

UNIVERSIDAD DE BUENOS AIRES

FACULTAD DE INGENIERA

75.52 TALLER DE PROGRAMACIÓN II

---

## Recursos

Capa de Negocios

---

*Integrantes:*

Hugo CHAVAR - 90541

Damián MANOFF - 93169

Yamila GLINSEK - 93219

Andrés SANABRIA - 93403

*capanegocio.recursos@yahoo.com.ar*

23 de enero de 2014

# Índice

<b>1. Introducción</b>	<b>2</b>
<b>2. Definiciones</b>	<b>2</b>
<b>3. Clases</b>	<b>2</b>
3.1. Archivos . . . . .	2
3.2. Recurso . . . . .	3
3.3. Encuestas . . . . .	3
3.4. Pregunta . . . . .	5
3.5. PreguntaRespuestaFija . . . . .	5
3.6. Encuesta Respondida . . . . .	6
3.7. Pregunta Respondida . . . . .	7
3.7.1. Respuesta Fija Respondida . . . . .	7
3.7.2. Pregunta Respuesta a Completar Respondida . . . . .	7
3.8. Links . . . . .	7
<b>4. Diagrama de Clases</b>	<b>9</b>
<b>5. Interfaces</b>	<b>10</b>

# 1. Introducción

Recursos da referencia a lo que podemos traducir como Archivos, Links y Encuestas. Los servicios que brindamos son:

1. Posibilitar la subida y descarga de archivos al sistema.
2. Crear encuestas, que pueden ser o no *Evaluadas*.
3. Permitir el acceso a responder encuestas y realizar su evaluación según el criterio establecido por el creador de la misma.
4. Linkear páginas externas y/o internas de la web.

# 2. Definiciones

Cosas que ya se han definido hasta el momento por la interacción con los grupos y docentes:

- \* Se utilizará como IDE *Eclipse* y plataforma *Java 7*.
- \* La comunicación entre capas se realiza por Web Services.
- \* Los archivos van a ser pasados entre capas como multi-part (o similar) a través de dichos Web Services.
- \* Tanto con la capa de arriba (Presentación) como la de abajo (Integración) implementarán la interfaz en XML.
- \* Los archivos serán pasados de nuestra capa a la *capa de Integración* para que ésta trate con el FileSystem o con la *capa de Acceso a datos* como ha de ser almacenado. Queda por definirse si es necesario una conversación previa entre capas antes de la transferencia del archivo.

# 3. Clases

## 3.1. Archivos

Para el manejo de archivos se tendrá la siguiente clase:

Archivo
path : string tamaño : integer tipo : string nombre : string
Guardar(Archivo) Obtener() filtrar(tipo)

Figura 1: Clase Archivo

## Atributos

**path** que es una cadena que cuenta con la dirección en la cual se encuentra guardado en el file system el archivo.

- Tipo De Datos : String
- Longitud : 100

**tamaño** es el peso del archivo en Megabytes, por lo tanto es solo numérico. Consideraremos un máximo de 100 MB para cada archivo. Por lo tanto la longitud máxima del campo será 4 caracteres.

- Tipo De Datos : Integer
- Longitud : 10

**tipo** se refiere a la extensión del Archivo (por ejemplo: mp3, mpeg4, doc).

- Tipo De Datos : String
- Longitud : 5

**nombre** es la identificación del archivo.

- Tipo De Datos : String
- Longitud : 40

## 3.2. Recurso

<i>Recurso</i>
idRecurso : Integer idAmbiente : Integer descripcion : String
Guardar(Recurso) Obtener() Crear()

