**EnCuadro 3.0**

**Glosario,**

**Apéndice y Bibliografía**

Universidad de la República

UTU – CETP

Paysandú, 16 de Mayo de 2014

**Tutor:**

Marcelo Scotto

**Cliente:**

Juan Cardelino

**Integrantes**:

Mauricio Carrey

Giancarlo Echeveste

Martín Nan

Índice

[1. Referencias. 5](#_Toc388017030)

[2. Apéndice. 9](#_Toc388017031)

[2.1. Casos de uso. 9](#_Toc388017032)

[2.1.1. Listar salas. 9](#_Toc388017033)

[2.1.2. Listar Obras Pertenecientes a Sala. 9](#_Toc388017034)

[2.1.3. Seleccionar Sala. 10](#_Toc388017035)

[2.1.4. Listar obras. 10](#_Toc388017036)

[2.1.5. Buscar Obra. 10](#_Toc388017037)

[2.1.6. Buscar Sala. 11](#_Toc388017038)

[2.1.7. Mostrar Contenido de Sala 11](#_Toc388017039)

[2.1.8. Mostrar Contenido de Obra 11](#_Toc388017040)

[2.1.9. Reproducir Audio 12](#_Toc388017041)

[2.1.10. Reproducir Video 12](#_Toc388017042)

[2.1.11. Identificar Sala QR. 13](#_Toc388017043)

[2.1.12. Identificar Obra QR. 13](#_Toc388017044)

[2.1.13. Reconocimiento Imagen. 14](#_Toc388017045)

[2.1.14. Recorrido Juego 15](#_Toc388017046)

[2.1.15. Listar Puntajes 15](#_Toc388017047)

[2.1.16. Estado del Juego 16](#_Toc388017048)

[2.1.17. Finalizar Juego 16](#_Toc388017049)

[2.1.18. Rellenar Cuestionario 17](#_Toc388017050)

[2.1.19. Alta visita 17](#_Toc388017051)

[2.2. Diagramas de secuencia del sistema. 18](#_Toc388017052)

[2.2.1. Listar Salas 18](#_Toc388017053)

[2.2.2. Lista Obra Perteneciente a Sala 19](#_Toc388017054)

[2.2.3. Seleccionar Sala 19](#_Toc388017055)

[2.2.4. Listar Obras 20](#_Toc388017056)

[2.2.5. Buscar Obra 20](#_Toc388017057)

[2.2.6. Buscar Sala 21](#_Toc388017058)

[2.2.7. Mostrar Contenido de Sala 22](#_Toc388017059)

[2.2.8. Mostrar Contenido de Obra 22](#_Toc388017060)

[2.2.9. Reproducir Audio 23](#_Toc388017061)

[2.2.10. Reproducir Video 24](#_Toc388017062)

[2.2.11. Identificar Sala QR 24](#_Toc388017063)

[2.2.12. Identificar Obra QR 25](#_Toc388017064)

[2.2.13. Reconocimiento de Imágenes 25](#_Toc388017065)

[2.2.14. Recorrido Juego 26](#_Toc388017066)

[2.2.15. Estado del Juego 27](#_Toc388017067)

[2.2.16. Finalizar Juego 28](#_Toc388017068)

[2.2.17. Lista Puntajes 28](#_Toc388017069)

[2.2.18. Rellenar Cuestionario 29](#_Toc388017070)

[2.2.19. Alta Visita 29](#_Toc388017071)

[2.3. Contratos. 30](#_Toc388017072)

[2.3.1. Listar salas. 30](#_Toc388017073)

[2.3.2. Listar obras pertenecientes a salas. 31](#_Toc388017074)

[2.3.3. Buscar Salas. 32](#_Toc388017075)

[2.3.4. Mostrar contenido de sala. 32](#_Toc388017076)

[2.3.5. Seleccionar Sala 33](#_Toc388017077)

[2.3.6. Listar Obras. 34](#_Toc388017078)

[2.3.7. Buscar Obras. 35](#_Toc388017079)

[2.3.8. Mostrar contenido de Obra. 36](#_Toc388017080)

[2.3.9. Reproducir Audio 38](#_Toc388017081)

[2.3.10. Reproducir video 39](#_Toc388017082)

[2.3.11. Identificar sala QR. 40](#_Toc388017083)

[2.3.12. Listado de Puntajes. 41](#_Toc388017084)

[2.3.13. Identificar Obra QR. 42](#_Toc388017085)

[2.3.14. Reconocimiento de imagen. 43](#_Toc388017086)

[2.3.15. Estado del juego. 44](#_Toc388017087)

[2.3.16. Finalizar Juego. 44](#_Toc388017088)

[2.3.17. Recorrido del Juego 45](#_Toc388017089)

[2.3.18. Rellenar Cuestionario 46](#_Toc388017090)

[2.3.19. Alta Visitante. 47](#_Toc388017091)

[2.4. Código fuente. 48](#_Toc388017092)

[2.4.1. Ejemplo de código para uso de SOAP. 48](#_Toc388017093)

[2.4.2. Ejemplo de código para uso de FTP. 49](#_Toc388017094)

[3. Glosario. 50](#_Toc388017095)

# 1. Referencias.

[0] Documentación del proyecto EnCuadro, Facultad de Ingeniería, Universidad de la República – Juan Braun, Martín Etchart, Pablo Flores y Mauricio González (Documento físico)

[1] Documentación del proyecto EnCuadro2.0, Tecnologo en Informatica, Universidad de la República – Mauricio Martinez, Maria Clara Fleitas, Martin Arevalo y Virginia Saldaña (Documento físico)

[2] Diccionario de informática, telecomunicaciones y ciencias afines – Mario León

[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. (Documento físico)

[7] Plan Ceibal:

http://www.ceibal.edu.uy/

[8] Página oficial de Layar:

http://www.layar.com/what-is-layar/

[9] Interactividad y museos - Claudia Esther Calva Montiel:

http://www.gabinetecomunicacionyeducacion.com/

[10] Página oficial de Java:

http://www.java.com/es/about

[11] SOAP - Ing. Sandro Moscatelli:

www.fing.edu.uy/inco/grupos/lins/docsgen/soap/soap.doc

[12] FTP:

http://es.wikipedia.org/wiki/File\_Transfer\_Protocol

[13] Java Barcode:

http://www.barcodelib.com.

[14] Zebra-Crossing:

https://code.google.com/p/zxing/.

[15] Plan Ceibal:

http://www.ceibal.edu.uy/Paginas/Inicio.aspx

[16] Definición de comunicación:

http://es.wikipedia.org/wiki/Comunicaci%C3%B3n

[17] New Scientist video – Transparent wall:

http://www.youtube.com/watch?feature=player\_embedded&v=Q5O13bk7z2s

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

[19] Qué es la realidad aumentada y sus aplicaciones – Angel Vilchez:

http://www.configurarequipos.com/doc1214.html

[20] Creación de marcadores para RA – Dirección operativa de incorporación de tecnologías (Plan integral de educación digital), Ministerio de Educación

[21] Definición de código QR:

http://www.codigos-qr.com//

[22] Información en el móvil – Natalia Arroyo Vázquez, Editorial UOC, 2011: Usos de código QR, pág. 80.

[23] Documentación oficial de Barcode:

http://www.barcodelib.com/java\_barcode/main.html

[24] Android

http://developer.android.com/index.html

[25] Definicion de RA

http://es.wikipedia.org/wiki/Realidad\_aumentada

[26] Streaming

http://es.wikipedia.org/wiki/Streaming

[27] Eclipse

https://www.eclipse.org/

[28] Paganza

http://emprendedoressub30.espectador.com/historia/89239/paganza

[29] Android Camara API

http://developer.android.com/guide/topics/media/camera.html#considerations

[29] Android Media Player API

http://developer.android.com/reference/android/media/MediaPlayer.html

[31] Universidad del Trabajo del Uruguay

<http://www.altillo.com/universidades/uruguay/Universidad_del_Trabajo_del_Uruguay.asp>

[32] Universidad de la República

<http://www.universidad.edu.uy/>

# 2. Apéndice.

En este documento se describen todos los casos de uso, requerimientos funcionales, diagramas de secuencia del sistema, contratos, y requerimientos que fueron modificados en nuestro proyecto.

## **2.1. Casos de uso.**

**Requerimientos funcionales.**

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, acceder a ciertas realidades aumentadas (Video, Audio, Reconocimiento de Img, etc), que se encuentran administrados en el servidor poder interactuar con un pequeño juego interactivo y contiene la capacidad para que el visitante pueda responder cuestionarios hechos por el museo.

**Modo manual.**

## 2.1.1. Listar salas.

|  |  |
| --- | --- |
| Caso de uso | **Listar** **salas** |
|
| Actores | Visitante del museo |
|
| Descripción | Este caso de uso lista todas las salas existentes en el sistema con su nombre para que el visitante seleccione la que desee refiriéndonos al caso de uso **‘Seleccionar sala’** |
|

## 2.1.2. Listar Obras Pertenecientes a Sala.

|  |  |
| --- | --- |
| Caso de uso | **Listar** **Obras Perteneciente a Sala** |
|
| Actores | Visitante del museo |
|
| Descripción | Se listan todas las obras pertenecientes a la sala seleccionada en el caso de uso ‘Seleccionar Sala’, para que el visitante seleccione y pase al caso de uso **‘Mostrar contenido de Obra’** |
|

## 2.1.3. Seleccionar Sala.

|  |  |
| --- | --- |
| Caso de uso | **Seleccionar Sala** |
|
| Actores | Visitante del museo |
|
| Descripción | El visitante selecciona la sala que desea y se mostraran los datos de esta en pantalla a menos que previamente allá seleccionado “Detalles” que obliga al sistema saltearse los datos de sala he ir directamente a la **‘Listar Obras Perteneciente a Sala’** |
|

## 2.1.4. Listar obras.

|  |  |
| --- | --- |
| Caso de uso | **Listar** **obras** |
|
| Actores | Visitante del museo |
|
| Descripción | Se listan todas las obras pertenecientes a la museo, para que el visitante seleccione y pase al caso de uso **‘Mostrar contenido de Obra’** |
|

## 2.1.5. Buscar Obra.

|  |  |
| --- | --- |
| Caso de uso | **Buscar Obra** |
|
| Actores | Visitante del museo |
|
| Descripción | El visitante ingresa el nombre de una obra o parte de este y el sistema lista las obras coincidentes, para que el visitante seleccione y pase al caso de uso **‘Mostrar contenido de Obra’** |
|

## 2.1.6. Buscar Sala.

|  |  |
| --- | --- |
| Caso de uso | **Buscar Sala** |
|
| Actores | Visitante del museo |
|
| Descripción | El visitante ingresa el nombre de una sala o parte de este y el sistema lista las obras coincidentes, para que el visitante seleccione y pase al caso de uso **‘Mostrar contenido de Sala’** |
|

## 2.1.7. Mostrar Contenido de Sala

|  |  |
| --- | --- |
| Caso de uso | **Mostrar contenido de Sala** |
|
| Actores | Visitante del museo |
|
| Descripción | Visualiza en pantalla los detalles de la sala junto a su imagen (descargada por ftp) que fue seleccionada anteriormente en el caso de uso **‘Seleccionar sala’.** Permitiéndole al visitante acceso a **‘Reproducir audio’**, **‘Reproducir Video’**, **‘Listar Obras Perteneciente a Sala’** y **‘Reconocimiento Imagen’**. |
|

## 2.1.8. Mostrar Contenido de Obra

|  |  |
| --- | --- |
| Caso de uso | **Mostrar contenido de Obra** |
|
| Actores | Visitante del museo |
|
| Descripción | Visualiza en pantalla los detalles e imagen (descargada por ftp) de la obra seleccionada en cualquiera de los casos de usos que listan las obras. Permitiéndole al visitante acceso a **‘Reproducir audio’** y **‘Reproducir Video’**. Luego se verificara la existencia de un juego que se encuentre activo, en caso de no haber ninguno no existirá suceso alguno. En cambio si existe algún juego se buscara si esta obra pertenece al juego y si es indicado consultar al visitante si este quiere comenzar a jugar, derivándose al caso de uso **‘Comienzo de juego’**. Si previamente ya se ha comenzado un juego entonces comienza el caso de uso **‘Buscando Pista’** |
|

## 2.1.9. Reproducir Audio

|  |  |
| --- | --- |
| Caso de uso | **Reproducir audio** |
|
| Actores | Visitante del museo |
|
| Descripción | Caso de uso derivado, tanto del contenido de sala como del contenido de obra, reproduciendo el archivo de audio que es descargado en el instante. Puede ser detenido y devuelto a reproducir. |
|

## 2.1.10. Reproducir Video

|  |  |
| --- | --- |
| Caso de uso | **Reproducir Video** |
|
| Actores | Visitante del museo |
|
| Descripción | Caso de uso derivado, tanto del contenido de sala como del contenido de obra, cuando el visitante indica que quiere reproduje el video relacionado, generando una conexión por streaming con el servidor y la dirección del video. Su reproducción comienza en cuanto el primer buff está completo. Tiene la opción de: Detener, Continuar, adelantar y atrasar. |
|

**Modo Automático.**

## 2.1.11. Identificar Sala QR.

|  |  |
| --- | --- |
| Caso de uso | **Identificar** **sala (QR)** |
|
| Actores | Visitantedel 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 correspondiendo al caso de uso ‘Mostrar contenido de Sala’, en caso de que el ID transmitido por el QR no sea una sala válida, se le notificará al usuario. |
|

## 2.1.12. Identificar Obra QR.

|  |  |
| --- | --- |
| Caso de uso | **Identificar** **Obra (QR)** |
|
| Actores | Visitantedel museo |
|
| Descripción | Este caso de uso permite identificar la Obra del museo que se encuentra al frente del 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 obra correspondiendo al caso de uso ‘Mostrar contenido de Obra’, en caso de que el ID transmitido por el QR no sea una sala válida, se le notificará al usuario. |
|

