#### Código en Objective-C:

- Método que ejecuta el llamado a la función del servidor.

```
-(conn *)initconFunc: (NSString *)nombrefuncion{  
    NSString *soapMessage = @”<?xml version=\”1.0\” encoding=\”utf-  
8\”?>\n”  
“<soap:Envelope xmlns:xsi=\”http://www.w3.org/2001/XMLSchema-instance\”  
xmlns:xsd=\”http://www.w3.org/2001/XMLSchema\”  
xmlns:soap=\”http://schemas.xmlsoap.org/soap/envelope/\”>\n”  
“<soap:Body>\n”  
“<nombrefuncion  
xmlns=\”http://10.0.2.109/server_php/server_php.php/nombrefuncion\”>\n”  
“</nombrefuncion>\n”  
“</soap:Body>\n”  
“</soap:Envelope>\n”;  
    NSMutableString *direccion = [NSMutableString  
stringWithString:kPostURL];  
    [direccion setString:[direccion  
stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding]];  
    NSURL *url = [NSURL URLWithString:direccion];  
    NSMutableURLRequest *request = [NSMutableURLRequest  
requestWithURL:url];  
    NSString *msgLength = [NSString stringWithFormat:@”%d”, [soapMessage  
length]];  
    [request addValue: @”text/xml; charset=utf-8”  
forHTTPHeaderField:@”Content-Type”];  
    NSString *s =  
@”http://10.0.2.109/server_php/server_php.php/nombrefuncion”;  
    [request addValue: s forHTTPHeaderField:@”SOAPAction”];  
    [request addValue: msgLength forHTTPHeaderField:@”Content-Length”];  
    [request setHTTPMethod:@”POST”];  
    [request setHTTPBody: [soapMessage  
dataUsingEncoding:NSUTF8StringEncoding]];}
```

```

[request setTimeoutInterval:60];
NSURLConnection *connection = [[NSURLConnection alloc]
initWithRequest:theRequest delegate:self];
}

- Métodos que se ejecutan al recibir datos, o al fallar la conexión. Los últimos tres métodos transforman el XML que devuelve la función del servidor a algo legible.

-(void)connection: (NSURLConnection *)connection didReceiveResponse:
(NSURLResponse *)response{
    [webData setLength: 0];
}

-(void)connection: (NSURLConnection *)connection didReceiveData: (NSData *)
*)data{
    [webData appendData:data];
}

-(void)connection: (NSURLConnection *) connection didFailWithError:
(NSError *)error{
    [connection release];
    [webData release];
}

-(conn *)connectionDidFinishLoading: (NSURLConnection *)connection{
NSString *theXML = [[NSString alloc] initWithBytes: [webData mutableBytes]
length:[webData length] encoding:NSUTF8StringEncoding];
    [theXML release];
if (xmlParser){
    [xmlParser release];
}
xmlParser = [[NSXMLParser alloc] initWithData: webData];
[xmlParser setDelegate: self];
[xmlParser setShouldResolveExternalEntities:YES];
[xmlParser parse];
[connection release];
[webData release];
}

-(void) parser: (NSXMLParser *) parser didStartElement: (NSString *)
elementName namespaceURI: (NSString *) namespaceURI qualifiedName:
(NSString *) qName attributes: (NSDictionary *) attributeDict {
    if([elementName isEqualToString:@"return"]){
        if (!soapResults){
            soapResults = [[NSMutableString alloc] init];
        }
        elementFound = YES;
    }
}

```

```
- (void)parser: (NSXMLParser *) parser foundCharacters: (NSString *)string{
    if (elementFound){
        [soapResults appendString: string];
    }
}

-(void)parser: (NSXMLParser *)parser didEndElement: (NSString
*)elementName namespaceURI: (NSString *)namespaceURI qualifiedName:
(NSString *)qName{
    if ([elementName isEqualToString:@"return"]){
        //hacer algo
    }
}
```