Figura 2: Clase Recurso

### Atributos

**descripcion** Es el texto que verá el usuario, sobre el cual puede hacer clic.

- Tipo De Datos : String
- Longitud : 100

**idAmbiente** Ambiente al cual pertenece el recurso.

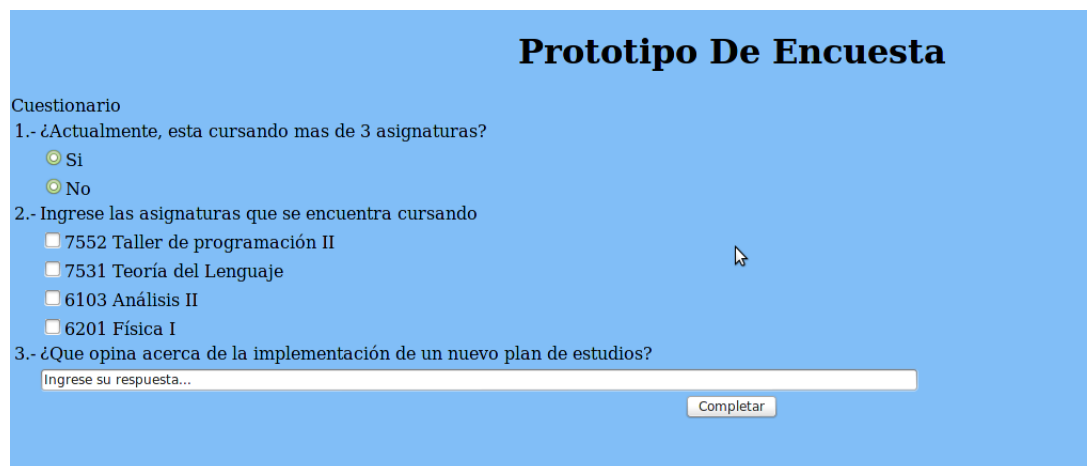
- Tipo De Datos : Integer
- Longitud : 10

**idRecurso** Identificador del recurso.

- Tipo De Datos : Integer
- Longitud : 10

## 3.3. Encuestas

La idea de encuesta es algo del siguiente tipo:



**Prototipo De Encuesta**

Cuestionario

1.- ¿Actualmente, esta cursando mas de 3 asignaturas?

☒ Si

☐ No

2.- Ingrese las asignaturas que se encuentra cursando

☐ 7552 Taller de programación II

☐ 7531 Teoría del Lenguaje

☐ 6103 Análisis II

☐ 6201 Física I

3.- ¿Que opina acerca de la implementación de un nuevo plan de estudios?

Ingrese su respuesta...

Completar

En donde claramente se pueden distinguir 3 tipos diferentes de preguntas:

1. Preguntas que solo admiten una opción como respuesta.
2. Preguntas que admiten multiples opciones como respuesta.
3. Preguntas cuya respuesta es a completar.

Además existirán dos tipos de encuestas:

1. Encuestas Evaluadas
2. Encuestas No Evaluadas.

Con respecto al primer tipo de encuestas solo podremos admitir preguntas del tipo 1 y 2 ya que se dificulta la tarea de evaluar respuestas a completar.

Las encuestas del segundo tipo admitirán todo tipo de preguntas.

Encuesta
evaluada : Boolean preguntas : List<Preguntas>
guardar() agregarPregunta(Pregunta) eliminarPregunta(nPregunta) evaluar()

Figura 3: Clase Encuesta

## Atributos

**evaluada** atributo de tipo booleano que se define en el constructor, es decir, *new Encuesta(evaluada=true)*, por defecto las encuesta son no evaluadas.

- Tipo De Datos : Boolean
- Longitud : 1

**preguntas** es una lista de preguntas. Como máximo se aceptaran **30 preguntas por encuesta**. El id de cada pregunta estará relacionado con la posición que ocupen en la lista más uno, es decir *preguntas[0]* será la pregunta número 1 asociada a esa encuesta.

- Tipo De Datos : List<Pregunta>
- Longitud : 30

Dicho esto tendremos que definir las preguntas que tendremos. En nuestro caso habrá dos tipos de preguntas

1. **Pregunta** : Es la clase padre donde la respuesta esperada podría ser múltiple o bien una descripción.
2. **PreguntaRespuestaFija** : son aquellas preguntas en donde el que la crea define las respuestas posibles, y el que las responde tiene la posibilidad de elegir una o mas respuestas, según lo defina el creador. En el prototipo son las preguntas 1 y 2.

### 3.4. Pregunta

<i>Pregunta</i>
Enunciado : String idPregunta : Integer respuestaACompletar : bool
Crear(enunciado)

Figura 4: Clase Pregunta

#### Atributos

**enunciado** será la pregunta en sí.

- Tipo De Datos : String
- Longitud : 200

**idPregunta** el id de la pregunta corresponde a un entero que identifique cada pregunta dentro de una encuesta

- Tipo De Datos : Integer
- Longitud : 2

**respuestaACompletar** es un booleano que muestra si una respuesta es a completar o es de opción múltiple. Por defecto todas las preguntas son a completar. En caso contrario se deberá crear una *PreguntaRespuestaFija*

- Tipo De Datos : Boolean
- Longitud : 1

### 3.5. PreguntaRespuestaFija

#### Atributos

**múltiplesRespuestas** define si la pregunta aceptará mas de una respuesta. Por defecto las múltiples respuestas están deshabilitadas.