## 2.1.13. Reconocimiento Imagen.

|  |  |
| --- | --- |
| Caso de uso | **Reconocimiento Imagen** |
|
| Actores | Visitante del museo |
|
| Descripción | 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’**), se reducirá el tamaño de la imagen y la envía al servidor, junto con el ID de la sala que el sistema recuerda. 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 un mensaje de que no ha sido posible reconocerla. |
|

**Juego Interactivo**

## 2.1.14. Recorrido Juego

|  |  |
| --- | --- |
| Caso de uso | **Recorrido Juego** |
|
| Actores | Visitante del museo |
|
| Descripción | Este caso de uso viene de la mano luego de que **‘Mostrar contenido de Obra’** lo derivara.  Con la verificación de que existe un juego este a su vez verifica que la pista predeterminada como inicial del juego activo pertenezca a la obra actual en la que se encuentra el visitante. La pista inicial varia dependiendo de la clase del juego si es circular o lineal ( con principio especifico o comienza en cualquier obra).  Con el juego previamente aceptado buscara la pista que se encuentra en la obra mostrándola en pantalla y esperando la confirmación del usuario de que ha terminado de leerla. Registrará en la sesión del visitante el progreso (La pista encontrada) y continuara con el recorrido.  Si esta pista encontrada es la última pista del juego comienza el caso de uso **‘Finalizar Juego’** |
|

## 2.1.15. Listar Puntajes

|  |  |
| --- | --- |
| Caso de uso | **Listar Puntajes** |
|
| Actores | Visitante del museo |
|
| Descripción | Mostrará todos los puntajes y su respectivo Nick Name ordenados de forma descendente. En el caso de que seleccione algún registro se le mostrara la posición en la que se encuentra. También puede ver su posición actual si el visitante selecciona “Mi posición” , en caso de que no haya finalizado el juego la posición indicada será 0 |
|

## 2.1.16. Estado del Juego

|  |  |
| --- | --- |
| Caso de uso | **Estado del juego** |
|
| Actores | Visitante del museo |
|
| Descripción | Muestra en pantalla la ultima pista que visualizo, el tiempo que este lleva en el juego, las obras encontradas y las faltantes. También la opción de que pueda finalizar el juego llevando al caso de uso **‘Finalizar Juego’** y un vistaso a la lista de puntaje o sea **‘Listar de puntajes’**  Se habilita el acceso al caso de uso únicamente si este acepto participar y actualmente esta jugando. |
|

## 2.1.17. Finalizar Juego

|  |  |
| --- | --- |
| Caso de uso | **Finalizar Juego** |
|
| Actores | Visitante del museo |
|
| Descripción | Muestra en pantalla un cartel de que el juego se finalizara y si desea ingrasar su puntaje debe de ingresar un Nick Name que quedara en el registro del museo, se modificara la sesión conforme a la finalización del juego. Una vez registrado este se dirige al caso de uso **‘Lista de puntajes’** |
|

**Cuestionario**

## 2.1.18. Rellenar Cuestionario

|  |  |
| --- | --- |
| Caso de uso | **Rellenar cuestionario** |
|
| Actores | Visitante del museo |
|
| Descripción | En el momento exacto que el usuario presiona “Back” en el menú principal se considera que este término el recorrido del museo y quiere salir de la aplicación, allí se le consulta si este quiere rellenar un cuestionario antes de irse de manera opcional.  Previamente el visitante acepto rellenar el cuestionario pero aun puede retirarse presionando “Back”, si este respondió algunas preguntas se le solicitara terminar el cuestionario antes de salir.  Las preguntas y respuestas son dinámicas tanto unas como otras varían en su contenido como en cantidad. Enviando los datos al servidor y luego recibiendo un ok, este finaliza y se le agradece al usuario por su ayuda al museo |
|

**Inicio de la Aplicación**

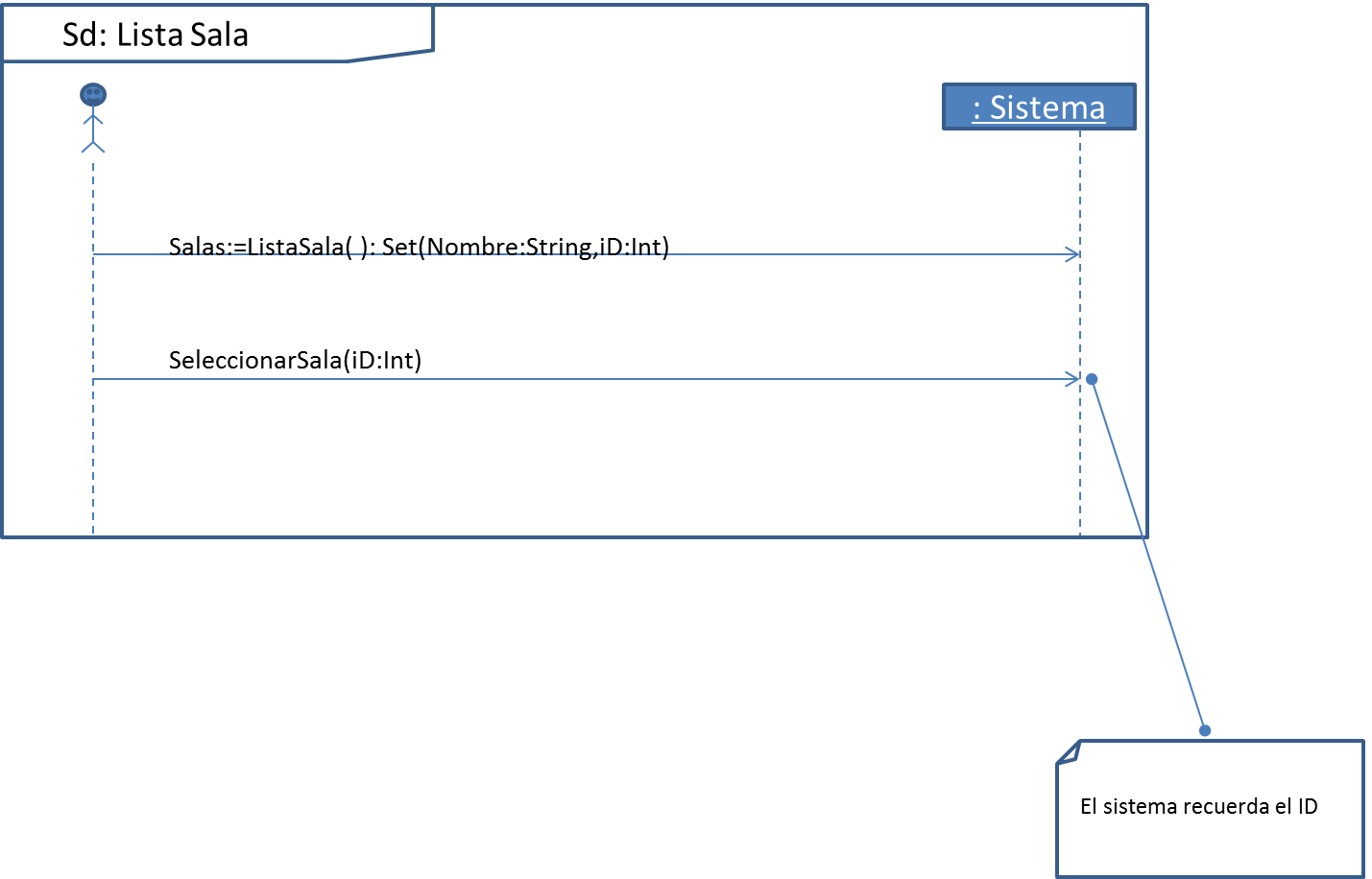
## 2.1.19. Alta visita

|  |  |
| --- | --- |
| Caso de uso | **Alta Visita** |
|
| Actores | Visitante del museo |
|
| Descripción | Se listan las preguntas y respuestas que el visitante rellena a elección, para continuar puede omitir o enviar los datos al servidor. Las respuestas seleccionadas serán parte de los atributos del visitante si es que estas existen sino dichos atributos quedaran vacíos. |
|

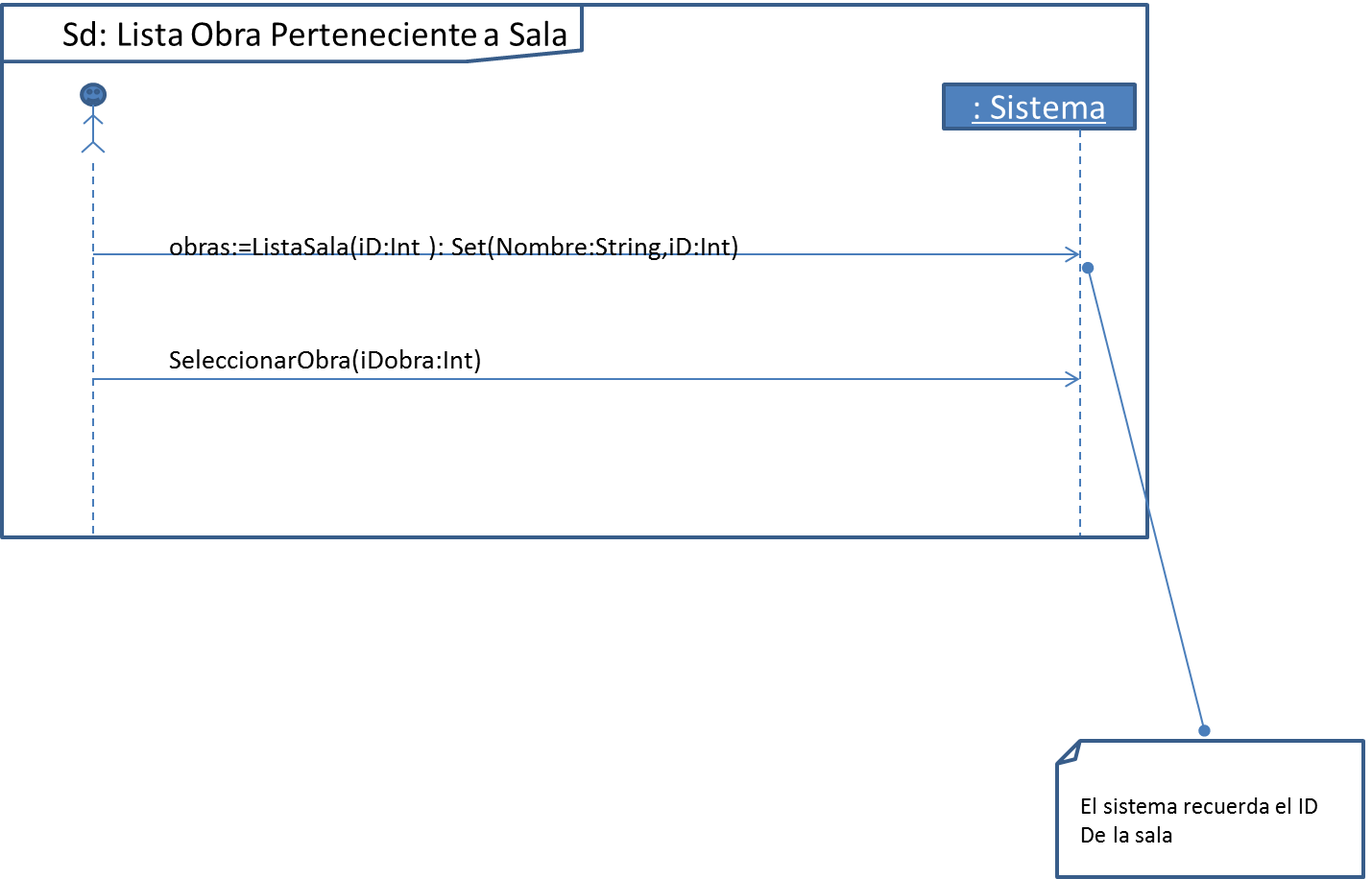
# 2.2. Diagramas de secuencia del sistema.

