



UNIVERSIDAD DE CÓRDOBA

ESCUELA POLITÉCNICA SUPERIOR

Ingeniería Técnica en
Informática de Gestión

PROYECTO FIN DE CARRERA

SIGAPTWIT

Sistema Informático para la Gestión de Aplicaciones
de Participación social a través de *Twitter*

Manual Técnico

Autoras: Ana María Pérez Girona
Carmen María Ramírez Ruiz

Directores: Cristina María Gámez Fernández
Juan María Palomo Romero

Córdoba, febrero de 2015

Fdo: Ana María Pérez Girona

Fdo: Carmen María Ramírez Ruiz

Córdoba, febrero de 2015

Dra. Cristina María Gámez Fernández, Profesora Contratada Doctora del Área de Conocimiento de Filología Inglesa y adscrita al Departamento de Filología Inglesa y Alemana de la Universidad de Córdoba, y **D. Juan María Palomo Romero**, Colaborador Honorario del Área de Conocimiento de Proyectos de Ingeniería (Departamento de Ingeniería Rural) de la Universidad de Córdoba,

INFORMAN

Que el presente Proyecto de Fin de Carrera, titulado ***SIGAPTWIT: Sistema Informático para la Gestión de Aplicaciones de Participación social a través de Twitter***, ha sido realizado, bajo su dirección, por las alumnas Ana María Pérez Girona y Carmen María Ramírez Ruiz, reuniendo, a su juicio, las condiciones exigidas en este tipo de trabajos.

Dra. Cristina María Gámez Fernández

D. Juan María Palomo Romero

Córdoba, febrero de 2015

ÍNDICE DE CONTENIDO

ÍNDICE DE CONTENIDO	I
ÍNDICE DE FIGURAS	VII
ÍNDICE DE TABLAS	XI
1. PRESENTACIÓN.....	1
1.1. INTRODUCCIÓN.....	1
1.2. PROBLEMA REAL	3
1.3. PROBLEMA TÉCNICO	4
1.4. FUNCIONAMIENTO	5
1.4.1. Entorno	5
1.4.2. Vida esperada	5
1.4.3. Ciclo de mantenimiento.....	5
1.4.4. Competencia	5
1.4.5. Aspecto externo.....	6
1.4.6. Estandarización	6
1.4.7. Programa de tareas	7
1.4.8. Pruebas	8
1.4.9. Seguridad	8
2. OBJETIVOS.....	9
2.1. OBJETIVO PRINCIPAL.....	9
2.2. OBJETIVOS SECUNDARIOS	9
2.3. OBJETIVOS DOCENTES	11
3. ANTECEDENTES	13
3.1. INTRODUCCIÓN.....	13
3.2. RED SOCIAL TWITTER	13
3.2.1. Definiciones	16
3.2.2. Impacto social	20
3.3. APLICACIONES PARA ESTUDIOS SOCIALES BASADAS EN TWITTER	20
3.3.1. Pollowers	21
3.3.2. Twtpoll	24

3.3.3. Sortweet	30
3.3.4. Sortwit	33
3.4. JUSTIFICACIÓN DEL PROYECTO	35
4. RESTRICCIONES.....	37
4.1. INTRODUCCIÓN.....	37
4.2. FACTORES INICIALES	37
4.3. FACTORES ESTRATÉGICOS	38
5. RECURSOS	41
5.1. INTRODUCCIÓN.....	41
5.2. RECURSOS DE HARDWARE	41
5.3. RECURSOS DE SOFTWARE	42
5.4. RECURSOS HUMANOS	43
6. ESPECIFICACIÓN DE REQUISITOS	45
6.1. INTRODUCCIÓN.....	45
6.2. REQUISITOS FUNCIONALES	46
6.2.1. Gestión de usuarios	46
6.2.2. Gestión de votaciones	46
6.2.3. Gestión de sorteos	48
6.2.4. Gestión de preguntas / respuestas libres	49
6.3. REQUISITOS NO FUNCIONALES.....	50
6.4. SUPUESTOS SEMÁNTICOS	51
7. MODELO DE DATOS	53
7.1. INTRODUCCIÓN.....	53
7.2. ANÁLISIS DE LOS TIPOS DE ENTIDAD	53
7.2.1. Tipo de entidad UsuarioAplicacion	54
7.2.2. Tipo de entidad Tweet	57
7.2.3. Tipo de entidad Actividad	59
7.2.4. Tipo de entidad Votación.....	63
7.2.5. Tipo de entidad Opcion	64
7.2.6. Tipo de entidad Sorteo	65
7.2.7. Tipo de entidad Ganador	67
7.2.8. Tipo de entidad RespuestaLibre	68
7.3. ANÁLISIS DE INTERRELACIONES	69
7.3.1. Tipo de interrelación UsuarioAplicacion-Actividad	70
7.3.2. Tipo de interrelación Votacion-Opcion.....	71
7.3.3. Tipo de interrelación Sorteo-Ganador	72
7.3.4. Tipo de interrelación Actividad-Tweet	73
7.3.5. Tipo de interrelación Tweet-Ganador	75
7.4. MODELO E-R	76

8. DESCRIPCIÓN FUNCIONAL.....	79
8.1. INTRODUCCIÓN.....	79
8.2. IDENTIFICACIÓN DE LOS ACTORES	79
8.2.1. Usuario web	80
8.2.2. Twitter	80
8.2.3. Tiempo	80
8.3. ANÁLISIS DE LOS CASOS DE USO	81
8.3.1. Caso de uso 0. Contexto del Sistema	81
8.3.2. Caso de uso 1. Gestionar Subsistema web	82
8.3.2.1. Caso de uso 1.1. GestionarUsuario	83
8.3.2.2. Caso de uso 1.2. GestionarActividad	84
8.3.3. Caso de uso 2. Gestionar Subsistema controlador	86
8.3.3.1. Caso de uso 2.1. ComprobarActividadesActivas	87
8.3.3.2. Caso de uso 2.2. ContabilizarTweets	90
8.3.3.3. Caso de uso 2.3. AlmacenarResultados	93
8.4. DIAGRAMAS DE SECUENCIA.....	95
8.4.1. Subsistema web	96
8.4.1.1. Diagrama de secuencia: GestionarUsuario	96
8.4.1.2. Diagrama de secuencia: GestionarActividad	97
8.4.2. Subsistema controlador	99
8.4.2.1. Diagrama de secuencia: ComprobarActividadesActivas	99
8.4.2.2. Diagrama de secuencia: ContabilizarTweets	101
8.4.2.3. Diagrama de secuencia: AlmacenarResultados	104
8.5. DIAGRAMAS DE ESTADO	108
8.5.1. Subsistema controlador	108
8.5.1.1. Diagrama de estado: Ciclo de vida de una actividad	109
8.5.1.2. Diagrama de estado: Contabilización de tweets	110
8.5.1.3. Diagrama de estado: Gestión de resultados	110
9. ESPECIFICACIÓN DE REQUISITOS DE LA INTERFAZ.....	113
9.1. INTRODUCCIÓN.....	113
9.2. CARACTERÍSTICAS DE LA INTERFAZ	114
10. DISEÑO DE DATOS	123
10.1. INTRODUCCIÓN.....	123
10.2. MODELO RELACIONAL.....	124
10.2.1. Tabla UsuarioAplicacion	124
10.2.2. Tabla Tweet	124
10.2.3. Tabla Actividad	125
10.2.4. Tabla Votacion	126
10.2.5. Tabla Opcion	126
10.2.6. Tabla Sorteo.....	127
10.2.7. Tabla Ganador.....	127
10.2.8. Tabla RespuestaLibre	128
10.3. NORMALIZACIÓN DEL MODELO RELACIONAL	128
10.3.1. Tabla UsuarioAplicacion	129
10.3.2. Tabla Tweet	129

10.3.3. Tabla Votacion	130
10.3.4. Tabla Opcion	130
10.3.5. Tabla Sorteo.....	130
10.3.6. Tabla Ganador.....	131
10.3.7. Tabla RespuestaLibre.....	131
10.4. ESQUEMA RELACIONAL	132
10.5. DIAGRAMA RELACIONAL.....	133
10.6. DEFINICIÓN SINTÁCTICA DE LAS TABLAS.....	134
11. DISEÑO DE PAQUETES	139
11.1. INTRODUCCIÓN.....	139
11.2. PAQUETE SIGAPTWIT_WEB	140
11.2.1. Paquete SIGAPTWIT_Web::estilos.....	142
11.2.2. Paquete SIGAPTWIT_Web::javascripts.....	143
11.2.3. Paquete SIGAPTWIT_Web::preguntas_respuesta_libre.....	143
11.2.4. Paquete SIGAPTWIT_Web::sorteos	144
11.2.5. Paquete SIGAPTWIT_Web::usuarios	144
11.2.6. Paquete SIGAPTWIT_Web::votaciones.....	144
11.3. PAQUETE SIGAPTWIT_CONTROL.....	145
12. DISEÑO DE CLASES.....	147
12.1. INTRODUCCIÓN.....	147
12.2. SUBSISTEMA CONTROLADOR	148
13. DISEÑO ARQUITECTÓNICO.....	157
13.1. DIAGRAMA DE DESPLIEGUE	157
13.2. DESCRIPCIÓN DE LOS NODOS.....	158
13.3. DESCRIPCIÓN DE LOS COMPONENTES.....	159
14. DISEÑO DE LA INTERFAZ	161
14.1. INTRODUCCIÓN.....	161
14.2. PÁGINA INICIAL	161
14.3. PÁGINA REGISTRO NUEVO USUARIO	163
14.4. PÁGINA PARA VER RESULTADOS	163
14.5. PÁGINA PARA GESTIONAR ACTIVIDADES	165
15. PRUEBAS	169
15.1. INTRODUCCIÓN.....	169
15.2. PRUEBAS REALIZADAS	169
15.2.1. Pruebas de instalación de la aplicación	170
15.2.2. Pruebas de funcionamiento de la aplicación	170
16. CONCLUSIONES	175