## 2.5.2. Ejemplo de código para uso de FTP.

### Código en Java:

- Función que sube los archivos al servidor. Ruta, es la ruta en la que queda alojado el archivo en el servidor, y source es el archivo.

```
public static void upload(String ruta, File source) throws  
MalformedURLException, IOException {  
    if(ruta != null && source != null){  
        StringBuffer sb = new StringBuffer(ruta);  
        sb.append(";type=i");  
        BufferedInputStream bis = null;  
        BufferedOutputStream bos = null;  
        try{  
            URL url = new URL(sb.toString());  
            URLConnection urlc = url.openConnection();  
            bos = new BufferedOutputStream(urlc.getOutputStream());  
            bis = new BufferedInputStream(new FileInputStream(source));  
            int i;  
            while ((i = bis.read()) != -1){  
                bos.write(i);  
            }  
        }finally{  
            if(bis != null) {  
                try{  
                    bis.close();  
                }catch (IOException ioe){  
                    ioe.printStackTrace();  
                }  
            }  
            if(bos != null){  
                try{  
                    bos.close();  
                }catch (IOException ioe){  
                    ioe.printStackTrace();  
                }  
            }  
        }  
    }else{  
        System.out.println("Input not available.");  
    }  
}
```

- Función que baja archivos del servidor a la PC. RutaFTP, es la ruta en la que se encuentra alojado el archivo en el servidor, y destination, el archivo a donde se bajará.

```
public static void download(String rutaFTP, File destination) throws  
MalformedURLException, IOException {  
    if(rutaFTP != null && destination != null){  
        StringBuffer sb = new StringBuffer(rutaFTP);  
        sb.append(";type=i");  
        BufferedInputStream bis = null;  
        BufferedOutputStream bos = null;  
        try{  
            URL url = new URL(sb.toString());  
            URLConnection urlc = url.openConnection();  
            bis = new BufferedInputStream(urlc.getInputStream());  
            bos = new BufferedOutputStream(new  
FileOutputStream(destination));  
            int i;  
            while((i = bis.read()) != -1){  
                bos.write(i);  
            }  
        }finally{  
            if(bis != null) {  
                try{  
                    bis.close();  
                }catch (IOException ioe){  
                    ioe.printStackTrace();  
                }  
            }  
            if(bos != null){  
                try{  
                    bos.close();  
                }catch(IOException ioe){  
                    ioe.printStackTrace();  
                }  
            }  
        }  
    }else{  
        System.out.println("Input not available");  
    }  
}
```

### Código en Objective-C:

- Funciones encargadas de subir archivos al servidor.

```
- (FTPUupload*)initWithString:(NSString*)file{
    [self startSend:file];
    return self;
}

-(void)startSend:(NSString *)filePath{
    BOOL success;
    NSURL *url;
    url = [[NetworkManager sharedInstance]
smartURLForString:@"obras:12345678@10.0.2.109"];
    success = (url != nil);
    if (success) {
        url =
CFBridgingRelease(CFURLCreateCopyAppendingPathComponent(NULL, ( CFURLRef)
url, ( CFStringRef) [filePath lastPathComponent], false));
        success = (url != nil);
    }
    if(!success){
        NSLog(@"Invalid URL");
    }
    else{
        self.fileStream = [NSInputStream
inputStreamWithFileAtPath:filePath];
        assert(self.fileStream != nil);
        [self.fileStream open];
        self.networkStream =
CFBridgingRelease(CFWriteStreamCreateWithFTPURL(NULL, ( CFURLRef) url));
        self.networkStream.delegate = self;
        [self.networkStream scheduleInRunLoop:[NSRunLoop currentRunLoop]
forMode:NSDefaultRunLoopMode];
        [self.networkStream open];
        [self sendDidStart];
    }
}
```

```

- (void)stream:(NSStream *)aStream handleEvent:(NSStreamEvent)eventCode{

    switch (eventCode) {
        case NSStreamEventOpenCompleted: {
            NSLog(@"Opened connection");
        } break;
        case NSStreamEventHasBytesAvailable: {
            assert(NO);
        } break;
        case NSStreamEventHasSpaceAvailable: {
            if (self.bufferOffset == self.bufferLimit) {
                NSInteger bytesRead;
                bytesRead = [self.fileStream read:self.buffer
maxLength:kSendBufferSize];
                if (bytesRead == -1) {
                    [self stopSendWithStatus:@"File read error"];
                } else if (bytesRead == 0) {
                    [self stopSendWithStatus:nil];
                } else {
                    self.bufferOffset = 0;
                    self.bufferLimit = bytesRead;
                }
            }
            if (self.bufferOffset != self.bufferLimit) {
                NSInteger bytesWritten;
                bytesWritten = [self.networkStream
write:&self.buffer[self.bufferOffset] maxLength:self.bufferLimit -
self.bufferOffset];
                assert(bytesWritten != 0);
                if (bytesWritten == -1) {
                    [self stopSendWithStatus:@"Network write error"];
                } else {
                    self.bufferOffset += bytesWritten;
                }
            }
        } break;
        case NSStreamEventErrorOccurred: {
            [self stopSendWithStatus:@"Stream open error"];
        } break;
        case NSStreamEventEndEncountered: {
        } break;
        default: {
            assert(NO);
        } break;
    }
}

- (void)sendDidStart{
    [[NetworkManager sharedInstance] didStartNetworkOperation];
}