**Modo manual.**

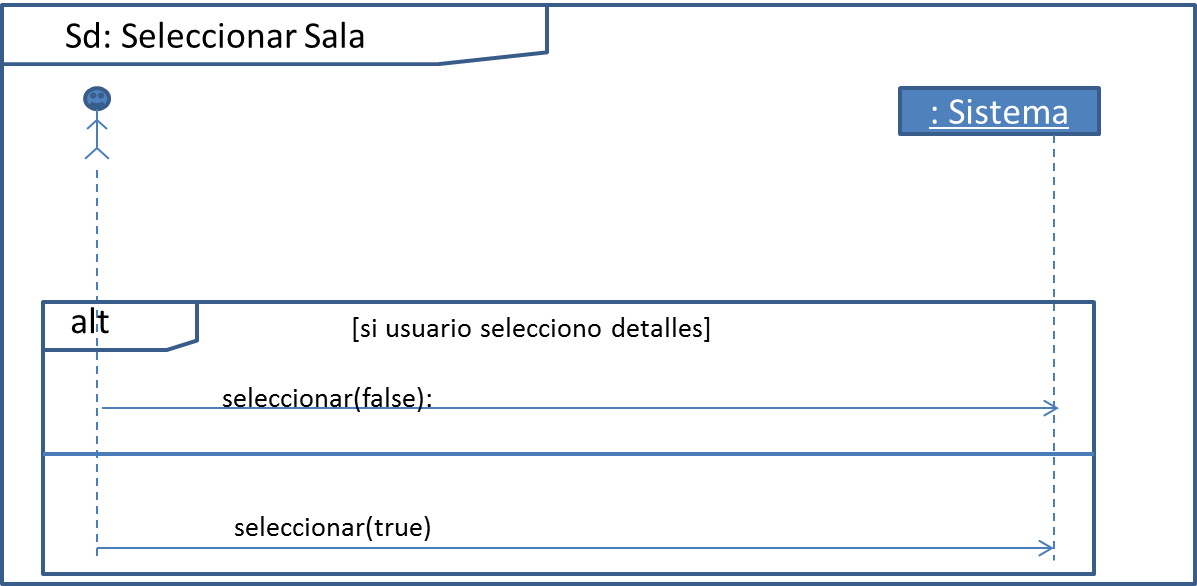
## 2.2.1. Listar Salas



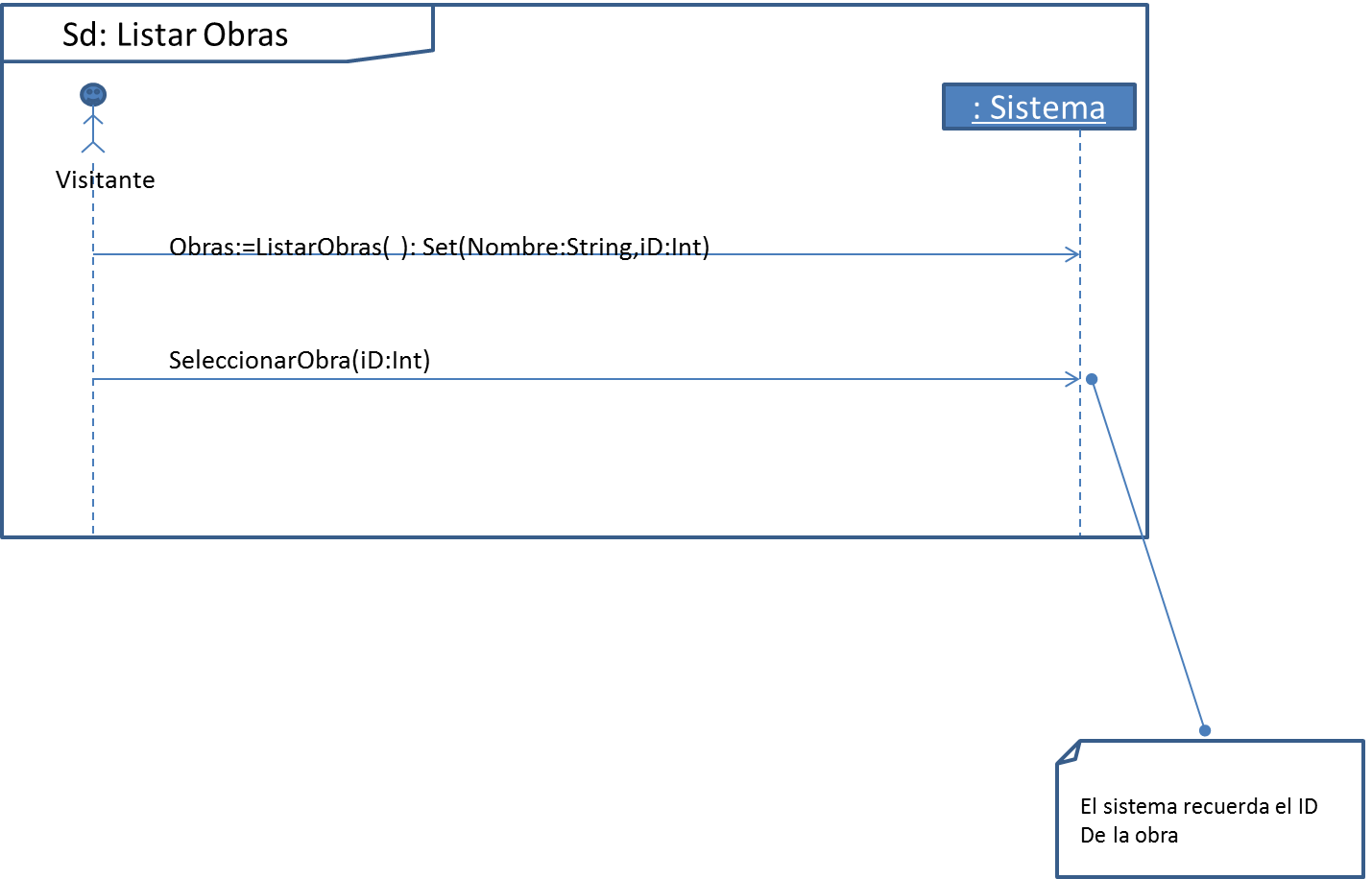
## 2.2.2. Lista Obra Perteneciente a Sala