16.1. INTRODUCCIÓN.....	175
16.2. CONCLUSIONES SOBRE LOS OBJETIVOS PLANTEADOS.....	176
16.3. CONCLUSIONES DE LA FASE DE PRUEBAS.....	176
16.4. DOCUMENTACIÓN.....	177
 17. FUTURAS MEJORAS	179
 18. BIBLIOGRAFÍA.....	181

ÍNDICE DE FIGURAS

Figura 3.1: Página web de Twitter	14
Figura 3.2: Vista general de Twitter	16
Figura 3.3: Publicar un nuevo tweet.....	17
Figura 3.4: Retwittear	17
Figura 3.5: Página web de Pollowers	21
Figura 3.6: Pollowers. Descripción de una encuesta	22
Figura 3.7: Pollowers. Configuración de una encuesta I.....	22
Figura 3.8: Pollowers. Configuración de una encuesta II.....	22
Figura 3.9: Pollowers. Preview de la encuesta	23
Figura 3.10: Pollowers. Visualización de aviso de redundancias.....	23
Figura 3.11: Tweet generado por Twitter para dar a conocer una encuesta	23
Figura 3.12: Tweet generado por Twitter para ver los resultados de una encuesta	24
Figura 3.13: Pollowers. Resultados reales de una encuesta.....	24
Figura 3.14: Página web de Twtpoll	25
Figura 3.15: Twtpoll. Diferentes opciones de precios	25
Figura 3.16: Twtpoll. Vista de contacto	27
Figura 3.17: Twtpoll. Vista de contacto	28
Figura 3.18: Twtpoll. Crear una encuesta.....	28
Figura 3.19: Twtpoll. Resultados reales de una encuesta	29
Figura 3.20: Sortweet. Tipos de sorteos	30
Figura 3.21: Sortweet. Planes y precios.....	31
Figura 3.22: Sortweet. Sorteo mediante hashtag.....	32
Figura 3.23: Sortweet. Sorteo mediante trivia	32
Figura 3.24: Página web de Sortwit.....	33
Figura 3.25: Sortwit. Planes y precios.....	34
Figura 3.26: Sortwit. Creación de un sorteo I	35
Figura 3.27: Sortwit. Creación de un sorteo II	35
Figura 7.1: Ejemplo tipo de entidad UsuarioAplicacion	56
Figura 7.2: Ejemplo tipo de entidad Tweet	58

Figura 7.3: Ejemplo tipo de entidad Actividad	61
Figura 7.4: Diagrama jerárquico entidad Actividad	62
Figura 7.5: Ejemplo tipo de entidad Votacion	64
Figura 7.6: Ejemplo tipo de entidad Opcion.....	65
Figura 7.7: Ejemplo tipo de entidad Sorteo.....	66
Figura 7.8: Ejemplo tipo de entidad Ganador	68
Figura 7.9: Ejemplo tipo de entidad RespuestaLibre.....	69
Figura 7.10: Ejemplo tipo de interrelación UsuarioAplicacion-Actividad	71
Figura 7.11: Ejemplo tipo de interrelación Votacion-Opcion	72
Figura 7.12: Ejemplo tipo de interrelación Sorteo-Ganador	73
Figura 7.13: Ejemplo tipo de interrelación Actividad-Tweet.....	74
Figura 7.14: Ejemplo tipo de interrelación Tweet-Ganador.....	76
Figura 7.15: Modelo E-R	77
Figura 8.1: Caso de uso 0. Contexto del sistema	82
Figura 8.2: Caso de uso 1. Gestionar Subsistema web	82
Figura 8.3: Caso de uso 1.1. GestionarUsuario.....	83
Figura 8.4: Caso de uso 1.2. GestionarActividad	84
Figura 8.5: Caso de uso 2. Gestionar Subsistema controlador	87
Figura 8.6: Caso de uso 2.1. ComprobarActividadesActivas.....	87
Figura 8.7: Caso de uso 2.2. ContabilizarTweets	90
Figura 8.8: Caso de uso 2.3. AlmacenarResultados	93
Figura 8.9: Diagrama de secuencia. CrearUsuario.....	96
Figura 8.10: Diagrama de secuencia: EliminarUsuario	97
Figura 8.11: Diagrama de secuencia. CrearActividad	98
Figura 8.12: Diagrama de secuencia. EliminarActividad.....	99
Figura 8.13: Diagrama de secuencia. CompararFechaHoraInicio.....	100
Figura 8.14: Diagrama de secuencia. VerEstado	100
Figura 8.15: Diagrama de secuencia. ListarActividad	101
Figura 8.16: Diagrama de secuencia. ObtenerHashtag	102
Figura 8.17: Diagrama de secuencia. Autenticar.....	103
Figura 8.18: Diagrama de secuencia. CapturarTweet.....	104
Figura 8.19: Diagrama de secuencia. FormarNuevoTweet.....	105
Figura 8.20: Diagrama de secuencia. InsertarDatos.....	106
Figura 8.21: Diagrama de secuencia. SeleccionarGanadores.....	107
Figura 8.22: Diagrama de secuencia. InsertarGanador	108
Figura 8.23: Diagrama de estado. Ciclo de vida de una actividad	109
Figura 8.24: Diagrama de estado. Contabilización de tweets	110
Figura 8.25: Diagrama de secuencia. Almacenar resultados.....	111

Figura 9.1: Estructura de la ventana inicial	116
Figura 9.2: Estructura de la ventana de acceso público	117
Figura 9.3: Estructura de la ventana de resultados públicos.....	117
Figura 9.4: Estructura de la ventana de actividades privadas	118
Figura 9.5: Estructura de la ventana para crear nueva actividad	119
Figura 9.6: Estructura de la ventana para mostrar resultados de actividades en espera	121
Figura 9.7: Estructura de la ventana para mostrar resultados de actividades en ejecución	121
Figura 9.8: Estructura de la ventana para mostrar resultados de actividades cerradas	122
Figura 10.1: Dependencias funcionales de la tabla UsuarioAplicacion	129
Figura 10.2: Dependencias funcionales de la tabla Tweet	129
Figura 10.3: Dependencias funcionales de la tabla Votacion	130
Figura 10.4: Dependencias funcionales de la tabla Opcion.....	130
Figura 10.5: Dependencias funcionales de la Tabla Sorteo	131
Figura 10.6: Dependencias funcionales de la tabla Ganador	131
Figura 10.7: Dependencias Funcionales de la tabla RespuestaLibre	132
Figura 10.8: Diagrama Relacional	133
Figura 10.9: Leyenda del Diagrama Relacional	134
Figura 11.1: Paquete SIGAPTWIT_Web	140
Figura 11.2: Paquete estilos	142
Figura 11.3: Paquete javascripts.....	143
Figura 11.4: Paquete preguntas_respuesta_libre	143
Figura 11.5: Paquete sorteos.....	144
Figura 11.6: Paquete usuarios	144
Figura 11.7: Paquete votaciones	144
Figura 11.8: Paquete SIGAPTWIT_Control.....	145
Figura 12.1: Clase Conexion	149
Figura 12.2: Clase Actividad.....	149
Figura 12.3: Clase Votacion	150
Figura 12.4: Clase Sorteo	151
Figura 12.5: Clase Ganador.....	151
Figura 12.6: Clase RespuestaLibre	152
Figura 12.7: Clase Tweet	153
Figura 12.8: Clase HiloPadre.....	154
Figura 12.9: Clase HiloHijo.....	154
Figura 12.10: Clase Listener.....	155
Figura 13.1: Diagrama de despliegue	158
Figura 14.1: Diseño página inicial	162
Figura 14.2: Diseño página nuevo usuario	163

Figura 14.3: Diseño página ver resultados I	164
Figura 14.4: Diseño página ver resultados II	165
Figura 14.5: Diseño página ver resultados III	165
Figura 14.6: Diseño página gestionar actividades	166
Figura 14.7: Diseño página gestionar votación	167