- Tipo De Datos : Boolean
- Longitud : 1

**respuestasPosibles** es una lista de tamaño máximo 6, es decir una pregunta a lo sumo podrá tener 6 respuestas correctas. Las respuestas se distinguen según la posición que ocupan dentro de la lista, es decir si `respuestaPosible[0] = Fija`, la respuesta fija estará asociada a la primera posición de la lista.

Vale aclarar que si las múltiples respuestas están apagadas, la lista contendrá si y solo si una respuesta, y no permitirá agregar más.

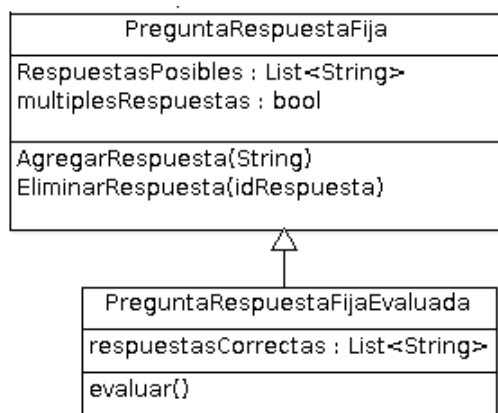


Figura 5: Clase PreguntRespuestaFija

- Tipo De Datos : List <String>
- Longitud : 6

A la vez, como este tipo de preguntas es posible evaluar habrá una clase **PreguntRespuestaFijaEvaluada** la cual tendrá las opciones correctas que se esperan.

- RespuestasCorrectas : esta lista contendrá los/el id de las respuestas que se consideran correctas. Es decir el tamaño será desde 0 a la cantidad de Respuestas agregadas en la instancia anterior. Como valor máximo se especificará el valor 6 en el caso de que todas las respuestas sean correctas.
  - Tipo De Datos : List<Integer>
  - Longitud : 6

### 3.6. Encuesta Respondida

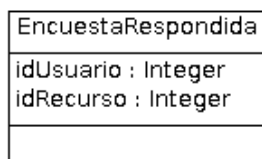


Figura 6: Clase EncuestaRespondida

#### Atributos

**idUsuario** permite relacionar la encuesta respondida con el usuario que se encargó de completarla.

- Tipo De Datos : Integer
- Longitud : 20

**idRecurso** Identificador del recurso perteneciente a la encuesta que responde la EncuestaRespondida.

- Tipo De Datos : Integer
- Longitud : 10

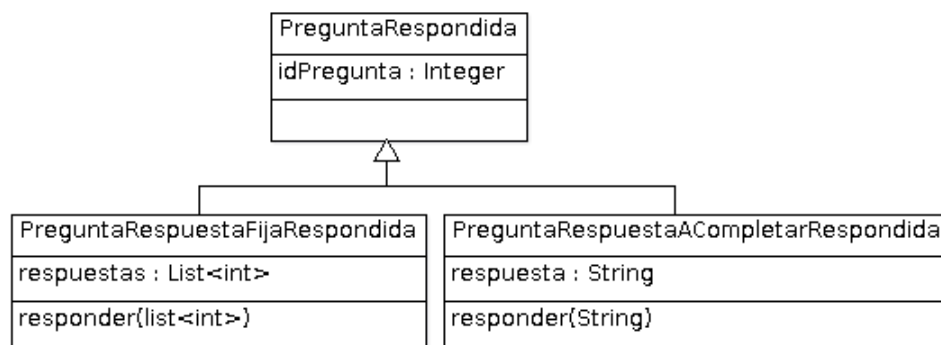


Figura 7: Clase PreguntaRespondida

### 3.7. Pregunta Respondida

#### Atributos

**idPregunta** identificador de la pregunta dentro de la encuesta.

- Tipo De Datos : Integer
- Longitud : 10

A su vez, las preguntas respondidas podrán ser fijas o a completar, siendo la forma de responderlas en ambos casos distintas.

#### 3.7.1. Respuesta Fija Respondida

**respuestas** por tratarse de posibles respuestas a las preguntas de respuesta fija de una Encuesta, este atributo es también una lista de tamaño máximo 6, es decir, se podrá responder como máximo 6 respuestas. Las respuestas se distinguen según la posición que ocupan dentro de la lista de la Encuesta original.