****

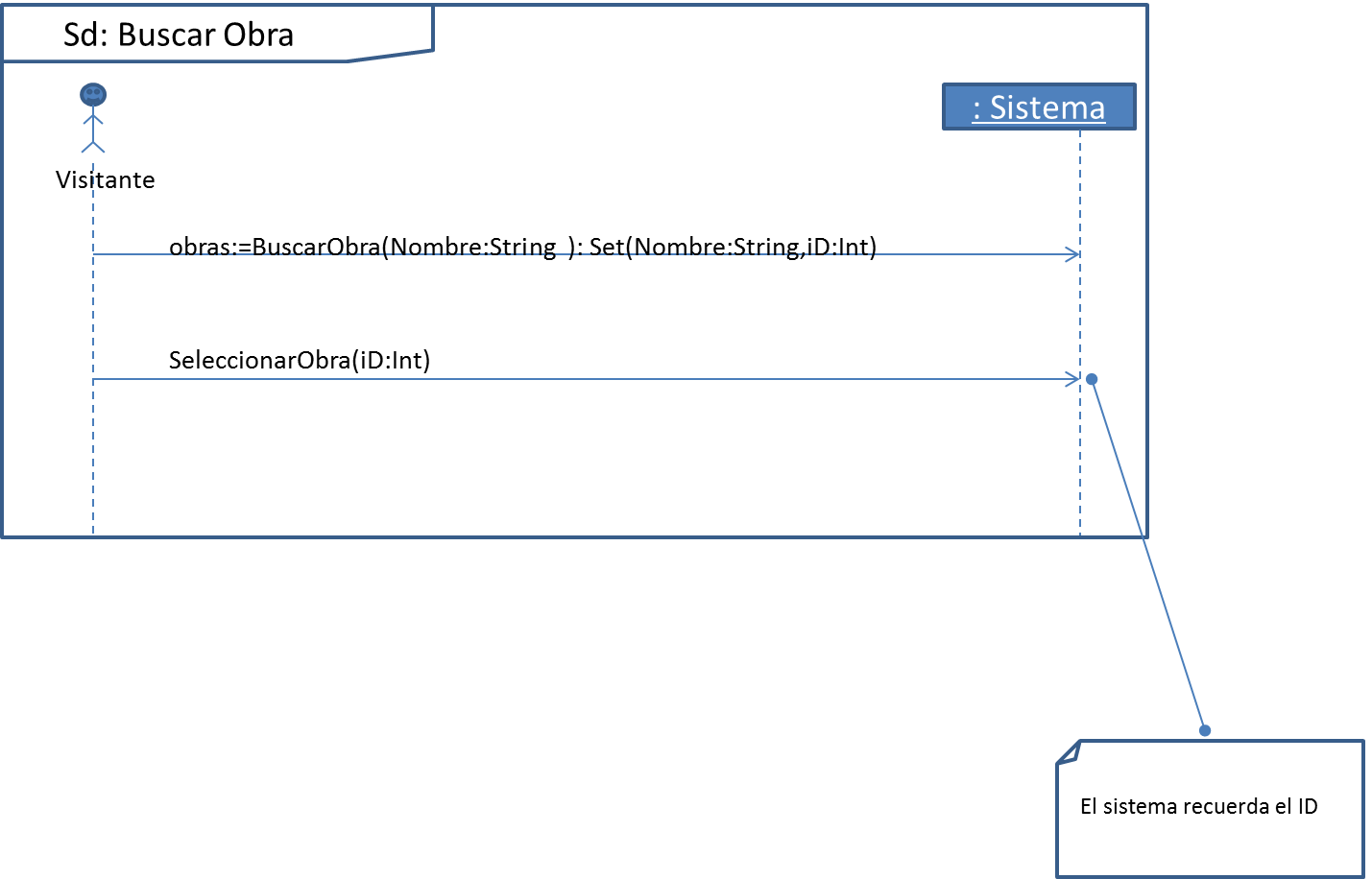
## 2.2.3. Seleccionar Sala

****

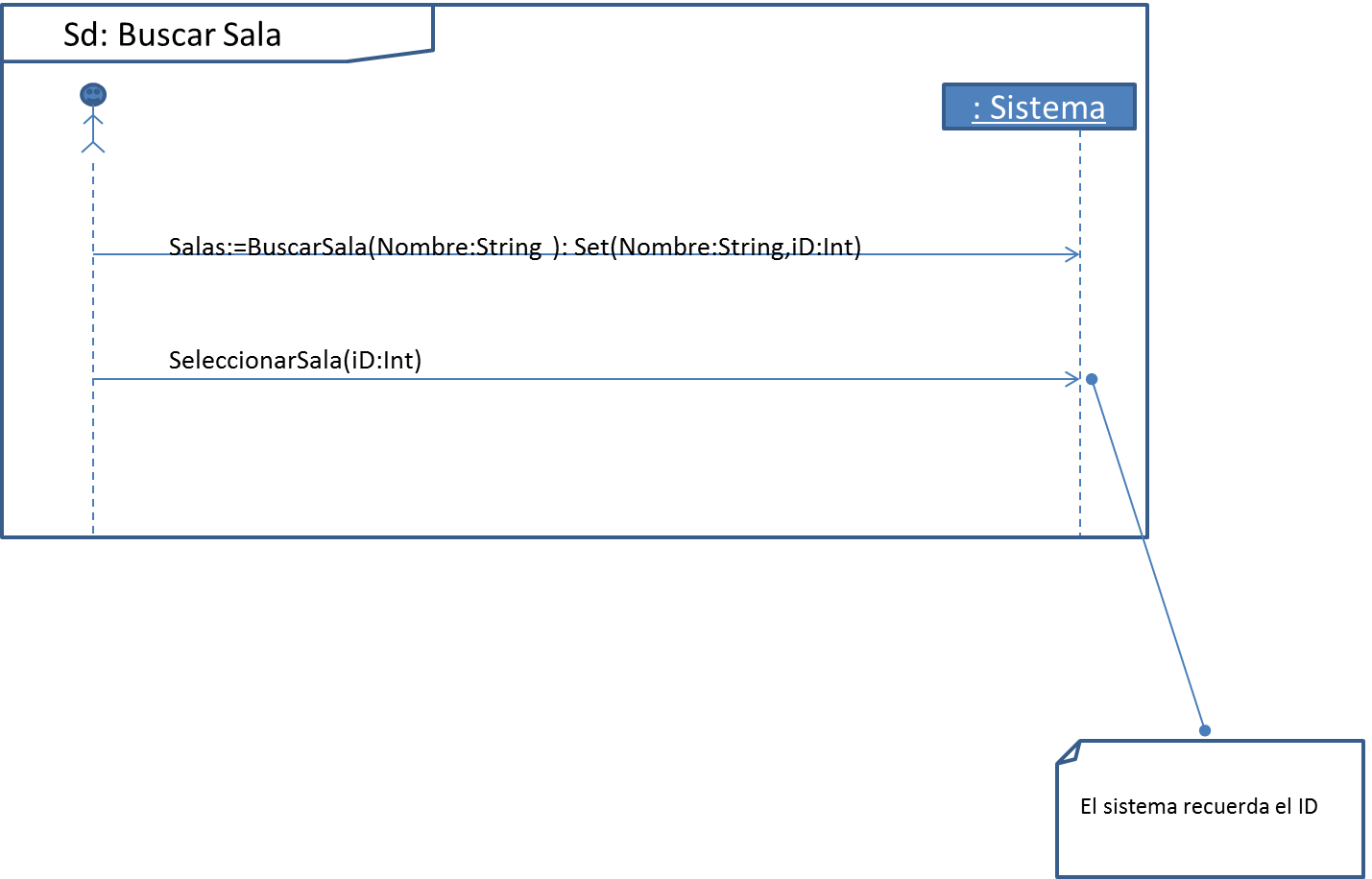
## 2.2.4. Listar Obras



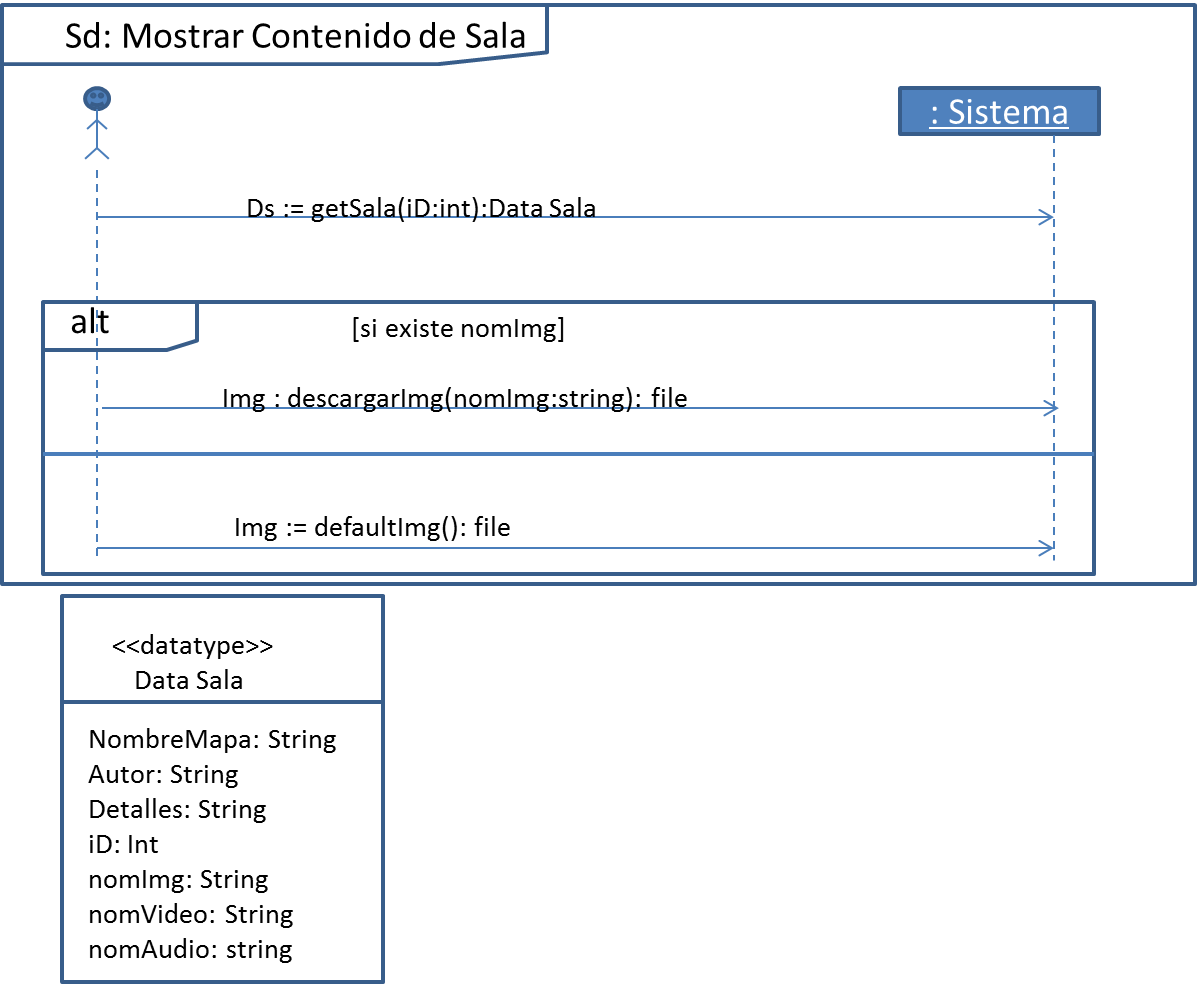
## 2.2.5. Buscar Obra

****

## 2.2.6. Buscar Sala

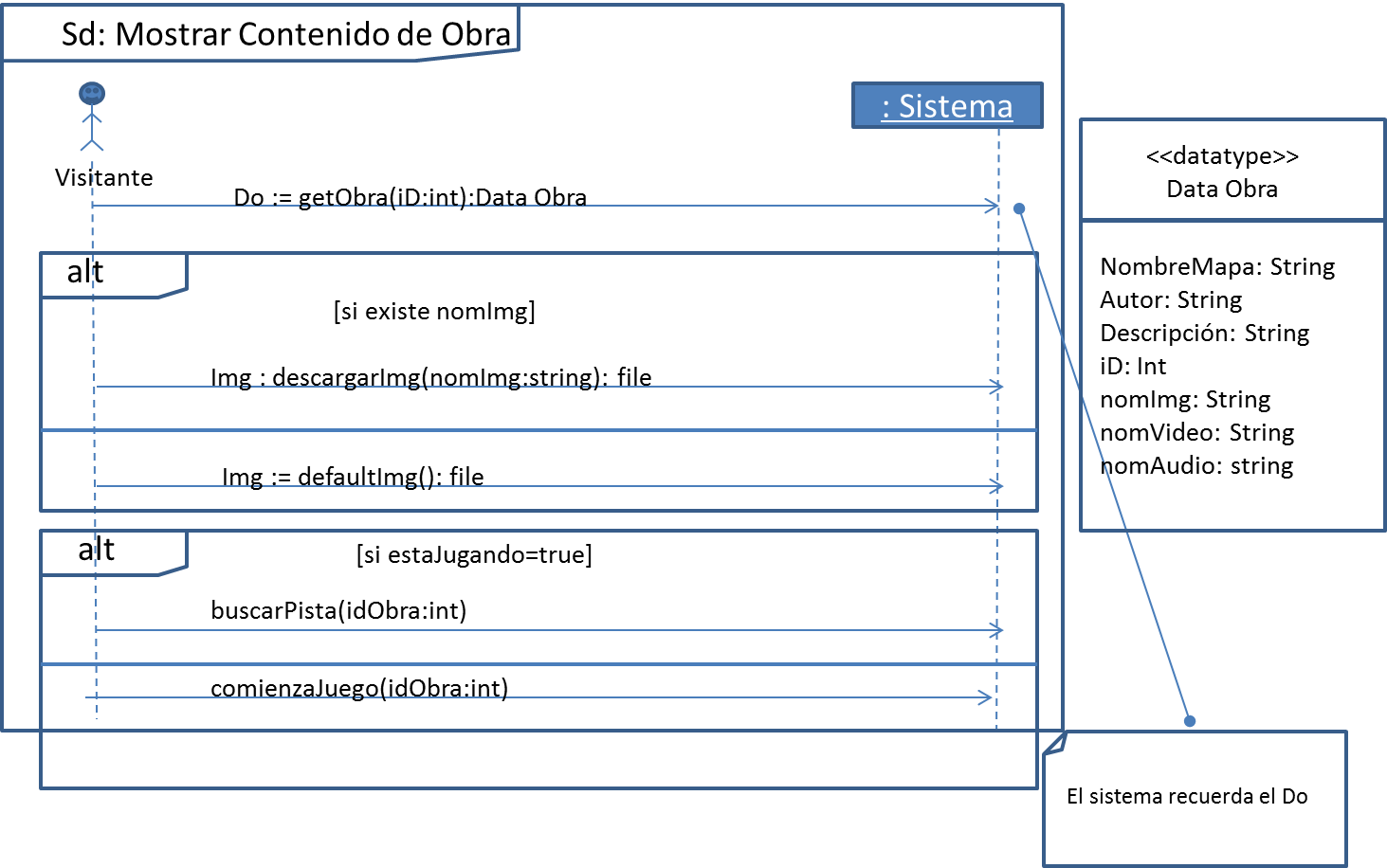


2.2.7. Mostrar Contenido de Sala

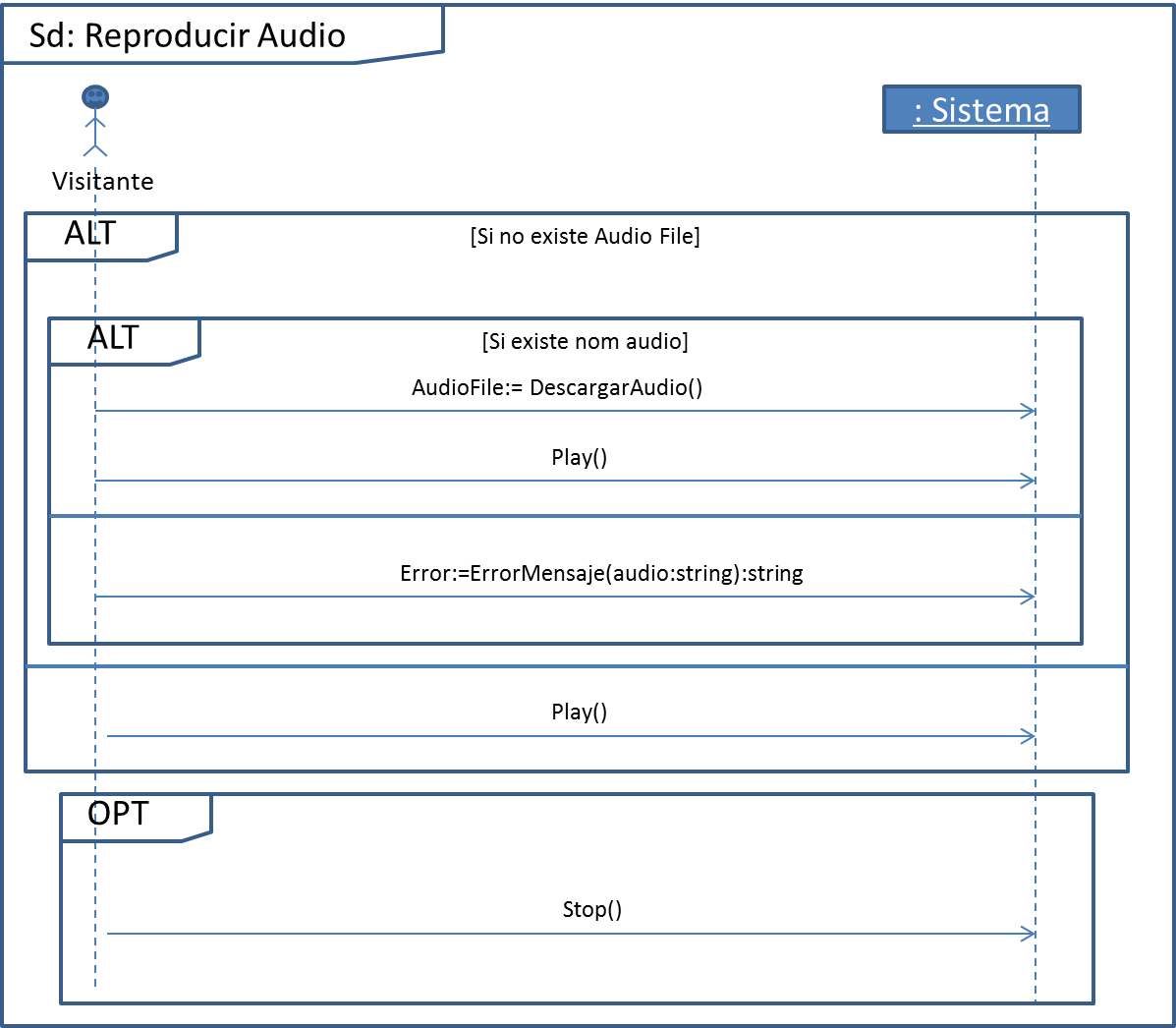
****

## 

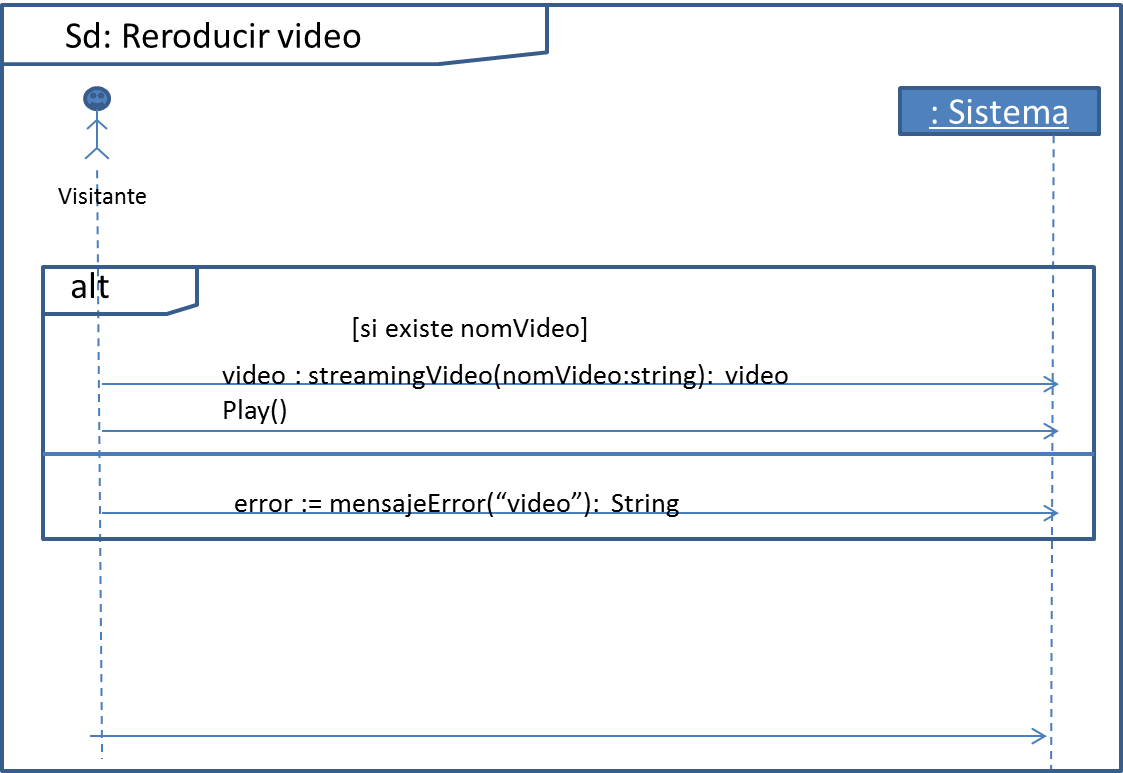
## 2.2.8. Mostrar Contenido de Obra

****

## 2.2.9. Reproducir Audio

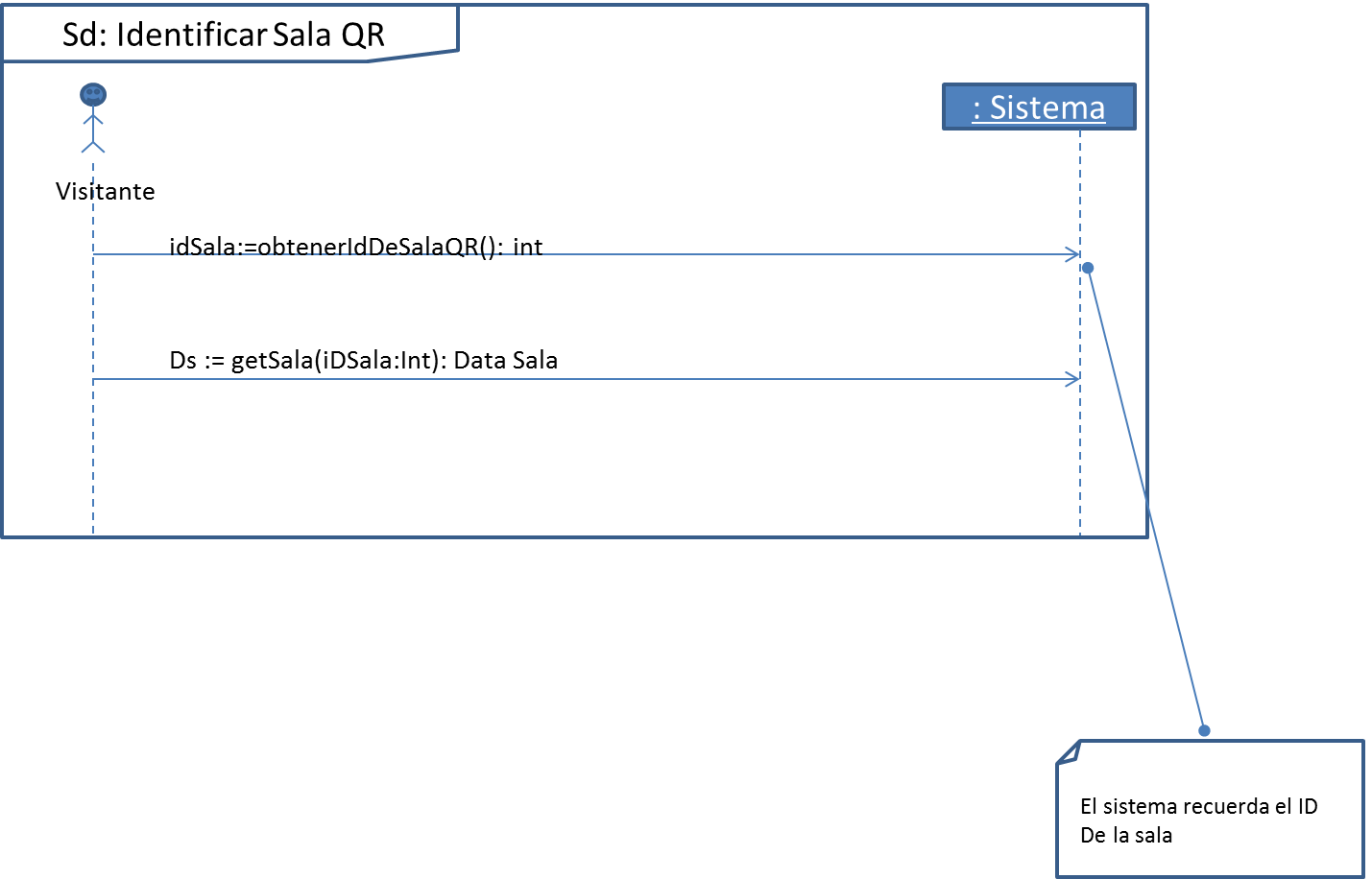
****

## 2.2.10. Reproducir Video

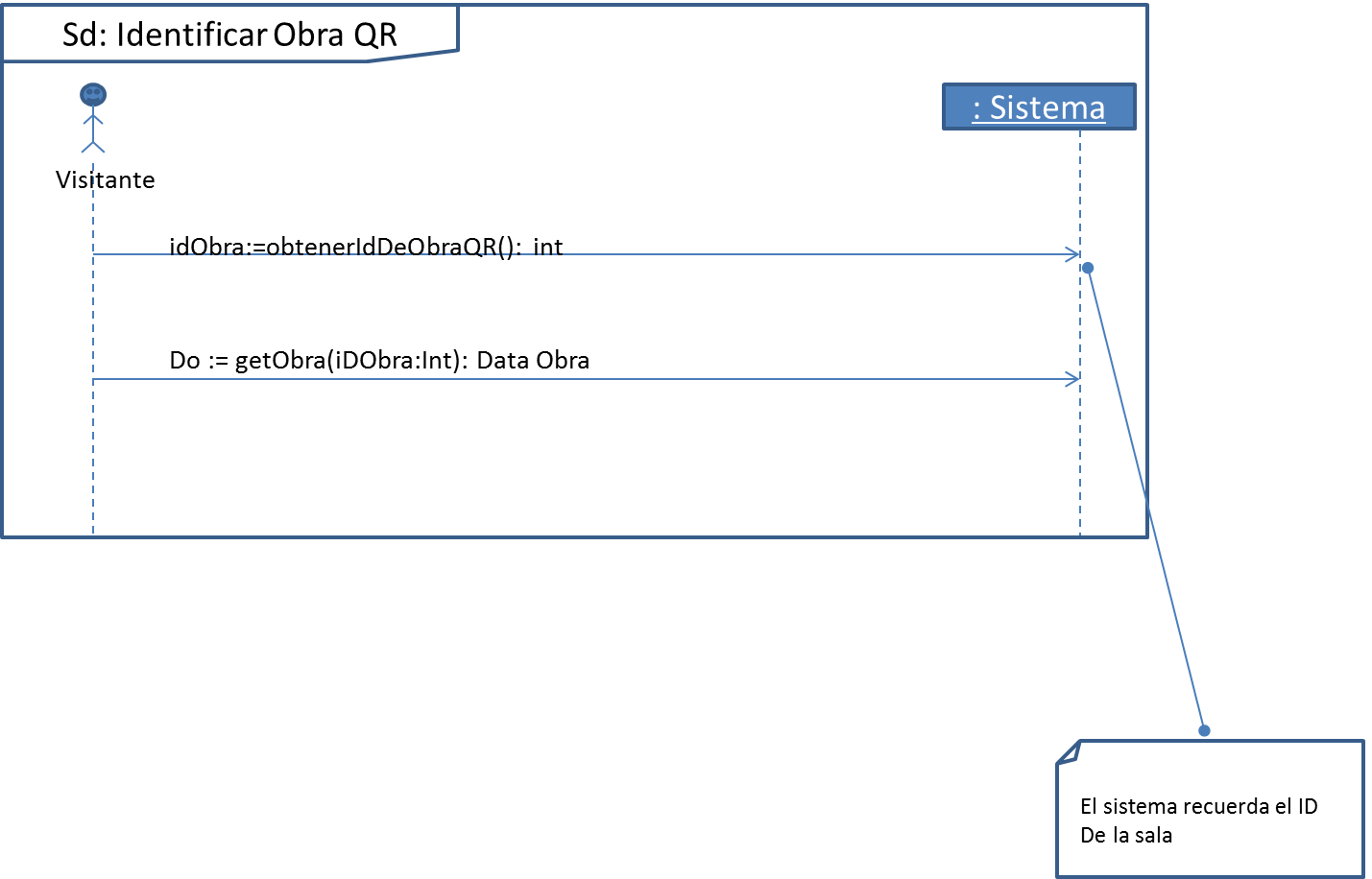
****

**Modo Automático**

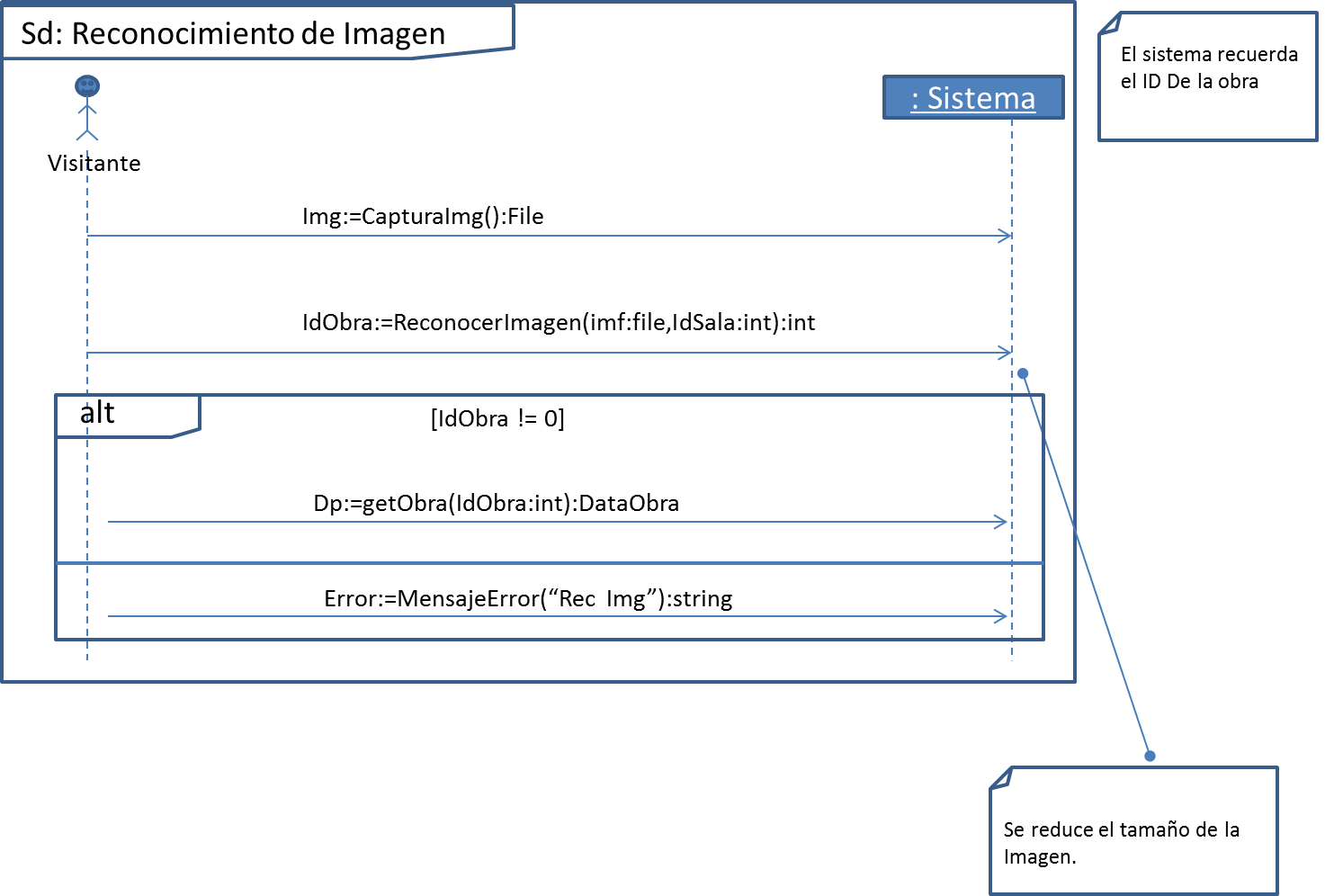
## 2.2.11. Identificar Sala QR

****

## 2.2.12. Identificar Obra QR

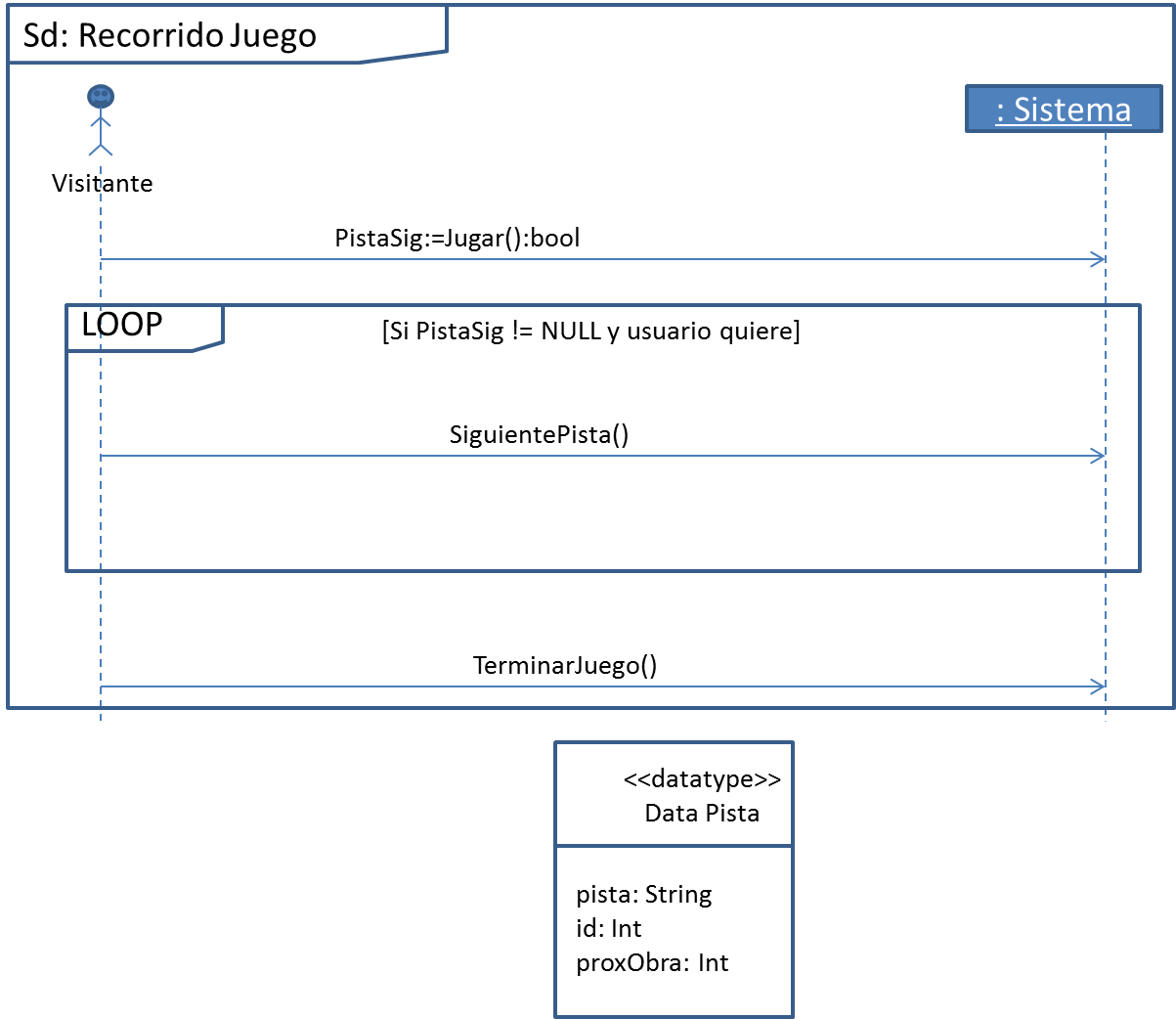
****

## 2.2.13. Reconocimiento de Imágenes

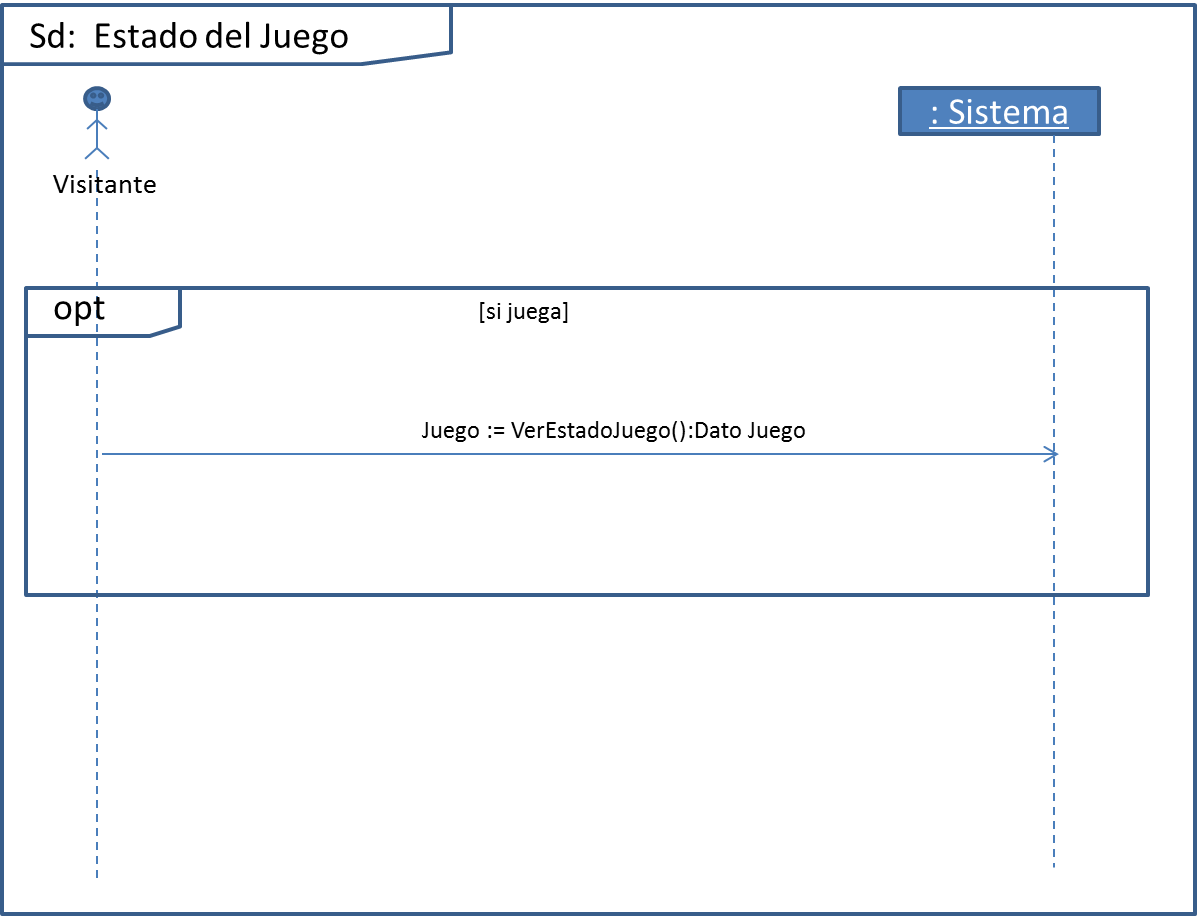
****

**Juego Interactivo**

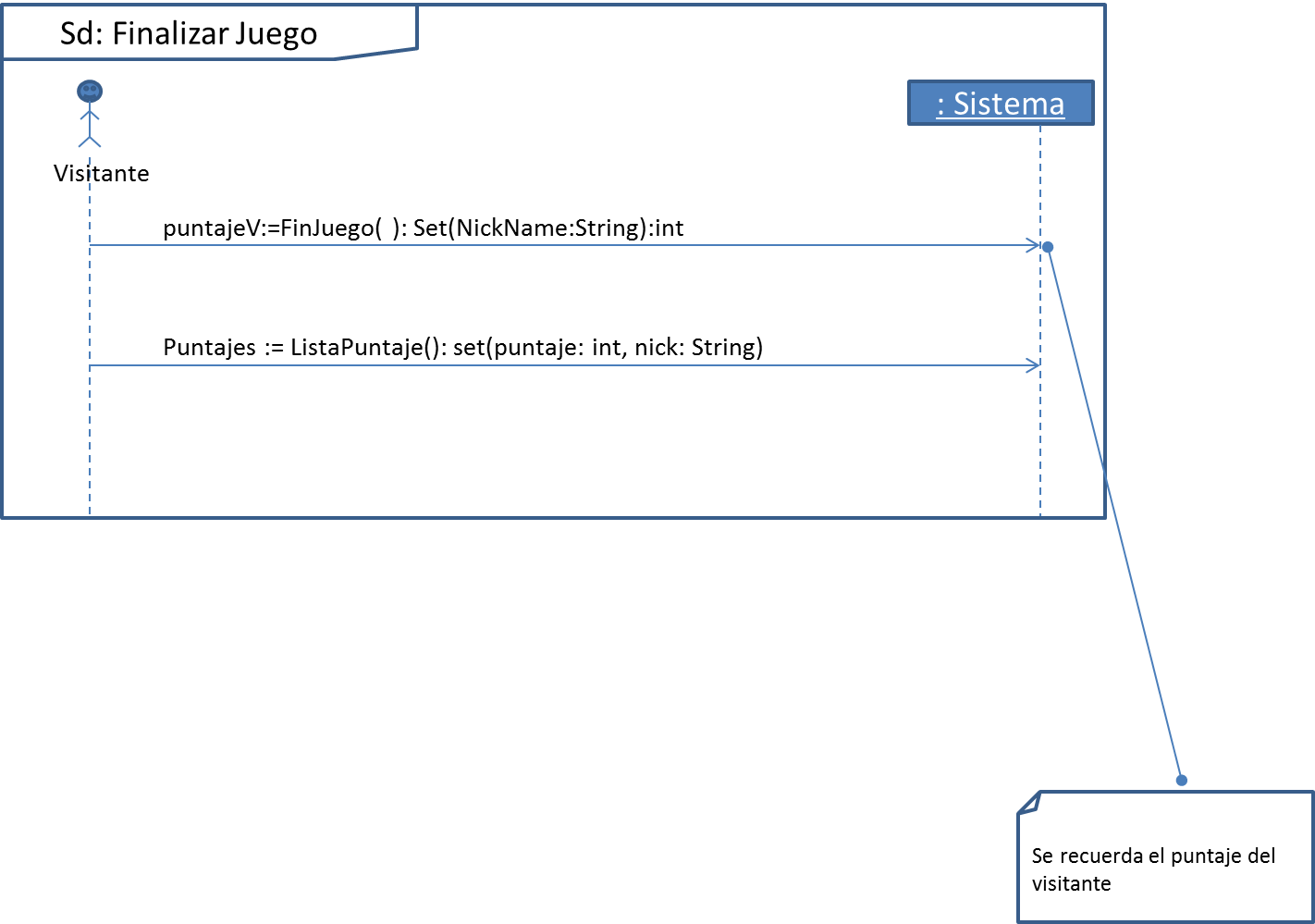
## 2.2.14. Recorrido Juego

****

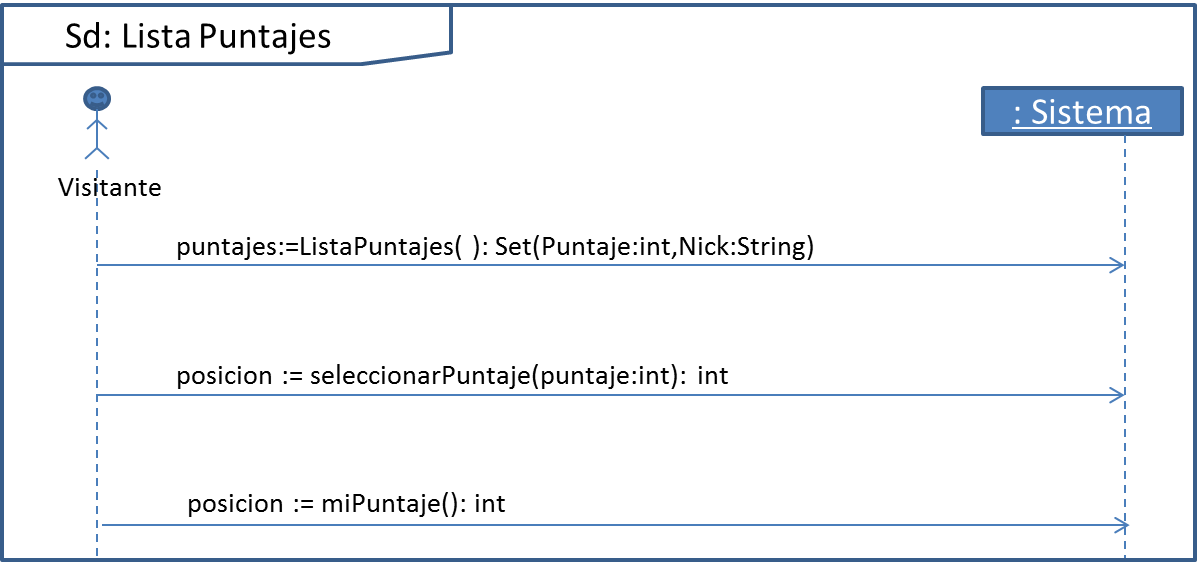
## 2.2.15. Estado del Juego

****

## 2.2.16. Finalizar Juego

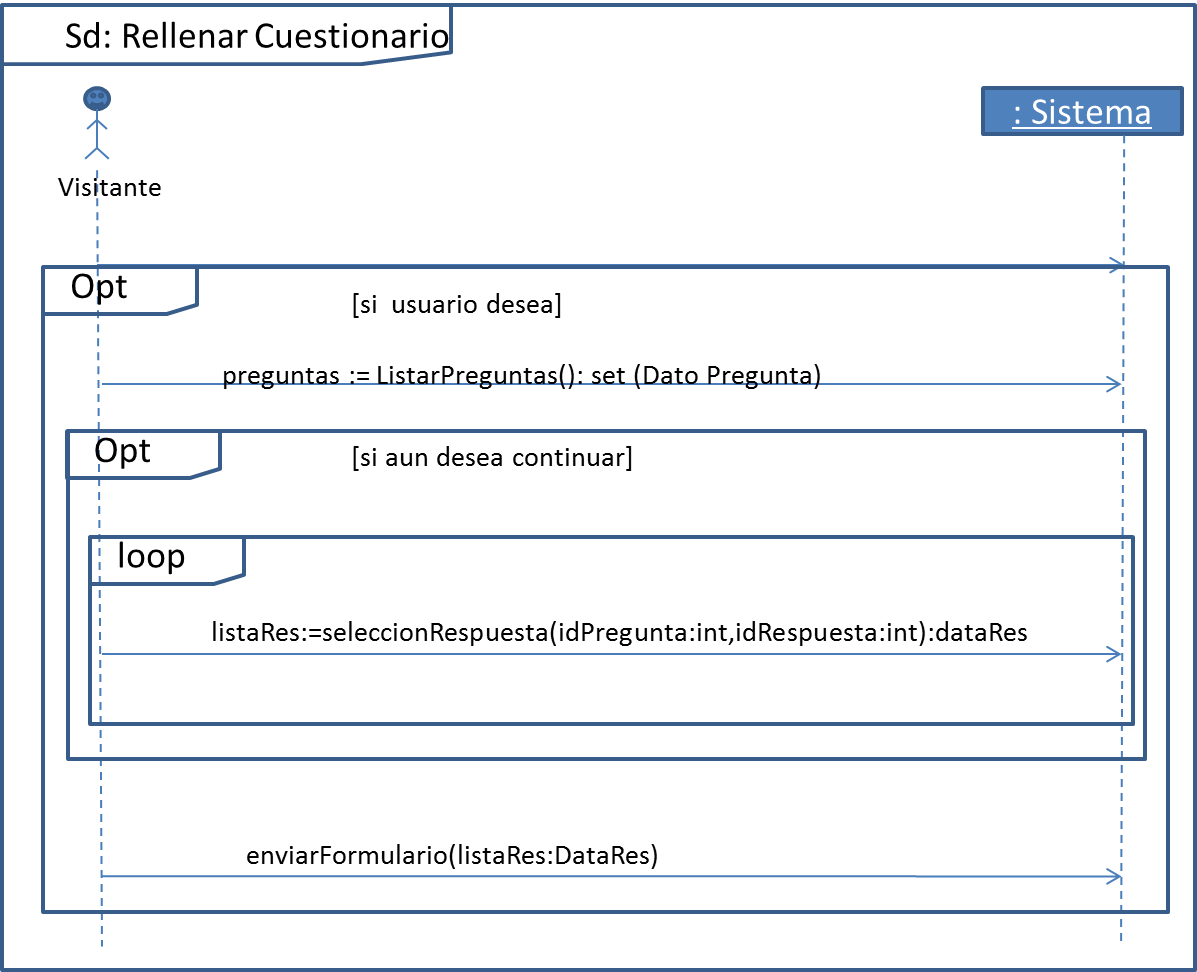
****

## 2.2.17. Lista Puntajes

****

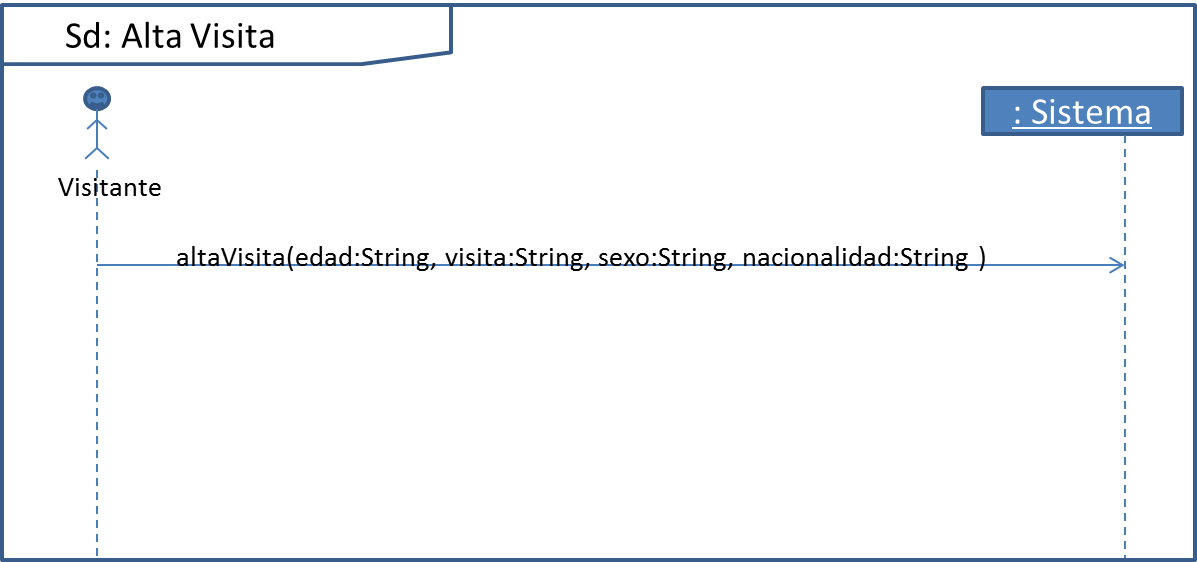
**Cuestionarios**

## 2.2.18. Rellenar Cuestionario

****

**Inicio de la Aplicación**

## **2.2.19. Alta Visita**

****

## 2.3. Contratos.

**Modo manual.**

## 2.3.1. Listar salas.

|  |  |
| --- | --- |
| **Operación** | **ListaSala()** |
|
| **Entrada** |  |
|
| **Salida** | Set del Nombre e id de Sala. |
|
| **Descripción** | Se obtienen el identificador y el nombre de todas las salas existentes. |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Pos1: Existe una lista de salas. |
|

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

## 2.3.2. Listar obras pertenecientes a salas.

|  |  |
| --- | --- |
| **Operación** | getObrasSala(idSala:int) |
|
| **Entrada** | Identificador de una sala. |
|
| **Salida** | Set del Nombre e id de obra. |
|
| **Descripción** | Se obtienen el identificador y el nombre de todas las obras existentes en la sala con id = idSala. |
|
| **Precondiciones** | Pre1: Existe en el sistema una sala con id = idSala. |
|
| **Postcondiciones** |  |
|

|  |  |
| --- | --- |
| **Operación** | seleccion(idObra:int) |
|
| **Entrada** | Identificador de una sala. |
|
| **Salida** |  |
|
| **Descripción** | Se selecciona una obra para que el sistema recuerde el nombre de la obra seleccionada. |
|
| **Precondiciones** | Pre1: Existe una lista de obras.  Pre2: Existe en el sistema una obra con Id = idObra. |
|
| **Postcondiciones** | Post1: El sistema recuerda el nombre de la obra seleccionada. |
|

## 2.3.3. Buscar Salas.

|  |  |
| --- | --- |
| **Operación** | buscarSalas(nombre:String) |
|
| **Entrada** | Nombre de una sala o parte de el. |
|
| **Salida** | Set del Nombre e id de sala. |
|