```

```
- (void)sendDidStopWithStatus:(NSString *)statusString{
    if (statusString == nil) {
        NSLog(@"Put Succedeed");
        finiteUpload = YES;
    }
    [[NetworkManager sharedInstance] didStopNetworkOperation];
}

- (uint8_t *)buffer{
    return self->_buffer;
}

- (void)stopSendWithStatus:(NSString *)statusString{
    if (self.networkStream != nil) {
        [self.networkStream removeFromRunLoop:[NSRunLoop currentRunLoop]
forMode:NSDefaultRunLoopMode];
        self.networkStream.delegate = nil;
        [self.networkStream close];
        self.networkStream = nil;
    }
    if (self.fileStream != nil) {
        [self.fileStream close];
        self.fileStream = nil;
    }
    [self sendDidStopWithStatus:statusString];
}

- (BOOL)isSending{
    return (self.networkStream != nil);
}
```

- Funciones que bajan archivos del servidor a una carpeta temporal del dispositivo móvil.

```
-(FTP*)initWithString:(NSString *)ruta yotroString:(NSString *)rutaFTP{
    finiteFTP = NO;
    anduvo = YES;
    BOOL success;
    NSURL *url;
    NSURLRequest *request;
    url = [[NetworkManager sharedInstance] smartURLForString:rutaFTP];
    success = (url != nil);
    if ( ! success) {
        finiteFTP = YES;
        anduvo = NO;
    }
    else{
        self.filePath = ruta;
        assert(self.filePath != nil);
        self.fileStream = [NSOutputStream
outputStreamToFileAtPath:self.filePath append:NO];
        assert(self.fileStream != nil);
        [self.fileStream open];
        request = [NSURLRequest requestWithURL:url];
        assert(request != nil);
        connection = [NSURLConnection connectionWithRequest:request
delegate:self];
        assert(connection != nil);
        [self receiveDidStart];
    }
    return self;
}

-(void)receiveDidStart{
    [[NetworkManager sharedInstance] didStartNetworkOperation];
}

-(void)receiveDidStopWithStatus:(NSString *)statusString{
    if (statusString == nil) {
        assert(self.filePath != nil);
        statusString = @"GET succeeded";
        finiteFTP = YES;
        anduvo = YES;
    }
    else{
        finiteFTP = YES;
        anduvo = NO;
    }
    [[NetworkManager sharedInstance] didStopNetworkOperation];
}
```

```
- (void)stopReceiveWithStatus:(NSString *)statusString{
    if (connection != nil) {
        [connection cancel];
        connection = nil;
    }
    if (self.fileStream != nil) {
        [self.fileStream close];
        self.fileStream = nil;
    }
    [self receiveDidStopWithStatus:statusString];
    self.filePath = nil;
}

-(void)connection:(NSURLConnection *)theConnection didReceiveData:(NSData *)data{
    #pragma unused(theConnection)
   NSUInteger dataLength;
    const uint8_t *dataBytes;
    NSInteger bytesWritten;
    NSUInteger bytesWrittenSoFar;
    assert(theConnection == connection);
    dataLength = [data length];
    dataBytes = [data bytes];
    bytesWrittenSoFar = 0;
    do {
        bytesWritten = [self.fileStream
write:&dataBytes[bytesWrittenSoFar] maxLength:dataLength -
bytesWrittenSoFar];
        assert(bytesWritten != 0);
        if (bytesWritten <= 0) {
            [self stopReceiveWithStatus:@"File write error"];
            break;
        } else {
            bytesWrittenSoFar += (NSUInteger) bytesWritten;
        }
    } while (bytesWrittenSoFar != dataLength);
}

-(void)connectionDidFinishLoading:(NSURLConnection *)theConnection{
    #pragma unused(theConnection)
    assert(theConnection == connection);
    [self stopReceiveWithStatus:nil];
}
```