ÍNDICE DE TABLAS

Tabla 3.1: Análisis comparativo	36
Tabla 7.1: Ejemplo práctico tipo de entidad UsuarioAplicacion.....	56
Tabla 7.2: Ejemplo práctico tipo de entidad Tweet.....	58
Tabla 7.3: Ejemplo práctico tipo de entidad Actividad.....	61
Tabla 7.4: Ejemplo práctico tipo de entidad Actividad.....	63
Tabla 7.5: Ejemplo práctico tipo de entidad Votacion	64
Tabla 7.6: Ejemplo práctico tipo de entidad Opcion	65
Tabla 7.7: Ejemplo práctico tipo de entidad Sorteo	67
Tabla 7.8: Ejemplo práctico tipo de entidad Ganador.....	68
Tabla 7.9: Ejemplo práctico tipo de entidad RespuestaLibre	69
Tabla 7.10: Ejemplo práctico tipo de interrelación UsuarioAplicacion-Actividad	71
Tabla 7.11: Ejemplo práctico tipo de interrelación Votacion-Opcion	72
Tabla 7.12: Ejemplo práctico tipo de interrelación Sorteo-Ganador.....	73
Tabla 7.13: Ejemplo práctico tipo de interrelación Actividad-Tweet	74
Tabla 7.14: Ejemplo práctico tipo de interrelación Tweet-Ganador	76
Tabla 8.1: Plantilla para definir los actores	80
Tabla 8.2: Actor Usuario web	80
Tabla 8.3: Actor Twitter.....	80
Tabla 8.4: Actor Tiempo	80
Tabla 8.6: Caso de uso 1.1.1. Crear usuario	83
Tabla 8.7: Caso de uso 1.1.2. Eliminar usuario	83
Tabla 8.8: Caso de uso 1.2.1. Crear actividad.....	84
Tabla 8.9: Caso de uso 1.2.2. Consultar actividad	85
Tabla 8.10: Caso de uso 1.2.3. Eliminar actividad	85
Tabla 8.11: Caso de uso 1.2.4. Ver resultados.....	86
Tabla 8.12: Caso de uso 2.1.1. Ver fecha hora actual.....	88
Tabla 8.13: Caso de uso 2.1.2. Comparar fecha hora inicio	88
Tabla 8.14: Caso de uso 2.1.3. Ver estado.....	88
Tabla 8.15: Caso de uso 2.1.4. Listar actividad	89
Tabla 8.16: Caso de uso 2.1.5. Lanzar actividad	89

Tabla 8.17: Caso de uso 2.2.1. Obtener hashtag	90
Tabla 8.18: Caso de uso 2.2.2. Autenticar	91
Tabla 8.19: Caso de uso 2.2.3. Capturar tweet.....	91
Tabla 8.20: Caso de uso 2.2.4. Seleccionar datos correctos.....	92
Tabla 8.21: Caso de uso 2.3.1. Formar nuevo tweet	93
Tabla 8.22: Caso de uso 2.3.2. Insertar datos.....	94
Tabla 8.23: Caso de uso 2.3.3. Seleccionar ganadores	94
Tabla 8.24: Caso de uso 2.3.4. Insertar ganadores.....	95
Tabla 12.1: Tabla de la clase Conexion	148
Tabla 12.2: Tabla de la clase Actividad	149
Tabla 12.3: Tabla de la clase Votacion.....	150
Tabla 12.4: Tabla de la clase Sorteo	150
Tabla 12.5: Tabla de la clase Ganador	151
Tabla 12.6: Tabla de la clase RespuestaLibre	152
Tabla 12.7: Tabla de la clase Tweet.....	152
Tabla 12.8: Tabla de la clase HiloPadre	153
Tabla 12.9: Tabla de la clase HiloHijo	154
Tabla 12.10: Tabla de la clase Listener	155
Tabla 13.1: Nodo: PC Usuario.....	158
Tabla 13.2: Nodo: SIGAPTWIT	158
Tabla 13.3: Componente: Navegador web.....	159
Tabla 13.4: Componente: Subsistema Controlador	159
Tabla 13.5: Componente: Subsistema Web	159
Tabla 15.1: Prueba 1.....	170
Tabla 15.2: Prueba 2.....	170
Tabla 15.3: Prueba 3.....	170
Tabla 15.4: Prueba 4.....	171
Tabla 15.5: Prueba 5.....	171
Tabla 15.6: Prueba 6.....	171
Tabla 15.7: Prueba 7	171
Tabla 15.8: Prueba 8.....	172
Tabla 15.9: Prueba 9.....	172
Tabla 15.10: Prueba 10.....	172
Tabla 15.11: Prueba 11.....	172
Tabla 15.12: Prueba 12.....	173
Tabla 15.13: Prueba 13.....	173
Tabla 15.14: Prueba 14.....	173
Tabla 15.15: Prueba 15.....	173

1. PRESENTACIÓN

1.1. INTRODUCCIÓN

En los últimos años, la introducción de las nuevas tecnologías en el ámbito doméstico, ha ido creciendo exponencialmente. En la actualidad, la mayoría de los hogares del primer mundo disponen de al menos un ordenador personal con conexión a Internet, convirtiéndose en un instrumento cotidiano en nuestras vidas. También, las entidades públicas permiten puntos de acceso gratuito a Internet, como bibliotecas, salas de aeropuertos y cafeterías, por citar algún ejemplo. Añadido a esto, la irrupción de dispositivos móviles inteligentes o *smartphones*, ha supuesto, gracias a la movilidad que ofrecen, que los usuarios utilicen Internet más frecuentemente a través de estos dispositivos que a través del propio PC.

Estos avances tecnológicos, han cambiado también la forma de vida actual, manteniendo a las personas constantemente comunicadas, con el resto del mundo, y actualizadas, en lo que a información se refiere. A parte de consultar el correo electrónico (enviar o recibir información) y acceder a datos de interés mostrados en los portales web, contemporáneamente se puede interactuar de infinitos modos en Internet. Un claro ejemplo de esto, son las redes sociales.

Una red social es una comunidad virtual donde sus usuarios, a través de Internet, interactúan con personas de todo el mundo con quienes comparten gustos o intereses en común. Funciona como una plataforma de comunicaciones que permite

conectar usuarios que se conocen o que desean conocerse, a través de la creación de un perfil y el envío-recepción de mensajes de texto e intercambiar gran cantidad de recursos multimedia, como fotos y vídeos, en un lugar fácil de acceder y administrado por los usuarios mismos.

Actualmente, está demostrado que las redes sociales desempeñan un papel de gran importancia, propiciando una gran fuente de información en tiempo real de hechos que acontecen, debido a su elevada popularidad y a la facilidad con la que los usuarios pueden enviar contenidos desde cualquier dispositivo, convirtiéndolas en algo casi imprescindible. Ejemplos de redes sociales son: *Facebook*, *Tuenti*, *LinkedIn* o *Twitter*, entre otros.

Por otro lado, cada día se hace más común y cotidiano ver un programa de televisión o escuchar una emisora de radio, y comentarlo en directo a través de una red social, con la ventaja que supone hacerlo en tiempo real, con mucha gente y desde cualquier lugar. Para los medios de comunicación, es una herramienta útil, ya que les permiten descubrir los intereses de sus espectadores a través de sus comentarios, beneficiosa información que posteriormente utilizan para mantener y aumentar las audiencias.

Cabe señalar algunas estadísticas que ponen de manifiesto la estrecha relación entre la red social *Twitter* y la televisión. *ConsumerLab*, de la firma *Ericsson*, difundió un estudio en 2012, que mostraba que un 62% de los consumidores de varios países consultados, aseguraba usar las redes sociales mientras veía algún programa televisivo (1). Por otro lado, la retransmisión de la entrega de los Oscar de la Academia 2013, generó en las redes sociales ocho millones de comentarios (2). España, aunque con indicadores más modestos, no ha estado ausente de esta tendencia. Prueba de ello, la gala final del primer programa emitido de “La Voz”, gracias a la conversación en las redes sociales, experimentó un incremento de usuarios de un 529%, al pasar de 65.000 usuarios, al principio, a 456.300 personas que sincronizaron su participación en medios interactivos con el espacio televisivo, según medición de la agencia de comunicaciones y marketing *Luparoom* (3).

No sólo los medios de comunicación y organizaciones privadas hacen uso de *Twitter* para conocer su mercado, además, los organismos públicos están haciendo de

esta herramienta su canal oficial de comunicación social en la gran mayoría de casos. Prácticamente todos los estamentos públicos participan activamente en esta red social, desde el Gobierno de España hasta las propias Universidades.

En conclusión, debido a que cada vez se lleva a cabo un uso de mayor valor de las redes sociales, se detecta la necesidad de desarrollar nuevas herramientas informáticas que, apoyándose en éstas, permitan a las distintas organizaciones obtener información útil de sus usuarios de primera mano y en tiempo real.

1.2. PROBLEMA REAL

Twitter se presenta como la mejor opción para las empresas entre las distintas redes sociales (4). Además, como se ha comentado anteriormente, es la preferida para los organismos públicos, debido al carácter, más informativo que lúdico, que tradicionalmente se le ha atribuido.

A continuación, se dividirá en tres sectores las diferentes entidades que utilizan esta herramienta, dependiendo de la finalidad principal de uso: organizaciones privadas, organismos públicos y órganos educativos.

Entre las organizaciones privadas, existe una tendencia ascendente en el uso de *Twitter* por parte de los medios de comunicación, los cuales, actualmente, se sirven de esta herramienta para conocer la opinión de sus espectadores, ya sea mediante el envío de mensajes de opinión o votaciones, ambos en tiempo real. Además, las diferentes organizaciones (por ejemplo las dedicadas a la moda, música, espectáculos...), los negocios locales y los personajes públicos (modelos, actores, cantantes, deportistas...), están viendo en *Twitter* una oportunidad para conocer aún más a sus clientes, así como para mejorar el rendimiento de sus actividades comerciales.