| **Descripción** | Se obtienen el identificador y el nombre de todas las salas existentes en la sala con nombre\*=nombre . |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Pos1: Existe una lista de salas. |
|

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

## 2.3.4. Mostrar contenido de sala.

|  |  |
| --- | --- |
| **Operación** | getSala(idSala:int) |
|
| **Entrada** | Identificador de una sala |
|
| **Salida** | datasala |
|
| **Descripción** | Se obtiene un datatipe que contiene los datos de la sala donde id = idSala |
|
| **Precondiciones** | Pre1: Existe en el sistema una sala con id = idSala. |
|
| **Postcondiciones** | Post1: Existe en el sistema un datasala |
|

|  |  |
| --- | --- |
| **Operación** | descargarImg(nomimg:String) |
|
| **Entrada** | Nombre de la imagen perteneciente a la sala |
|
| **Salida** | File |
|
| **Descripción** | Se descarga la imagen de la sala donde  nomimg = nombreImagen |
|
| **Precondiciones** | Pre1: Existe en el sistema el nombre de la imajen |
|
| **Postcondiciones** | Pos1: Existe un archivo que contiene la imagen |
|

|  |  |
| --- | --- |
| **Operación** | defoultImg() |
|
| **Entrada** |  |
|
| **Salida** | File |
|
| **Descripción** | Se obtiene imagen por defecto del sistema |
|
| **Precondiciones** | Pre1: Existe en el sistema una imagen predeterminada |
|
| **Postcondiciones** | Pos1: Existe un archivo que contiene la imagen |
|

## 2.3.5. Seleccionar Sala

|  |  |
| --- | --- |
| **Operación** | Seleccionar(opt:Boolean) |
|
| **Entrada** | Boolean |
|
| **Salida** |  |
|
| **Descripción** | Acceso rápido al listado de obras de la sala sin la necesidad de pasar por el contenido de la sala |
|
| **Precondiciones** | Pre1: Existe en el sistema una sala con id = idSala. |
|
| **Postcondiciones** |  |
|

## 2.3.6. Listar Obras.

|  |  |
| --- | --- |
| **Operación** | **ListaObras()** |
|
| **Entrada** |  |
|
| **Salida** | Set del Nombre e id de obra. |
|
| **Descripción** | Se obtienen el identificador y el nombre de todas las obras existentes. |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Pos1: Existe en el sistema una lista de obras |
|

|  |  |
| --- | --- |
| **Operación** | **SeleccionarObra(idObra:int)** |
|
| **Entrada** | Identificador de una sala. |
|
| **Salida** |  |
|
| **Descripción** | Se selecciona una sala para que el sistema recuerde el idSala de la sala seleccionada. |
|
| **Precondiciones** | Pre1: Existe una lista de Obras.  Pre2: Existe una sala con id = idObra. |
|
| **Postcondiciones** | Post1: El sistema recuerda el idObra de la Obra seleccionada. |
|

## 2.3.7. Buscar Obras.

|  |  |
| --- | --- |
| **Operación** | buscarObras(nombre:String) |
|
| **Entrada** | Nombre de una Obra o parte de el. |
|
| **Salida** | Set del Nombre e id de Obras. |
|