- Tipo De Datos : List <Integer>
- Longitud : 6

#### 3.7.2. Pregunta Respuesta a Completar Respondida

**respuesta** Atributo que guarda la respuesta proporcionada por el usuario respondente. Al ser una respuesta a completar, se admite un único texto.

- Tipo De Datos : String
- Longitud : 100

### 3.8. Links

Los links serán las referencias a otros sitios webs o referencias a artículos dentro de este mismo sitio.

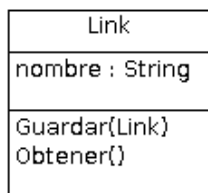


Figura 8: Clase Link

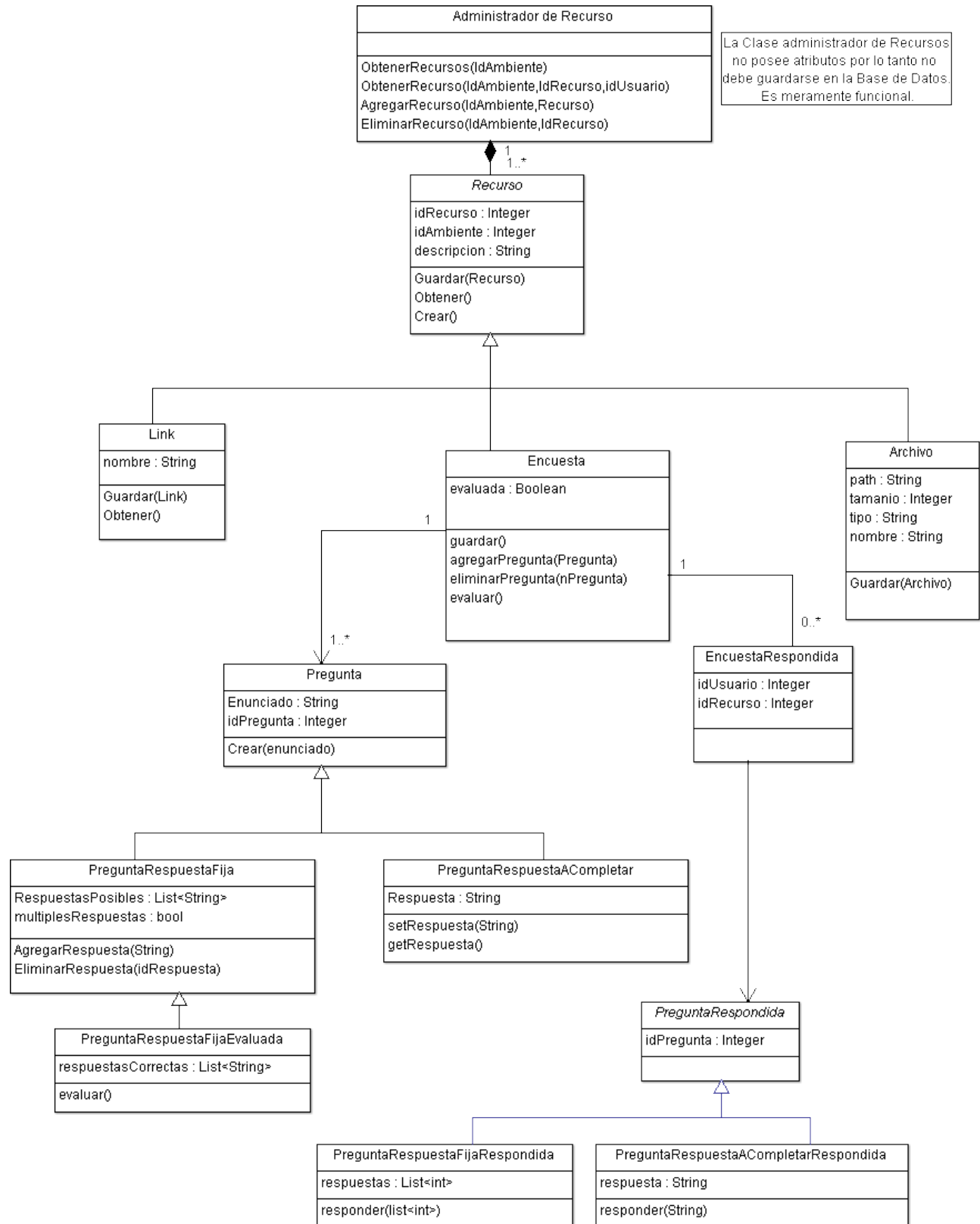


## Atributos

**nombre** es el path adonde apunta el hipervínculo.

- Tipo De Datos : String
- Longitud : 200

## 4. Diagrama de Clases



## 5. Interfaces

### Provistas a la capa de Presentación

- \* *OBTENER RECURSO*: Este método devuelve ya sea un Archivo, un Link o una Encuesta segun lo que se solicite. Es requisito tener el ID de recurso a consultar por lo que primero deben obtenerse con *OBTENER LISTA RECURSOS*.