En el caso de los organismos públicos, *Twitter* es utilizado para ofrecer información institucional en tiempo real. Además, puede ser utilizado para conocer la opinión de los usuarios de una manera rápida y directa, mediante el uso de votaciones y sondeos.

Por último, debido al carácter universitario y educativo de este proyecto, existe un amplio abanico de aplicaciones en el entorno educativo. Dentro de la propia Universidad de Córdoba, se pueden citar órganos educativos como las Escuelas y Facultades, el *ceiA3*, las diversas Cátedras (Cooperación, Estudios sobre el Hambre y Pobreza, Participación Ciudadana, Estudios de las Mujeres “Leonor Guzmán”), la Unidad de Igualdad y el Consejo Social, entre otros. Estos podrían beneficiarse, utilizando las herramientas adecuadas, de esta red social de una manera más productiva. Además, como actividad pionera, estas aplicaciones podrían incluirse en distintas actividades docentes.

Para poder solucionar el problema detectado en los diferentes sectores mencionados, es necesario el desarrollo de una serie de herramientas que, mediante el uso de *Twitter*, aporten el valor añadido deseado:

- *Votaciones*. Conocer la opinión de los espectadores, decidir las nuevas tendencias de moda o conocer las necesidades y motivaciones de la comunidad educativa son algunos ejemplos de posibles votaciones.
- *Sorteos*. De igual forma, el cliente/usuario se siente valorado al comprobar que, su participación es premiada mediante sorteos, regalos directos y/o promociones especiales.
- *Respuesta libre*. Al contrario de las votaciones predefinidas, en este caso, la respuesta del usuario es totalmente libre, por lo que es útil para la realización de concursos, juegos, ejercicios educativos y obtención de la opinión de los usuarios.

1.3. PROBLEMA TÉCNICO

Los problemas que se plantean, y que hay que solucionar mediante la realización de este proyecto, como se ha dicho anteriormente, son:

- El primero es la necesidad de conocer la opinión de los espectadores de organizaciones privadas para así, mejorar su rendimiento.
- El segundo es la necesidad de conocer la opinión de usuarios de manera rápida y directa mediante votaciones y sondeos en organismos públicos.

1.4. FUNCIONAMIENTO

El proyecto *SIGAPTWIT* se dividirá en dos subsistemas independientes: *Subsistema web* y *Subsistema controlador*.

1.4.1. ENTORNO

Los módulos que se van a desarrollar deben integrarse en un subsistema web y en un subsistema controlador que maneje los datos, tanto de las votaciones, como de los sorteos y de las preguntas / respuestas libres. Por lo tanto, el funcionamiento de dichos módulos será multiplataforma y usará para su desarrollo los lenguajes *PHP* y *HTML*, así como el sistema *MySQL* para gestionar la base de datos. También se utilizará el lenguaje de programador *Python* (5) para todo lo relacionado con el subsistema controlador.

1.4.2. VIDA ESPERADA

Este proyecto, como la mayoría del software existente, podría cambiar constantemente, ya que estará sometido a posibles modificaciones futuras que puedan dotarlo de prestaciones más potentes.

1.4.3. CICLO DE MANTENIMIENTO

Al tratarse de un proyecto de fin de carrera, se considera que su autor no será el responsable del mantenimiento futuro de los módulos desarrollados y, por tanto, no se tendrá en cuenta este apartado. No obstante, el desarrollo se hará de forma modular para facilitar posibles futuras mejoras a otros programadores.

1.4.4. COMPETENCIA

SIGAPTWIT, por ser la primera aplicación de estas características, no presenta competencia alguna.

En el capítulo 3. “*Antecedentes*”, se detallan las aplicaciones informáticas que presentan algunas similitudes con la aplicación a desarrollar. No obstante, no se ha encontrado ninguna otra aplicación igual en el mercado, que englobe todas las funcionalidades que el proyecto *SIGAPTWIT* propone.

1.4.5. ASPECTO EXTERNO

La interfaz de la aplicación será amigable y acorde a la interfaz propia de un sistema web.

Se hará entrega de un CD-ROM con el contenido de los ficheros tanto del manual de usuario como del manual técnico.

1.4.6. ESTANDARIZACIÓN

El subsistema web va a intentar responder a los estándares actuales. La interfaz contiene una cabecera, un menú con los servicios que proporciona, un cuerpo con los servicios solicitados y un pie de página.

También, la reusabilidad de código durante la codificación y el uso de ficheros estándar van a ser tenidos en cuenta en este período.

Se utilizará UML (*Unified Modeling Language*) (6) como notación para las fases de análisis y de diseño.

El subsistema controlador se desarrollará utilizando el lenguaje de programación *Python* y haciendo uso de algunas librerías implementadas en ese mismo lenguaje como son: *JSON (JavaScript Object Notation)* (7) y *MySQLdb*, que es una interfaz compatible al popular servidor de base de datos *MySQL*, y que permite la comunicación entre la base de datos y *Python*.

Además de esto, también se ha utilizado la librería de código abierto *Tweepy version 2.3.0* (8), que permite a *Python* comunicarse con la plataforma *Twitter* y el uso de su *API* (9).

Todas estas librerías y herramientas están bastante extendidas y no representan un problema en el aspecto de la estandarización.

Al usar el lenguaje de programación Python, es recomendable emplear una serie de normas relativas al estilo de codificación. A pesar de que las reglas proporcionadas por el lenguaje son muy amplias y otorgan al programador una gran libertad, resulta habitual respetar ciertas consideraciones que facilitan la lectura y el mantenimiento de los programas desarrollados. Las principales instrucciones que se emplearán en el desarrollo de la presente aplicación son las siguientes:

- Los nombres de los paquetes deberán ser lo más breve posible.
- Los nombres de las clases e interfaces comenzará siempre por una letra mayúscula.
- Los nombres de objetos, métodos y variables miembro, así como los nombres de las variables locales de los métodos, comenzarán siempre por una letra minúscula.
- Cuando un nombre conste de varias palabras, se colocará una a continuación de la otra, sin ningún carácter separador entre ellas. La primera letra de cada palabra, excepto la primera de la cual depende el resto de palabras, se escribirá en mayúscula.
- El código fuente deberá estar correctamente documentado de forma que cualquier programador informático pueda entenderlo y utilizarlo de cara a futuras mejoras.

1.4.7. PROGRAMA DE TAREAS

Las tareas a realizar son las siguientes:

1. Recopilación de información. Recopilación de las nuevas formas de crear votaciones, sorteos y respuestas/libres.
2. Especificación de requisitos. Especificación y explicación de los distintos módulos que se van a implementar en el sistema.
3. Análisis. Consiste en el análisis del sistema, que incluye el modelo de datos, el análisis funcional y el análisis de la interfaz.

4. Diseño. Esta etapa consiste en el diseño de datos, del sistema, arquitectónico, y de la interfaz.
5. Codificación. El diseño que se ha realizado en el paso anterior se traduce a código legible para la máquina.
6. Pruebas. Se realizará un plan de pruebas para encontrar los fallos que tenga el software, con la finalidad de que quede libre de errores. Se documentará cada una de las pruebas que se realicen.
7. Documentación. Durante el desarrollo de todo el proyecto, se irá elaborando este manual técnico que describirá todas las fases de desarrollo. Además se realizarán estos otros manuales: *Manual de Código*, *Manual de Implementación* y *Manual del Usuario*.

1.4.8. PRUEBAS

Durante el diseño y la implementación del producto, éste será sometido a diferentes pruebas para asegurar su correcto funcionamiento.

En la documentación de las pruebas se indicará la siguiente información:

- Pruebas realizadas.
- Problemas encontrados.
- Soluciones adoptadas.

El capítulo de Pruebas describirá las pruebas realizadas.

1.4.9. SEGURIDAD

Toda la información de carácter sensible será encriptada y codificada en la base de datos.

Por otro lado, el proceso de identificación del usuario consistirá en el uso de un usuario y contraseña mediante procedimientos seguros, permitiendo el acceso únicamente a usuarios registrados.

2. OBJETIVOS

2.1. OBJETIVO PRINCIPAL

El objetivo principal de este proyecto (Sistema Informático para la Gestión de Aplicaciones de Participación social a través de *Twitter*) es el desarrollo de un Sistema Informático, que a través del uso de los *hashtags* de la red social *Twitter* permita la gestión de votaciones, sorteos y actividades de respuesta libre.

Se pretende alcanzar este objetivo mediante el desarrollo de dos subsistemas que compondrán el Sistema Informático:

- Subsistema web
- Subsistema controlador

2.2. OBJETIVOS SECUNDARIOS

La aplicación *SIGAPTWIT* constará de dos subsistemas, cuyos objetivos se definen a continuación.

- Subsistema para la gestión web.
 - Poder controlar la gestión de usuarios, es decir, dar de alta a un nuevo usuario y permitir la baja del sistema del mismo.