| **Descripción** | Se obtienen el identificador y el nombre de todas las Obras existentes con nombre\*=nombre . |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Pos1: Existe una lista de obra. |
|

|  |  |
| --- | --- |
| **Operación** | **SeleccionarObra(idObra:int)** |
|
| **Entrada** | Identificador de una obra. |
|
| **Salida** |  |
|
| **Descripción** | Se selecciona una sala para que el sistema recuerde el idObra de la Obra seleccionada. |
|
| **Precondiciones** | Pre1: Existe una lista de Obra.  Pre2: Existe una sala con id = idObra. |
|
| **Postcondiciones** | Post1: El sistema recuerda el idObra de la sala seleccionada. |
|

## 2.3.8. Mostrar contenido de Obra.

|  |  |
| --- | --- |
| **Operación** | getObra (idObra:int) |
|
| **Entrada** | Identificador de una Obra |
|
| **Salida** | dataObra |
|
| **Descripción** | Se obtiene el datatipe dataObra  que contiene los datos de la sala donde id = idObra |
|
| **Precondiciones** | Pre1: Existe en el sistema una obra con id = idObra. |
|
| **Postcondiciones** | Post1: Existe en el sistema un dataObra |
|

|  |  |
| --- | --- |
| **Operación** | descargarImg(nomimg:String) |
|
| **Entrada** | Nombre de la imagen perteneciente a la Obra |
|
| **Salida** | File |
|
| **Descripción** | Se descarga la imagen de la Obra donde  nomimg = nombreImagen |
|
| **Precondiciones** | Pre1: Existe en el sistema el nombre de la imajen |
|
| **Postcondiciones** | Pos1: Existe un archivo que contiene la imagen |
|

|  |  |
| --- | --- |
| **Operación** | defoultImg() |
|
| **Entrada** |  |
|
| **Salida** | File |
|
| **Descripción** | Se obtiene imagen por defecto del sistema |
|
| **Precondiciones** | Pre1: Existe en el sistema una imagen predeterminada |
|
| **Postcondiciones** | Pos1: Existe un archivo que contiene la imagen |
|

|  |  |
| --- | --- |
| **Operación** | BuscarPista(idObra) |
|
| **Entrada** | Id de la obra actual |
|
| **Salida** |  |
|
| **Descripción** | Busca la existencia de una pista para esta obra |
|
| **Precondiciones** | Pre1: Existe Obra donde idObra= idObraPista  Pre2: La constante del sistema jugando=true |
|
| **Postcondiciones** | Pos1: La constante del sistema idProximaObra=newIdProximaPista  Pos2: Constante del sistema cantObras++ |
|

|  |  |
| --- | --- |
| **Operación** | ComenzarJuego(idObra:int) |
|
| **Entrada** | Id de la obra actual |
|
| **Salida** |  |
|
| **Descripción** | Comprobar que exista juegos activos y que la obra actual pertenezca al juego |
|
| **Precondiciones** | Pre1: Existe Obra donde idObra =idObraInicio  Pre2: La constante del sistema jugando=false |
|
| **Postcondiciones** | Pos1: La constante del sistema idProximaObra=newIdProximaPista  Pos2: Constante del sistema cantObras++ |
|

## 2.3.9. Reproducir Audio

|  |  |
| --- | --- |
| **Operación** | DescargarAudio(nomAudio:string) |
|
| **Entrada** | Nombre del archivo de audio |
|
| **Salida** | File |
|