### 3. Glosario.

**Android:** Es un sistema operativo inicialmente pensado para dispositivos móviles con pantalla táctil como teléfonos inteligentes o tabletas.

**Apache Tomcat:** Es un servidor web con soporte de servlets y JSPs.

**C++:** Es un lenguaje de programación, como extensión del lenguaje de programación C.

**Caso de uso:** Es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia el actor sobre el propio sistema. Se representan en diagramas de casos de uso, que sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas.

**Casos de usos críticos:** Son los casos de uso elementales para el sistema, es decir, que sin ellos, no tiene sentido que los demás casos de uso existan.

**Clase:** Es una construcción que se utiliza como un modelo para crear objetos de ese tipo.

**Cliente FTP:** Es quien emplea el protocolo FTP para conectarse a un servidor FTP para transferir archivos.

**C-Make:** Es una herramienta multiplataforma de generación o automatización de código.

**Codecs:** Es la abreviatura de codificador-decodificador. Describe una especificación desarrollada en software, hardware o una combinación de ambos, capaz de transformar un archivo con un flujo de datos o una señal.

**Detección QR:** Se realiza mediante un dispositivo que contenga una aplicación de lector de QR, mediante este programa se detecta el código que brinda la información que contiene.

**Detección Sift:** Método para extraer características distintivas de las imágenes en escala de grises, es decir que puede ser utilizado para reconocer la misma característica entre diferentes vistas de un mismo objeto o escenas.

**Entorno de desarrollo:** Programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios.

Entorno distribuido: Es un modelo para resolver problemas de computación masiva utilizando un gran número de ordenadores organizados en racimos incrustados en una infraestructura de telecomunicaciones distribuida.

Entorno multisensorial: Entornos con materiales determinados diseñados para que los usuarios estén expuestos a estímulos controlados para conseguir objetivos, es decir si se quiere estimular el oído, se utiliza material sonoro.

Feedback: En español, retroalimentación. Es un proceso circular en el cual parte de la salida es remitida de nuevo a la entrada como información sobre la primera respuesta, haciendo así que el sistema se autorregule para mantener un equilibrio u orientarse a una meta.

FileChooser: Es una clase java que nos permite mostrar fácilmente una ventana para la selección de un fichero.

Flash: Aplicación de creación y manipulación de gráficos vectoriales con posibilidades de manejo de código mediante un lenguaje de scripting.

Gmail: Servicio de correo electrónico con posibilidades POP3 e IMAP gratuito proporcionado por la empresa estadounidense Google.

Google Docs: Programa gratuito basado en web para crear documentos en línea con la posibilidad de colaborar en grupo. Incluye procesador de textos, hoja de cálculo, programa de presentación básico, creador de dibujos y un editor de formularios destinados a encuestas.

GPS: En español, Sistema de Posicionamiento Global. Es un sistema global de navegación por satélite que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo.