- Poder controlar la gestión de votaciones, esto es, crear votaciones, ofreciendo varias opciones, para permitir que los usuarios voten la deseada.
- Poder controlar la gestión de sorteos, es decir, crear un sorteo mediante un *hashtag* para, posteriormente, elegir al azar uno o varios usuarios participantes que resultarán los ganadores de dicho sorteo.
- Poder controlar la gestión de actividades de pregunta / respuesta libre, que serán respondidas libremente por los usuarios.
- Poder visualizar los resultados de las diferentes actividades, tanto en tiempo real como una vez finalizadas.
- Permitir la visualización de resultados tanto públicos como privados, a elección del creador de la actividad.
- Subsistema controlador:
 - Poder controlar la gestión de la participación de los usuarios.
 - Poder controlar la contabilización de los votos, es decir, almacenar únicamente aquellos *tweets* que cumplan los requisitos, descartando el resto.
 - Poder controlar la contabilización de participantes en sorteos, es decir, almacenar información relativa a los participantes evitando redundancias.
 - Poder controlar la contabilización y clasificación de las respuestas libres, mediante una selección exclusiva de las respuestas por parte de los usuarios y su posterior clasificación.
 - Poder controlar fraudes en la participación, es decir, controlar que un mismo usuario de *Twitter* no pueda participar más de una vez.

2.3. OBJETIVOS DOCENTES

La aplicación *SIGAPTWIT* pretende contribuir a la mejora de la calidad de la docencia de las asignaturas de inglés impartidas en la *Escuela Politécnica Superior*.

En particular, el principal objetivo docente que se desea alcanzar con la utilización de *SIGAPTWIT* es fomentar la participación de los profesores de las asignaturas de inglés con el alumnado a través de *Twitter*. Además, el presente proyecto es extensible a otras asignaturas de la *Universidad de Córdoba*.

En definitiva, *SIGAPTWIT* pretende ser un nuevo recurso para hacer más participativa las actividades docentes a través de las nuevas tecnologías.

3. ANTECEDENTES

3.1. INTRODUCCIÓN

Los antecedentes del presente proyecto de fin de carrera se pueden clasificar en dos categorías:

- Red social *Twitter*.
- Sistemas de gestión de votaciones, sorteos y preguntas / respuestas libres basados en *Twitter*.

3.2. RED SOCIAL *TWITTER*

Twitter es un servicio gratuito de *microblogging*, que hace las veces de red social y que permite a sus usuarios enviar micro-entradas basadas en texto, denominadas *tweets*, de una longitud máxima de 140 caracteres. Estas actualizaciones se muestran en la página de perfil del usuario, y son también enviadas de forma inmediata a otros usuarios que han elegido la opción de recibirlas. A estos usuarios se les puede restringir el envío de estos mensajes sólo a miembros de su círculo de amigos o permitir su acceso a todos los usuarios, que es la opción por defecto.

En la figura 3.1 se presenta una pantalla referente a la vista general de *Twitter*.

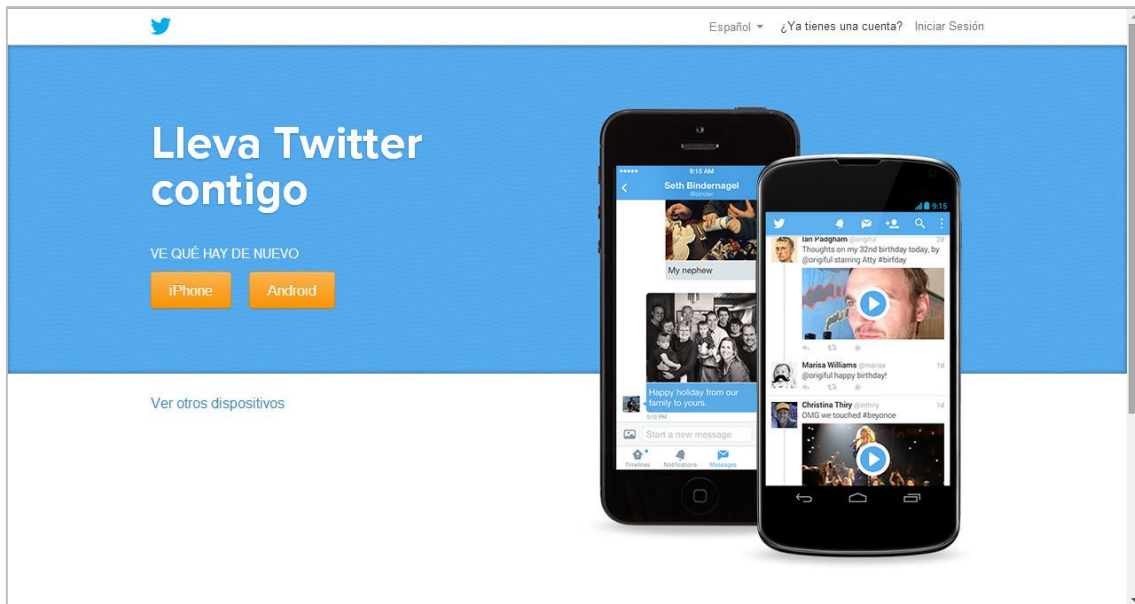


Figura 3.1: Página web de *Twitter*

Twitter nació en el año 2006, una serie de jóvenes emprendedores que trabajaban para la compañía de *PodcastsOdeo, Inc.*, de San Francisco, Estados Unidos, se vieron inmersos en un día completo de brainstorming de ideas. Jack Dorsey propuso una idea en la que se podrían usar SMS para decirle a un grupo pequeño qué estaba haciendo una persona. Fue una idea como para mantener informado a un grupo de gente sobre qué se estaba haciendo en un momento dado. Una vez iniciado el proyecto probaron varios nombres. El nombre original durante un tiempo fue "Status" (Stat.us), pasando por *twitch* (tic) a causa del tipo de vibraciones de los móviles, pero se quedaron con *Twitter*.

Algunas características importantes de la red *Twitter* son:

- Es simple. De todas las grandes redes sociales, dispone de una interface simple y sencilla de manejar.
- Es rápida. Si se desea enviar un mensaje de forma rápida y que llegue a un gran número de usuarios, la virilidad de *Twitter* es incuestionable.
- Es concisa. El hecho de no admitir más de 160 caracteres en la biografía y más de 140 en un *tweet*, obtiene mensajes claros, directos y simples, lo ideal para una buena comunicación.

- Menor coste y mayor alcance entre empresas y usuarios. El coste de gestión de *Twitter* es muy inferior al de redes como *Facebook* o *Google +*. Sobre el alcance, un *tweet* puede (o no) ser visto por todos tus seguidores, mientras que en *Facebook* la mayoría no se visualizan, debido al complejo sistema de priorizar y mostrar las historias. En *Twitter*, una cuenta de una empresa y otra de cualquier usuario, disponen de las mismas herramientas y posibilidades de actuación.
- Es la que mejor combina ocio y negocio. Si se piensa en negocio y redes sociales, *LinkedIn* es la opción más conocida, si se piensa en ocio y redes, la primera opción podría ser *Facebook*. *Twitter* aglutina por igual estas dos facetas de las redes.
- Tiene una destacada posición en los medios de comunicación. Cuando un medio de comunicación de cualquier país, temática o alcance, quiere hacer llegar a su público de qué se ha hablado en redes sociales, la casi totalidad de ellos sólo hacen referencia a los *TrendingTopic* de *Twitter* y rara vez al resto de redes.
- Se adapta a la movilidad y es accesible. El manejo de *Twitter* en aplicaciones móviles está muy depurado. Un *tweet* es la evolución de un SMS, con el que se inició la mensajería móvil. Además, cualquier usuario con discapacidad puede hacer uso de esta aplicación sin excesiva complicación.
- Tiene buena comunicación con otras redes sociales y webs. Prácticamente todos los medios sociales interactúan con *Twitter*, lo que no ocurre con el resto de redes.
- Es la que más herramientas externas tiene. De las miles de herramientas auxiliares para redes sociales que existen, una gran mayoría son herramientas exclusivas para *Twitter*. Y centrados en herramientas de medición, todas evalúan con bastante exactitud *Twitter*, mientras que las demás o se evalúan con mayor dificultad o simplemente no se evalúan.

En *Twitter*, comparado con otras redes sociales como *Facebook* o *LinkedIn*, las relaciones son, por así decirlo, asimétricas. En *Twitter* los dos extremos de la relación

no se ponen simplemente en contacto el uno con el otro, sino que se diferencia entre seguidores (“followers”) y seguidos (“followed”).

Esto se refleja en dos listas de cuentas *Twitter* diferenciadas: la lista del total de usuarios que sigue un usuario concreto, y la lista de los usuarios que sigue ese mismo usuario. Cuando un usuario sigue a alguien y ese alguien también le sigue a él, se dice además que son “co-followers”.

Si un usuario sigue a alguien, quiere decir que verá sus *tweets* en su cronología. Es decir, un usuario de *Twitter* decide a quien seguir, pero el usuario al que sigue no necesariamente tiene que seguirle a él (hacerle un “follow-back”). Esa es una diferencia importante con otras redes como *Facebook*. En la figura 3.2 se presenta una pantalla con la interfaz de *Twitter* como cliente.