| **Descripción** | Se obtiene el archivo de audio que es descargado |
|
| **Precondiciones** | Pre1: Existe en el sistema una obra con id = idObra.  Pre2: No existe archivo con nombre nomAudio |
|
| **Postcondiciones** | Post1: Existe un archivo con nombre nomAudio |
|

|  |  |
| --- | --- |
| **Operación** | ErrorMensager(audio:string) |
|
| **Entrada** | Constante que representa la reproducción de audio |
|
| **Salida** | String |
|
| **Descripción** | Obtiene el mensaje de error correspondiente al error de audio no encontrado o inexistente |
|
| **Precondiciones** | Pre2: La variable nomAudio no contiene datos |
|
| **Postcondiciones** |  |
|

|  |  |
| --- | --- |
| **Operación** | Play() |
|
| **Entrada** |  |
|
| **Salida** |  |
|
| **Descripción** | Comienza la reproducción de audio |
|
| **Precondiciones** | Pre1: Existe archivo con nombre nomAudio |
|
| **Postcondiciones** | Post1: Existe un archivo con nombre nomAudio |
|

|  |  |
| --- | --- |
| **Operación** | Stop() |
|
| **Entrada** |  |
|
| **Salida** |  |
|
| **Descripción** | Detiene la reproducción de audio |
|
| **Precondiciones** | Pre1: Existe archivo con nombre nomAudio |
|
| **Postcondiciones** | Post1: Existe un archivo con nombre nomAudio |
|

## 2.3.10. Reproducir video

|  |  |
| --- | --- |
| **Operación** | **StrimingVideo(nomVideo:String):Video** |
|
| **Entrada** | String |
|
| **Salida** | Video |
|
| **Descripción** | Se obtiene el video mediante una conexión striming con el servidor |
|
| **Precondiciones** | Pre1: Existe nomVideo!=null |
|
| **Postcondiciones** | Pos1: Existe video en memoria |
|

|  |  |
| --- | --- |
| **Operación** | Play() |
|
| **Entrada** |  |
|
| **Salida** |  |
|
| **Descripción** | Comienza la reproducción de video |
|
| **Precondiciones** | Pre1: Existe video con nombre nomVideo |
|
| **Postcondiciones** | Post1: Existe un video con nombre nomVideo |
|

|  |  |
| --- | --- |
| **Operación** | ErrorMensager(video:string) |
|
| **Entrada** | Constante que representa la reproducción de video |
|
| **Salida** | String |
|
| **Descripción** | Obtiene el mensaje de error correspondiente al error de video no encontrado o inexistente |
|
| **Precondiciones** | Pre2: La variable nomAudio no contiene datos |
|
| **Postcondiciones** |  |
|

**Modo automático.**

## 2.3.11. Identificar sala QR.

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

|  |  |
| --- | --- |
| **Operación** | **getsSala(idSala:int)** |
|
| **Entrada** | Identificador de Sala. |
|
| **Salida** | Data type dataSala. |
|
| **Descripción** | Se obtienen los datos de la sala con id = idSala, en un data type llamado dataSala con los datos nombre de la imagen, nombre, descripcion correspondientes, etc |
|
| **Precondiciones** | Pre1: Existe en el sistema una sala con id = idSala. |
|
| **Postcondiciones** | Post1: Se identificó la sala a la que se ingresa. |