Irix: Sistema operativo compatible con Unix, con soporte de gráficos 3D, video y transferencia de datos de gran ancho de banda.

Json: es un formato ligero de intercambio de datos.

LAN: Interconexión de computadoras y periféricos para formar una red dentro de una empresa u hogar, limitada generalmente a un edificio.

Lenguaje de programación: Lenguaje artificial que puede ser usado para controlar el comportamiento de una máquina, especialmente una computadora.

Librerías: Recopilación de ficheros cabecera y bibliotecas con rutinas.

**Linux:** Es un núcleo libre de sistema operativo basado en Unix.

**Mac OS:** Sistema operativo creado por Apple para su línea de computadoras Macintosh.

**Mac OSX:** Serie de sistemas operativos basados en Unix desarrollado, comercializado y vendido por Apple.

**Multiplataforma:** Que tiene la capacidad de soportar múltiples plataformas.

**Museo interactivo (in situ):** Es el uso de las herramientas tecnológicas para crear un diálogo con el visitante a través de pantallas interactivas, juegos de video, audiovisuales, herramientas táctiles y/o sonoras.

**Museo virtual:** Conocido también como museo digital o museo on-line, es donde la muestra física ahora es transportable, itinerante e imaginaria, es un entorno digital de acceso a través de internet que tiene la principal característica de la interactividad.

**Open Office:** Es una suite de código abierto que cuenta con aplicaciones de procesador de textos, hoja de cálculo, gráficos y bases de datos.

**Open source:** Denominación para aquellas aplicaciones que tienen su código fuente liberado.

**Plugins:** Programa que puede anexarse a otro para aumentar sus funcionalidades (generalmente sin afectar otras funciones ni afectar la aplicación principal).

**Project glass:** Programa de investigación y desarrollo de Google para desarrollar lentes de realidad aumentada.

**Protocolo:** Es el lenguaje (conjunto de reglas formales) que permite comunicar nodos, computadoras entre sí. Al encontrar un lenguaje común no existen problemas de compatibilidad entre ellas.

**Quicktime:** Conjunto de bibliotecas y un reproductor multimedia (Quicktime player) desarrollados por Apple.

**Realidad virtual:** Ciencia basada en el empleo de ordenadores y otros dispositivos, cuyo fin es producir una apariencia de realidad que permita al usuario tener la sensación de estar presente en ella.

**Servidor:** En internet, los servidores son los proveedores de todos sus servicios.

Servidor FTP: Computadora que funciona como servidor para ofrecer ficheros a través del protocolo de FTP a clientes FTP o a un navegador que lo soporte.

SMID: Sistema Móvil de Información Digital basado en tecnologías de RA.

Soap: Es un protocolo para el intercambio de mensajes sobre redes de computadoras.

Solaris: Sistema operativo de tipo Unix certificado oficialmente como versión de Unix.

Subversion: Es un sistema de control de versiones usado para que varios desarrolladores puedan trabajar en un mismo proyecto en forma más o menos ordenada.

Swing: Es una biblioteca gráfica para Java.

WAN: Es una red de nodos que se extiende en una gran franja de territorio, ya sea a través de una ciudad, un país o incluso a nivel mundial.

Waypoints: Es un conjunto de coordenadas que identifican un punto concreto en el espacio.

Web service: Es un sistema de software diseñado para permitir interoperabilidad máquina a máquina en una red.

Wikispaces: Es un sitio web cuyas páginas pueden ser editadas por múltiples voluntarios, es decir permite desarrollar aprendizaje colaborativo través del navegador web, los usuarios pueden crear, modificar o borrar un mismo texto que comparten.

Windows: Es un sistema operativo con interfaz gráfica para computadoras personales propiedad de la empresa Microsoft.

Xcode: Es una serie de herramientas de desarrollo en Mac OS X.

XML: Acrónimo del inglés extensible markup language (lenguaje de marcado extensible). Su objetivo es conseguir páginas web más semánticas separando la estructura del contenido y permitiendo el desarrollo de vocabularios modulares.