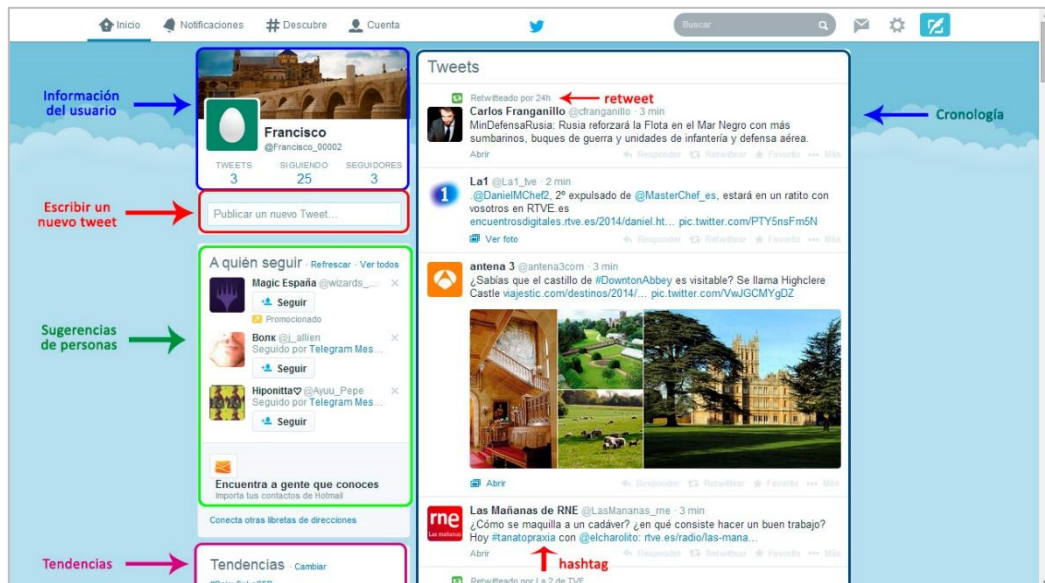


Figura 3.2: Vista general de *Twitter*

3.2.1. DEFINICIONES

- **Tweet.** Mensaje de 140 caracteres publicado vía *Twitter*. En estos caracteres también se puede escribir cualquier enlace de una página externa, como un enlace a algún video de *YouTube*, insertar una fotografía, mencionar a otro usuario o escribir un *hashtag*.
- En la figura 3.3 se presenta una pantalla de como publicar un *tweet*.



Figura 3.3: Publicar un nuevo *tweet*

- Retweet. Un *retweet* es una republicación del *tweet* de otro usuario. La característica de *retwiteo* de *Twitter* ayuda al usuario y al resto, a compartir rápidamente un *tweet* con todos tus seguidores. Los *retweets* tienen el mismo aspecto que los *tweets* normales, con el nombre del autor y el nombre del usuario al lado, pero se distinguen por el icono de *retweet* y el nombre del usuario que *retwitteó* el *tweet*.
- Retwittear. Hacer clic en *retwittear* de algún *tweet* de cualquier usuario seguido. La figura 3.4 muestra un ejemplo.



Figura 3.4: *Retwittear*

- MD. Un mensaje directo (MD) es un mensaje privado que puede enviar un usuario a través de *Twitter* a uno de sus seguidores. Un usuario solamente puede enviar un mensaje directo a otro usuario, si éste último también sigue al primero. Únicamente pueden recibir mensajes directos aquellos usuarios que se siguen recíprocamente. Los MD que se eliminan desaparecen del historial del remitente y del destinatario.
- Hashtag. Se llama *hashtag* en *Twitter* a una palabra que va precedida del símbolo #. Los *hashtags* permiten diferenciar, destacar y agrupar una palabra o tópico específico en esta red social. Con esto se consigue crear una etiqueta para aquellas palabras y así poder agruparlas y separarlas de otros

temas que incluyen el mismo término, pero que estén usándolo con un sentido diferente al que se desea otorgarle. Los *hashtags* se usan también para obtener resultados de búsqueda dentro de *Twitter*. Al hacer clic en *Twitter* sobre un *hashtag* es posible obtener como resultado *tweets* similares que incluyen el mismo término.

- TT. Los famosos “*trendingtopic* (TT)”. Un *trendingtopic* es una de las palabras o frases más repetidas en un momento concreto en *Twitter*.
- Cronología o Timeline. Conjunto ordenado en el tiempo de todos los mensajes que llegan de aquellos usuarios a los que se sigue.

Como puede observarse en la figura 3.2, la pantalla general de *Twitter* está estructurada en dos partes:

- Parte superior. En ella se encuentra una barra con varios botones y un campo de búsqueda.
 - El botón “Inicio” muestra la pantalla general (la que se puede ver en la imagen).
 - Si se hace clic en “Notificaciones” se podrá observar cuando un usuario ha mencionado a otro usuario en uno de sus *tweets*, cuando tiene un nuevo seguidor, cuando otro usuario ha *retwitteado* o ha marcado como favorito alguno de los *tweets* del usuario.
 - El botón “Descubre” hará más fácil descubrir la información de interés del usuario. Dicha sección mostrará contenido personalizado según los intereses de cada usuario.
 - El botón “Cuenta” permite ver más detalladamente la cuenta del mismo usuario, es decir, sus *tweets* (ordenados cronológicamente), sus seguidores, sus seguidos, sus favoritos o sus fotos/videos.
 - El campo de “Búsqueda” permite buscar desde cualquier usuario existente en *Twitter* hasta cualquier palabra para informar de los *tweets* en los que está contenida.

- El botón “Mensajes directos” permite al usuario ver los mensajes directos con otros usuarios, escribir uno nuevo o borrar algún mensaje.
- Desde el botón “Configuración” se puede configurar la cuenta del usuario, es decir, se puede modificar el nombre del usuario, contraseña, el diseño y los colores, las notificaciones por correo electrónico, el idioma, etc.
- El último botón permite publicar un nuevo *tweet* rápidamente. Ahí es donde se publica el nuevo *tweet* en el que, si se desea, se puede mencionar a algún usuario, insertar un determinado *hashtag*, añadir una foto, compartir algún enlace exterior o incluso añadir la ubicación actual.
- Parte central. En ella se encuentran varias partes:
 - Información del usuario: en ella aparecen datos del usuario. En la imagen, como ejemplo, el nombre del usuario sería *Francisco* y *@francisco_00002* el número de *tweets* que el usuario lleva escritos o *retwitteados*. También se muestra el número de usuarios que sigue, el número de seguidores y el campo para escribir un nuevo *tweet*.
 - Escribir un nuevo *tweet*: desde ahí también se puede publicar un nuevo *tweet*.
 - Sugerencias para seguir: muestra varias sugerencias de usuarios a quienes seguir.
 - Tendencias: ahí se muestran los diez TT más relevantes según el país del usuario. También pueden encontrarse varios *hashtags* de los que, en ese momento, se esté hablando de ellos.
 - Timeline o cronología. Es la línea en la que se ven todos los *tweets* del usuario y a los demás usuarios que sigue en orden

cronológico. Estos *tweets* también pueden ser los *tweets* retwitteados.

3.2.2. IMPACTO SOCIAL

Twitter se ha convertido en una de las redes sociales con mayor impacto social en los últimos tiempos, con cifras de usuarios que aumentan día a día de forma vertiginosa.

Es tanto este impacto, que uno de los grupos televisivos más importantes de España utiliza esta red, entre otros objetivos, para conocer mejor a los espectadores. Se analiza esa información que llega por las redes sociales para saber qué es lo que les gusta y lo que les disgusta a los espectadores.

Pero sin duda, uno de los usos que mayor impacto social ha tenido desde los inicios de esta red social, es el seguimiento en directo de algún acontecimiento inmediato. El volumen de conversaciones en torno a unas elecciones, una catástrofe o un evento internacional, es seguido con ímpetu a través de conversaciones en directo con usuarios de todo el mundo.

Todo esto, hace que el impacto social de esta red esté muy por encima de las posibilidades que ofrecen el resto de redes sociales.

3.3. APLICACIONES PARA ESTUDIOS SOCIALES BASADAS EN *TWITTER*

Las encuestas son la mejor manera de conocer la forma de pensar de un gran número de usuarios sobre un asunto en específico, y combinadas con las redes sociales pueden dar una fuente de información bastante amplia y exacta.

Para la creación de votaciones a través de *Twitter* se analizan las siguientes webs:

- *Pollowers* (10)
- *Twtpoll* (11)

Para el caso de los sorteos, se analizan las siguientes aplicaciones webs:

- *Sortweet* (12)

- *Sortwit* (13)

Para el caso de las preguntas / respuestas libres no se ha encontrado ninguna aplicación con similares características.

3.3.1. FOLLOWERS

Followers permite crear encuestas para que los seguidores de un usuario puedan responder a ellas y ver los resultados en tiempo real.

A continuación, en la figura 3.5 se muestra una imagen de la pantalla principal de su página web.



Figura 3.5: Página web de *Pollowers*

Lo primero que se debe hacer es ingresar al sitio web de *Pollowers*, y el mismo sitio define unos sencillos pasos:

1. Ingresar con la cuenta del usuario de *Twitter*.
2. Enviar una encuesta.
3. Los seguidores contestan haciendo “reply” al *tweet*.
4. Ver los resultados en tiempo real.

El primer proceso es hacer clic en “Ingresa con *Twitter*” y aparecerá el cuadro de configuración de la encuesta y la opción de envío. Esto se muestra en la figura 3.6.

Figura 3.6: *Pollsters*. Descripción de una encuesta

En la parte inferior se encuentra un panel de configuración básico como la duración de la encuesta, el *hashtag* y el idioma.

Figura 3.7: *Pollsters*. Configuración de una encuesta I

Opciones avanzadas permite configurar la publicación de resultados al inicio y final de la encuesta, y además pedir a los seguidores que hagan *Re-poll* o que ayuden a dar a conocer la encuesta.