## 2.3.12. Listado de Puntajes.

|  |  |
| --- | --- |
| **Operación** | **ListaPuntajes ()** |
|
| **Entrada** |  |
|
| **Salida** | Set de puntaje y nickname del visitante |
|
| **Descripción** | Se obtiene la lista de puntajes y su Nickname del usuario de forma descendente |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Post1: Existe en memoria lista de dataPuntaje |
|

|  |  |
| --- | --- |
| **Operación** | **SeleccionarPuntaje(puntaje:int)** |
|
| **Entrada** | Puntaje:int |
|
| **Salida** | Posicion: int |
|
| **Descripción** | Se obtiene la posición del puntaje seleccionado que se refiere a la posición de otro visitante |
|
| **Precondiciones** | Pre1: Existe listado de puntajes |
|
| **Postcondiciones** |  |
|

|  |  |
| --- | --- |
| **Operación** | **MiPuntaje()** |
|
| **Entrada** |  |
|
| **Salida** | Posicion: int |
|
| **Descripción** | Se obtiene la posición del visitante actual |
|
| **Precondiciones** | Pre1: Existe listado de puntajes |
|
| **Postcondiciones** |  |
|

## 2.3.13. Identificar Obra QR.

|  |  |
| --- | --- |
| **Operación** | **obtenerIdObraDeQR()** |
|
| **Entrada** |  |
|
| **Salida** | Integer. Identificador de la Obra |
|
| **Descripción** | Se obtiene el identificador de la Obraa través del dispositivo móvil enfocando el código QR de dicha Obra. |
|
| **Precondiciones** | Pre1: Existe en el sistema una sala con id = idObra. |
|
| **Postcondiciones** |  |
|

|  |  |
| --- | --- |
| **Operación** | **getsSala(idSala:int)** |
|
| **Entrada** | Identificador de Sala. |
|
| **Salida** | Data type dataSala. |
|
| **Descripción** | Se obtienen los datos de la sala con id = idObra, en un data type llamado dataObra con los datos nombre de la imagen, nombre, descripcion correspondientes, etc |
|
| **Precondiciones** | Pre1: Existe en el sistema una obra con id = idObra. |
|
| **Postcondiciones** | Post1: Se identificó la obra a la que se ingresa. |

## 2.3.14. Reconocimiento de imagen.

|  |  |
| --- | --- |
| **Operación** | **CaptireInage()** |
|
| **Entrada** |  |
|
| **Salida** | File. |
|
| **Descripción** | Se saca una foto y se obtiene el archivo foto. |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Post1: Existe ahora una foto nueva. |
|

|  |  |
| --- | --- |
| **Operación** | **ReconocerImagen(img:file,idSala)** |
|
| **Entrada** | Archivo a subir (file) e ID de la sala |
|
| **Salida** | idObra:Integer, el ID de la obra |
|
| **Descripción** | Se sube la foto al servidor. Para que este la reconozca y pueda darnos el ID de la obra. Luego se elimina para no tener residuos |
|
| **Precondiciones** | Pre1: Existe img tal que img!=null  Pre2: Existe sala tal que id=idSala |
|
| **Postcondiciones** | Post1: No existe img |
|

|  |  |
| --- | --- |
| **Operación** | **getObra(idObra)** |
|
| **Entrada** | idObra:Integer |
|
| **Salida** | do:dataObra |
|
| **Descripción** | Se obtiene los datos de las obras contenidos en do |
|
| **Precondiciones** | Pre1: Existe obra en el sistema tal que id =idObra |
|
| **Postcondiciones** | Post1: Se identificó una obra. |
|

|  |  |
| --- | --- |
| **Operación** | ErrorMensager(ercimg:string) |
|
| **Entrada** | Constante que representa el reconocimiento de img |
|
| **Salida** | String |
|
| **Descripción** | Obtiene el mensaje de error correspondiente al error del reconocimiento de imagen fallido |
|
| **Precondiciones** | Pre1: No existe obra tal que id=idObra |
|
| **Postcondiciones** |  |
|

**Juego**

## 2.3.15. Estado del juego.

|  |  |
| --- | --- |
| **Operación** | **VerEstadoDelJuego()** |
|
| **Entrada** |  |
|
| **Salida** | Juego:Data juegos. Datos del juego actual |
|
| **Descripción** | Muestra en pantalla los datos del juego tales como pista, tiempo, obras actuales encontradas y total |
|
| **Precondiciones** | Pre1:En el sistema existe la constante jugando=true |
|
| **Postcondiciones** |  |
|

## 2.3.16. Finalizar Juego.

|  |  |
| --- | --- |
| **Operación** | **FinJuego(nickname:String)** |
|
| **Entrada** | nickname:String. Nombre que figurara en la lista de puntajes |
|
| **Salida** | Puntaje:int. |
|
| **Descripción** | Muestra en pantalla los datos del cuego tales como pista, tiempo, obras actuales encontradas y total |
|
| **Precondiciones** | Pre1:En el sistema existe la constante jugando=true |
|
| **Postcondiciones** |  |
|

|  |  |
| --- | --- |
| **Operación** | **ListarPuntaje()** |
|
| **Entrada** |  |
|
| **Salida** | Lista de nickname y puntajes de todos los visitantes |
|
| **Descripción** | Muestra en pantalla un listado de los puntajes generados por los visitantes en orden descendente |
|
| **Precondiciones** |  |
|
| **Postcondiciones** |  |
|

## 2.3.17. Recorrido del Juego

|  |  |
| --- | --- |
| **Operación** | Jugar () |
|
| **Entrada** |  |
|
| **Salida** | Boolean |
|
| **Descripción** | Verifica que la obra actual pertenezca al juego y a la pista correspondiente |
|
| **Precondiciones** | Pre1:En el sistema existe la constante jugando=false  Pre2:Existe idObra=idPista |
|
| **Postcondiciones** | Pos1:En el sistema existe la constante jugando=true |
|

|  |  |
| --- | --- |
| **Operación** | **SiguientePista()** |
|
| **Entrada** |  |
|
| **Salida** |  |
|
| **Descripción** | Comprueba que la obra en la que se encuentra, pertenezca a la pista correspondiente salvándola y aumentando el contador de obras encontradas |
|
| **Precondiciones** | Pre1: En el sistema existe la constante jugando=true  Pre2: La siguiente pista debe ser PistaSig!=null |
|
| **Postcondiciones** | Post1: Existe una nueva referencia en PistaSig |
|

|  |  |
| --- | --- |
| **Operación** | **TerminarJuego()** |
|
| **Entrada** |  |
|
| **Salida** |  |
|
| **Descripción** | Finaliza el juego, este continua en el caso de uso FinalizarJuego |
|
| **Precondiciones** | Pre1: En el sistema existe la constante jugando=true  Pre2: La siguiente pista debe ser PistaSig=null |
|
| **Postcondiciones** |  |
|

**Cuestionario**

## 2.3.18. Rellenar Cuestionario

|  |  |
| --- | --- |
| **Operación** | ListarPregunta() |
|
| **Entrada** |  |
|
| **Salida** | Lista de DataPregunta |
|
| **Descripción** | Visualiza el listado de preguntas con sus correspondientes respuestas |
|
| **Precondiciones** | Pre1: Existe cuestionario activo en servidor |
|
| **Postcondiciones** | Pos1: Existe listado de dataPregunta |
|

|  |  |
| --- | --- |
| **Operación** | **SeleccioneRespuesta(idPregunta:int)** |
|
| **Entrada** | idPregunta:int e idRespuesta:int |
|
| **Salida** | dr:dataRes (datos de la respuesta seleccionada) |
|
| **Descripción** | Salva la respuesta seleccionada del usuario |
|
| **Precondiciones** | Pre1: Existe respuesta donde id=idRepuesta  Pre2: Existe Pregunta donde id=idPregunta |
|
| **Postcondiciones** | Pre1: El istado de dataRes del sistema contiene haora a dr |
|

**Inicio del recorrido**

## 2.3.19. Alta Visitante.

|  |  |
| --- | --- |
| **Operación** | **AltaVisita(edad:String,visita:String,sexo:String,**  **nacionalidad:String)** |
|
| **Entrada** | edad:String, visita:String, sexo:String,  nacionalidad:String |
|
| **Salida** |  |
|
| **Descripción** | Recoge los datos del usuario y los utiliza para dar de alta un visitante con esos datos, en caso de que este no los indique se crea aunque estos datos estarán varios |
|
| **Precondiciones** |  |
|
| **Postcondiciones** | Pos1:Se crea la sesión del usuario  Pos2:Exste una nueva instancia de visitante en el servidor |
|

## 2.4. Código fuente.

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

Determinando la dirección del WS y cual consumir se le envían los parámetros adecuados y necesarios para luego retórnalos en una cadena de caracteres

**Código en Java:**

**public** String setFormulario(**int** id\_visitante, String datos){

String SOAP\_ACTION = "http://10.0.2.109/server\_php/server\_php.php/setCuestionarioVisitante";

String NAMESPACE = "http://10.0.2.109/server\_php/";

String METHOD\_NAME = "setCuestionarioVisitante";

String URL = "http://10.0.2.109/server\_php/server\_php.php?wsdl";

SoapObject request = **new** SoapObject(NAMESPACE, METHOD\_NAME);

// ingresa parámetros que se resiviran en el WS

request.addProperty("id\_visitante", id\_visitante);

request.addProperty("id\_result", datos);

SoapSerializationEnvelope envelope = **new** SoapSerializationEnvelope(

SoapEnvelope.*VER11*);

envelope.setOutputSoapObject(request);

envelope.dotNet = **true**;

**try** {

HttpTransportSE androidHttpTransport = **new** HttpTransportSE(URL);

// llamado al WS

androidHttpTransport.call(SOAP\_ACTION, envelope);

// obtener el SoapResult del envelope (respuesta del WS)

SoapObject result = (SoapObject) envelope.bodyIn;

**if** (result != **null**) {

// retornamos el contenido del SoapResult

**return** result.getProperty(0).toString();

} **else** {

**return** "error:=>no se encontrosala=>";

}

}**catch**(SocketTimeoutException e){

**return** "TiemOut Exeption: " + e.toString();

}**catch** (Exception e) {

**return** "error =>" + e.toString();

}

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

El servidor contiene un servicio de ftp que se divide en 3 clientes distintos, que de los cuales se necesita de un inicio de sesión por cada uno.

**Código en Java:**

**private** **boolean** isObraLogin = **false**, isSalaLogin = **false**, isZonaLogin = **false**;

**public** FTPClient salaClient = **null**, obraClient = **null**, zonaClient = **null**;

String host = "10.0.2.109";

//Salas

String sUser = "salas", sPassword = "12345678";

//Obras

String oUser = "obras", oPassword = "12345678";

//Zonas

String zUser = "zonas", zPassword = "12345678";

Context context;

MyFTP(Context c){

context = c;

}

**public** **boolean** LoginObras(){

**boolean** status = **false**;

**try**{

obraClient = **new** FTPClient();

obraClient.connect(host, 21);

// Comprueba el código de respuesta

**if** (FTPReply.*isPositiveCompletion*(obraClient.getReplyCode())) {

// Login con usuario y contraceña

status = obraClient.login(oUser, oPassword);

isObraLogin = status;

obraClient.setFileType(FTP.*BINARY\_FILE\_TYPE*);

obraClient.enterLocalPassiveMode();

}

} **catch**(Exception e) {

System.*out*.println(e);

**return** **false**;

}

**return** status;

}

# 3. Glosario.

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

Algoritmos: es un conjunto prescrito de instrucciones o reglas bien definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad.

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-Servidor: es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes.

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 ditribuido: 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.

GoogleDrive: es un servicio de alojamiento de archivos.

HTML: Siglas de *HyperText Markup Language* («lenguaje de marcas de hipertexto»), hace referencia al lenguaje de marcado para la elaboración de páginas web.

iOS: es un sistema operativo móvil de la empresa Apple Inc. Multimedia Es cualquier combinación de texto, arte gráfico, sonido, animación y vídeo que llega a nosotros por computadora u otros medios electrónicos.

IPads: es una línea de tabletas diseñadas y comercializadas por Apple Inc.

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

Java Emulador: es un software que permite ejecutar programas o videojuegos en una plataforma (sea una arquitectura de hardware o un sistema operativo) diferente de aquella para la cual fueron escritos originalmente.

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.

Máquina Virtual (JVM): es una máquina virtual de proceso nativo, es decir, ejecutable en una plataforma específica, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial, el cual es generado por el compilador del lenguaje.

Modelo Iterativo e Incremental: Modelo de desarrollo que consta de diversas etapas, en cada una de ellas con un incremento Github: es una forja para alojar proyectos utilizando el sistema de control de versiones Git.

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, en 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.

Renderizar: proceso de generar una imagen o vídeo mediante el cálculo de iluminación GI partiendo de un modelo en 3D

Root o Super-Usuario: se les considera a aquiellos usuarios con todos los permisos posibles en el sistema.

RgoUML: es una aplicación de diagramado de UML Dia: es una aplicación de diagramado de UML orientado a objetos: es un paradigma de programación que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.

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.

Smartphones: o también llamado teléfono inteligente es un término comercial para denominar a un teléfono móvil que ofrece más funciones que un teléfono móvil común.

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.

Tablets: es una computadora portátil de mayor tamaño que un teléfono inteligente o una PDA, integrada en una pantalla táctil (sencilla o multitáctil) con la que se interactúa primariamente con los dedos o un estilete (pasivo o activo), sin necesidad de teclado físico ni ratón.

Tags, marcadores o matches: son características reconocibles de cada imagen.

Testing: pruebas que se le realizan al software con el objetivo de encontrar errores y solucionarlos antes de la versión final.

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.

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

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.

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.