**INPUT** XML\_PARAMETROS

**OUTPUT** XML\_RESULTADO

Como parámetro se necesitan los siguientes items detallados de la siguiente forma:

```
<parametro>
  <recurso>
    <recursoId>INT idDelRecurso</recursoId>
    <tipo>STRING tipo (ARCHIVO, LINK, ENCUESTA)</tipo>
  </recurso>
</parametro>
```

Como resultado se obtendrán los siguientes archivos dependiendo la respuesta del servicio:

- ARCHIVO

```
<response>
  <success>true</success>
  <archivo>
    <tipo>Archivo</tipo>
    <ambitoId>14</ambitoId>
    <descripcion>Archivon de texto</descripcion>
    <tipoArchivo>txt</tipoArchivo>
    <nombreArchivo>Auxiliar</nombreArchivo>
    <rawFile>bGFsdWphbjo0TWNibjIvUGNTd3JJ0jU40jUwMTpBYTovaG9tZS9sYW</rawFile>
  </archivo>
</response>
```

Donde el tag *rawFile* contiene un archivo serializado con JAXB, el tipo de dato que devuelve es un *DataHandler*, el cliente para deserializar debe poseer una clase con el siguiente atributo:

```
@XmlElement(name = "rawFile")
@XmlMimeType("application/octet-stream")
private DataHandler dataHandler;
```

- LINK

```
<response>
  <success>true</success>
  <recurso>
    <recursoId>1002</recursoId>
    <tipo>Link</tipo>
    <ambitoId>-1</ambitoId>
    <descripcion>un link a google copado</descripcion>
    <link>www.google.com.ar</link>
  </recurso>
</response>
```

- ENCUESTA NO EVALUABLE

```
<response>
  <success>true</success>
  <encuesta evaluada="false">
    <recursoId>11003</recursoId>
    <tipo>Encuesta</tipo>
    <ambitoId>-1</ambitoId>
    <descripcion>una encuesta chica</descripcion>
    <preguntas>
      <preguntaConOpciones multiplesCorrectas="false" idPregunta="1" enunciado="
        cuantas materias te faltan para recibirte?">
        <respuestas>1,menos de 5,entre 5 y 10,mas de 10</respuestas>
      </preguntaConOpciones>
      <preguntaConOpciones multiplesCorrectas="false" idPregunta="2" enunciado="
        que materia fue la mas dificil?">
        <respuestas>analisis 2,algebra 2,taller 1,org de datos</respuestas>
      </preguntaConOpciones>
    </preguntas>
  </encuesta>
</response>
```

- ENCUESTA EVALUABLE

```
<response>
  <success>true</success>
  <encuesta evaluada="true">
    <recursoId>11004</recursoId>
    <tipo>Encuesta</tipo>
    <ambitoId>-1</ambitoId>
    <descripcion>una encuesta grande</descripcion>
    <preguntas>
      <preguntaConOpciones multiplesCorrectas="false" correctas="3" idPregunta="
        1" enunciado="de que color es el caballo blanco de san martin?">
        <respuestas>rojo,verde,azul,blanco</respuestas>
      </preguntaConOpciones>
      <preguntaConOpciones multiplesCorrectas="true" correctas="0,7,10,11,12,6"
        idPregunta="2" enunciado="Un test unitario debe presentar las
        siguientes caractersticas">
        <respuestas>Rapido,Moldeable,Configurable,Acoplable,Lento,Extensible,
          Repetible,Profesional,Maduro,Amplio,Simple,Independiente,
          Automatizable</respuestas>
      </preguntaConOpciones>
      <preguntaSinOpciones correcta="4" idPregunta="5" enunciado="cuantas patas
        tiene un gato?">
    </preguntas>
  </encuesta>
</response>
```

Notar que existen dos tags para las preguntas *preguntaConOpciones* y *preguntaSinOpciones* para diferenciar las que deben ser completadas en un TextBox de las que usan un mecanismo para tildar la/s opcion/es correcta/s. Dentro de *preguntaConOpciones* se diferencian por el atributo *multipleCorrectas*, para que sea posible elegir una o muchas, esto hay que hacerlo debido a que una pregunta puede ser marcada como *multipleCorrectas* = "true" pero tenga una sola que es correcta. Se enviará *multipleCorrectas* = "false" cuando no hayan respuestas correctas, es decir, cuando no sea evaluable la encuesta.