Figura 3.8: *Pollsters*. Configuración de una encuesta II

Finalmente en la parte derecha se muestra el “preview” de la encuesta y la opción de enviarla.

22



Figura 3.9: *Pollowers*. Preview de la encuesta

Pollowers funciona íntegramente con el timeline de *Twitter*. No creará una (encuesta) con un enlace acortado donde los usuarios podrán votar, sino que se podrá votar por medio de *replies* (contestaciones), siempre de forma numérica. Esta herramienta permite que un mismo usuario pueda responder con tantos tweets, como respuestas tenga la encuesta. Pero si, por ejemplo en este caso, un usuario ha respondido “Si”, y al cabo de un tiempo vuelve a responder lo mismo, *Pollowers* no se lo permite. Se muestra la siguiente pantalla de aviso (figura 3.10).



Figura 3.10: *Pollowers*. Visualización de aviso de redundancias

Una vez realizada la encuesta y publicada en *Twitter*, *Pollowers* crea automáticamente un *tweet* en esta red social con un enlace en la cuenta del usuario para que sus seguidores lo ayuden con un *retweet* y otros usuario puedan ver la encuesta y poder votar. A continuación se muestra una imagen donde se puede ver el *tweet* generado.

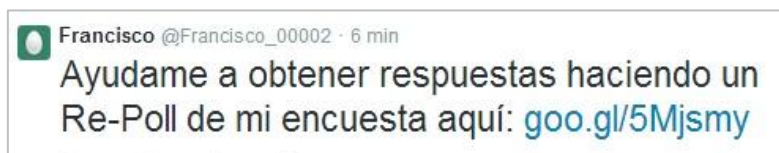


Figura 3.11: *Tweet* generado por *Twitter* para dar a conocer una encuesta

También se crea un *tweet* mediante el cual el usuario que ha creado la encuesta puede ver los resultados de dicha encuesta en tiempo real. Estos resultados

los pueden ver tanto el usuario que ha creado la encuesta como cualquier otro usuario de *Twitter* que haga clic en el enlace.

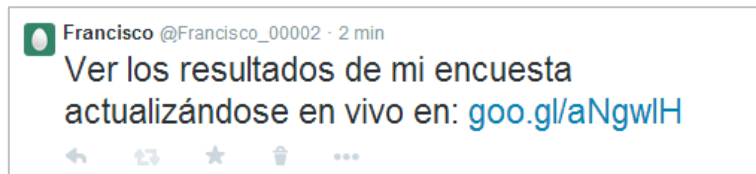


Figura 3.12: Tweet generado por *Twitter* para ver los resultados de una encuesta

Finalmente, los resultados pueden observarse a través de la página de *Pollers* (figura 3.13).

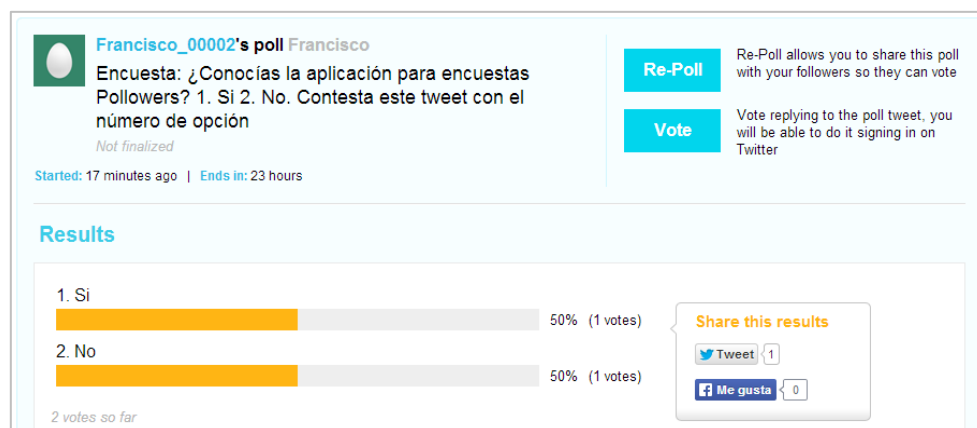


Figura 3.13: *Pollers*. Resultados reales de una encuesta

3.3.2. TWTPOLL

TwtPoll permite hacer encuestas rápidas a los seguidores de un usuario de *Twitter*. Es tan sencillo como introducir el usuario, la pregunta y las posibles respuestas. Automáticamente ofrece la posibilidad de compartirla en *Twitter* y *Facebook*, y proporciona los enlaces a la encuesta y a los resultados.

La pantalla principal de la web se muestra en la figura 3.14



Figura 3.14: Página web de Twtpoll

Esta herramienta tiene una opción gratuita y otra de pago con más opciones.

Lo primero que se debe hacer es ingresar al sitio web de Twtpoll desde la página principal. El usuario tiene varias opciones:

- Ver los precios para registrarse como usuario Premium, en el que hay más opciones a la hora de crear una encuesta.
- Contactar para cualquier duda.
- Registrarse como usuario nuevo o identificarse.
- Crear una encuesta.

A continuación se detallarán una por una cada una de las opciones anteriormente mencionadas.

Si se hace clic en el botón *Pricing*, la web muestra las opciones en la figura 3.15.

Per Question	Per Month
\$7	\$19
0 QUESTIONS (\$0)	UP TO 10 QUESTIONS PER MONTH
~3,000 VOTES	~3,000 VOTES PER MONTH
2 WEEKS	Monthly Subscription
CONTINUE	CONTINUE

Figura 3.15: Twtpoll. Diferentes opciones de precios

Las opciones de precio al subir una encuesta en *Twtpoll* son las siguientes:

- Free: esta opción es gratis, pero sólo permite 100 votos para cada encuesta.
- \$7 por encuesta: esta opción dura dos semanas y hay que pagar 7 dólares por cada encuesta realizada. A su vez, tiene varias opciones de compra:
 - Hasta 3000 votos. \$7
 - Hasta 5000 votos cuestan + \$19
 - Hasta 1000 votos cuestan + \$49
 - Hasta 2000 votos cuestan + \$79
 - Hasta 5000 votos cuestan + \$129
 - Hasta 100000 votos cuestan + \$199
 - Votos ilimitados durante dos semanas cuestan + \$399
- \$19 al mes: esta opción permite múltiples opciones, entre ellas:
 - Hasta 10 encuestas al mes con 3000 votos al mes entre todas las encuestas. \$19
 - Hasta 15 encuestas al mes. + \$29
 - Hasta 20 encuestas al mes. + \$49
 - Hasta 50 encuestas al mes. + \$99
 - Hasta 100 encuestas al mes. + \$149
 - Encuestas ilimitadas al mes. + \$299
 - Hasta 5000 votos al mes. + \$19
 - Hasta 10000 votos al mes. + \$49
 - Hasta 20000 votos al mes. + \$79
 - Hasta 50000 votos al mes. + \$129
 - Hasta 100000 votos al mes. + \$199
 - Votos ilimitados al mes. + \$399

- Esta opción también permite la opción de suscripción por meses desde 1 mes hasta 1 año.

Si se cliquea en el botón “Contact us”, la web muestra un formulario a rellenar para enviárselo a *Twtpoll*, para preguntar por cualquier duda que pueda surgir (figura 3.16).

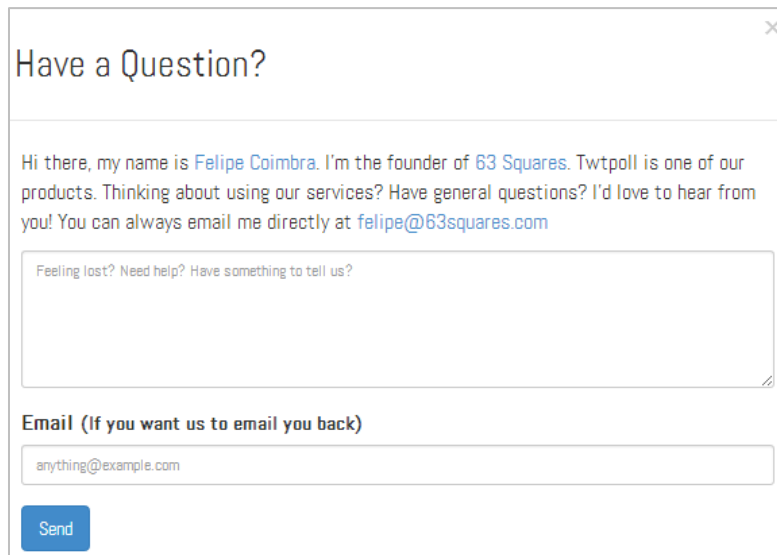
The image shows a web form titled "Have a Question?". It includes a closing button (X) in the top right corner. The form contains a paragraph of text: "Hi there, my name is Felipe Coimbra. I'm the founder of 63 Squares. Twtpoll is one of our products. Thinking about using our services? Have general questions? I'd love to hear from you! You can always email me directly at felipe@63squares.com". Below this text is a large text input area with the placeholder text "Feeling lost? Need help? Have something to tell us?". Underneath the text area is a label "Email (If you want us to email you back)" followed by an email input field containing the text "anything@example.com". At the bottom left of the form is a blue "Send" button.