❖ Los posibles errores detectados se mostrarán de la siguiente manera:

- ERROR PARAMETROS INVALIDOS EN GENERAL

```
<response>
  <reason>Parametros invalidos</reason>
  <success>>false</success>
</response>
```

- ERROR PARAMETROS ESPECÍFICOS INVALIDOS FALTA EL TIPO

```
<response>
  <reason>Parametros invalidos: falta elemento
      tipo en el elemento 'recurso'</reason>
  <success>>false</success>
</response>
```

- ERROR PARAMETROS ESPECÍFICOS INVALIDOS FALTA EL ID RECURSO:

```
<response>
  <reason>Parametros invalidos: falta elemento
      recusoId en el elemento 'recurso'</reason>
  <success>>false</success>
</response>
```

- ERROR PARAMETROS ESPECÍFICOS INVALIDOS TIPO DE RECURSO INEXISTENTE  
(los válidos son: Link, Encuesta y Archivo):

```
<response>
  <reason>Parametros invalidos: tipo de recurso inexistente</reason>
  <success>>false</success>
</response>
```

- ERROR AL OBTENER RECURSO:

```
<response>
  <reason>Error al intentar obtener el recurso, ID: 16</reason>
  <success>>false</success>
</response>
```

- ERRORES PROVENIENTES DE LA CAPA DE INTEGRACION:

```
<response>
  <reason>Xml recibido de integracion contiene errores. Recibido: com.ws.
      handler.recursoHandlerXML</reason>
  <success>>false</success>
</response>
```

❖ *OBTENER LISTA RECURSOS***INPUT** ID-AMBIENTE**OUTPUT** LISTA-DE [ID-RECURSO, DESCRIPCION, TIPO]

```

<response>
  <success>true</success>
  <recursos>
    <recurso>
      <recursoId>11002</recursoId>
      <tipo>Link</tipo>
      <ambitoId>-1</ambitoId>
      <descripcion>un link a google copado</descripcion>
    </recurso>
    <recurso>
      <recursoId>11003</recursoId>
      <tipo>Encuesta</tipo>
      <ambitoId>-1</ambitoId>
      <descripcion>una encuesta chica</descripcion>
    </recurso>
    <recurso>
      <recursoId>11004</recursoId>
      <tipo>Encuesta</tipo>
      <ambitoId>-1</ambitoId>
      <descripcion>una encuesta grande</descripcion>
    </recurso>
  </recursos>
</response>

```

❖ *AGREGAR ARCHIVO***INPUT** ID-AMBIENTE, DESCRIPCIÓN, ARCHIVO**OUTPUT** MENSAJE DE CONFIRMACIÓN❖ *AGREGAR LINK***INPUT** ID-AMBIENTE, DESCRIPCIÓN, LINK**OUTPUT** MENSAJE DE CONFIRMACIÓN❖ *AGREGAR ENCUESTA***INPUT** ID-AMBIENTE, DESCRIPCIÓN, ENCUESTA**OUTPUT** MENSAJE DE CONFIRMACIÓN

- ❖ *GUARDAR ENCUESTA RESPONDIDA*: Es requisito tener el ID de recurso a consultar por lo que primero deben obtenerse con *OBTENER LISTA RECURSOS*.

**INPUT** ID-ENCUESTA, ID-USUARIO, LISTA-RESPUESTAS**OUTPUT** MENSAJE DE CONFIRMACIÓN

- ❖ *OBTENER ENCUESTA RESPONDIDA*: Es requisito tener el ID de recurso a consultar por lo que primero deben obtenerse con *OBTENER LISTA RECURSOS*.

**INPUT** ID-ENCUESTA, ID-USUARIO**OUTPUT** LISTA-RESPUESTAS

- ❖ *BORRAR RECURSO*: Es requisito tener el ID de recurso a consultar por lo que primero deben obtenerse con *OBTENER LISTA RECURSOS*.

**INPUT** ID-AMBIENTE, ID-RECURSO

**OUTPUT** MENSAJE DE CONFIRMACIÓN