Figura 3.16: *Twtpoll*. Vista de contacto

Si se hace clic en el botón “Register/Sign in” se muestra una pantalla con opciones para identificarse (figura 3.17), tales como si ya se es usuario de la web o acceder mediante la cuenta de *Facebook*, cuenta de *Google* o cuenta de *Twitter*. También permite la opción de crear una cuenta nueva.

Figura 3.17: Twtpoll. Vista de contacto

Por último, si se desea empezar a crear una encuesta, se hace clic en “create a survey”. Es preciso elegir una opción de las que se muestran en la figura 3.18.

Figura 3.18: Twtpoll. Crear una encuesta

- **#HASHTAG SURVEY:** este primer tipo, se trata de encuestas que son respondidas, mediante la red social, escribiendo un *hashtag* y a continuación la respuesta. Por ejemplo: pregunta “¿Conocías la aplicación para encuestas

twtpoll?” Opciones: “*#twtpoll* Si” o “*#twtpoll* No”. Hay que votar a través de la página web de *Twtpoll* mediante el enlace que se crea en la cuenta de *Twitter* del usuario que ha creado la encuesta.

- ONE-QUESTION SURVEY: este segundo tipo, tiene diferentes opciones de preguntas y en cada una se puede insertar, alguna foto o algún video. A continuación se muestran las diferentes opciones de preguntas con diferentes ejemplos:
 - Opción múltiple con una única respuesta válida
 - Opción múltiple con varias respuestas válidas
 - Desplegable con una única respuesta válida
 - Matriz con varias preguntas y cada una con varias respuestas
 - Escala de calificación (entre distintos valores)
 - Clasificación (elegir entre primer lugar, segundo lugar, tercer lugar,...)
 - Cuadro de texto único (para responder libremente)
 - *Twitter* comentario (para responder libremente a través de un *tweet* de la red social *Twitter*)

En todos los tipos de preguntas anteriormente mencionados pueden comprobarse sus respectivos resultados en diferentes gráficos. En la figura 3.19 se muestra un ejemplo.

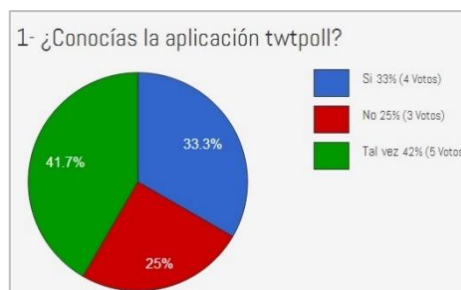


Figura 3.19: *Twtpoll*. Resultados reales de una encuesta

Esta aplicación permite mostrar los resultados tanto públicos, para que cualquier usuario los pueda ver, como privados, para que solo el usuario que ha creado la encuesta pueda verlos. También da opción a elegir para que las encuestas puedan ser votadas de diferentes formas, por ejemplo, controlando la red social desde la que se conecta el usuario o su dirección *ip*.

- MULTI-QUESTION SURVEY: este tercer tipo, a diferencia de los dos anteriores, permite crear encuestas de más de una pregunta. Tiene todas las opciones que el anterior tipo.

3.3.3. SORTWEET

Es una herramienta que permite crear y administrar sorteos profesionales en *Twitter*, garantizando que los ganadores del sorteo realizado no son seleccionados por afinidad con el organizador del sorteo, y ofreciendo a los seguidores lo que es llamado “Contenido de Valor”. De esta forma se permite la participación activa de público en *Twitter*, de interactuar con los usuarios y hacer crecer una comunidad.

Esta aplicación ofrece la posibilidad de elegir entre tres tipos de sorteo (figura 3.20), según la necesidad de cada usuario: a través de *Hashtag*, Foto o *Trivia*.



Los usuarios deberán *twittear* un *Hashtag* para participar del sorteo. Podrán hacerlo desde cualquier cliente de *Twitter* o desde la web de *Sortweet*.



Los usuarios deberán contestar una *trivia* para participar en el sorteo. El ganador será seleccionado entre los que contesten correctamente.



Los usuarios deberán *Twittear* o subir una foto para participar en el sorteo. El administrador del concurso debe aprobar las fotos antes de que aparezcan.

Figura 3.20: *Sortweet*. Tipos de sorteos

Es preciso iniciar sesión con el nombre de usuario o cuenta de correo electrónico de *Twitter*, autorizando a *Sortweet* para que pueda utilizar la cuenta de *Twitter* del perfil que propone el sorteo, es decir, autorizar a esta aplicación para leer *tweets* de la cronología, ver a quién sigue el usuario y seguir a nuevos perfiles, actualizar el perfil y publicar *tweets* en nombre del usuario que desea realizar el sorteo. Por otro lado, la aplicación no tiene capacidad para acceder a mensajes directos ni a la propia contraseña de *Twitter*.

Sortweet permite la opción de crear un sorteo de prueba de forma gratuita. No obstante, para optar a la realización de más sorteos, ofrece los planes y precios mostrados en la figura 3.21.

Plan A	Plan B	Plan C
1 Sorteo	6 Sorteos	12 Sorteos
3 tipos de sorteos disponibles	3 tipos de sorteos disponibles	3 tipos de sorteos disponibles
filtro anti-spam	filtro anti-spam	filtro anti-spam
máximo 10 días por sorteo	máximo 10 días por sorteo	máximo 10 días por sorteo
1 cuenta de twitter	6 Cuentas de twitter	12 Cuentas de twitter
EUR 20	EUR 80	EUR 120
Realizar Pago seguro	Realizar Pago seguro	Realizar Pago seguro

Figura 3.21: *Sortweet*. Planes y precios

Las figuras 3.22 y 3.23 muestran los diferentes campos que es necesario completar para la creación de un sorteo a través de *Sortweet*.

The screenshot shows the 'Sortweet' web application interface. At the top, there is a dark teal header with the 'sortweet' logo on the left and navigation links 'MI PERFIL', 'SORTEOS', 'PLANES', 'NOSOTROS', and 'SALIR' on the right. Below the header, the main section is titled 'Escoge el tipo de Sorteo'. A dropdown menu is set to 'Hashtag'. A light blue informational box states: 'Los usuarios deben Twitter el hashtag para participar. Pueden escribir con sus palabras el tweet. Los ganadores son seleccionados al azar por Sortweet.' Below this, a text prompt asks to 'Introduzca el hashtag que los usuarios deben Twitrear para participar en el sorteo, *'. There are two input fields: the first is labeled with a '#' icon and contains the text 'Hashtag'; the second is labeled with an '@' icon and contains '@Twitter'. Below the input fields, there are three checkboxes: 'Seguir a la cuenta que organiza el sorteo', 'Requerir número telefónico', and 'Requerir fecha de nacimiento'. An orange 'Siguiendo' button is located at the bottom right of the form.

Figura 3.22: Sortweet. Sorteo mediante *hashtag*

The screenshot shows the 'Sortweet' web application interface for creating a trivia contest. The header and navigation links are identical to the previous figure. The 'Escoge el tipo de Sorteo' section has a dropdown menu set to 'Trivia'. A light blue informational box states: 'Los usuarios deben responder a una pregunta de la trivia para participar. Los ganadores son seleccionados entre los que respondan correctamente. Los ganadores son seleccionados al azar por Sortweet.' Below this, there is a 'Pregunta' label followed by a large text input field. To the left of the input fields, there are four 'Respuesta' labels. The first 'Respuesta' field has a radio button selected next to the word 'Correcta'. The other three 'Respuesta' fields have radio buttons next to the word 'Incorrecta'. Below the trivia section, there is a text prompt: 'Introduzca el hashtag que los usuarios deben Twitrear para participar en el sorteo, *'. This is followed by two input fields: the first is labeled with a '#' icon and contains 'Hashtag'; the second is labeled with an '@' icon and contains '@Twitter'. At the bottom, there are three checkboxes: 'Seguir a la cuenta que organiza el sorteo', 'Requerir número telefónico', and 'Requerir fecha de nacimiento'. An orange 'Siguiendo' button is at the bottom right.

Figura 3.23: Sortweet. Sorteo mediante *trivia*

3.3.4. SORTWIT

Sortwit es un proyecto desarrollado por *AdSocia*, empresa dedicada a la generación de herramientas que ayudan a las empresas a convertir *Twitter* en un canal de captación de nuevos clientes y comunicación con los actuales. Utilizan metodologías ágiles de desarrollo junto a la filosofía de *customer development* y *lean startup*.

Esta plataforma permite crear y administrar promociones y sorteos en *Twitter* de una forma efectiva. Es necesario registrarse para crear un primer sorteo. Este registro es gratuito. Para la contabilidad de participaciones en un sorteo, se puede emplear un *hashtag*, un *RT* o una mención. En su página web, permiten la opción de que sean los administradores de *Sortwit*, los encargados de crear y guiar el sorteo que el usuario contrate, pero es preciso comunicarlo previamente a través de un formulario de contacto web.



Figura 3.24: Página web de *Sortwit*

Se trata de una aplicación de pago que funciona a través de créditos. El costo actual de un sorteo es de 10 créditos. De esta forma se puede elegir la cantidad de créditos a comprar, según las necesidades de cada cliente, y con los cuales poder realizar los sorteos y promociones.

La siguiente figura 3.25 muestra el coste por crédito de esta aplicación y la forma en que se puede realizar el pago. En la página web de *Sortwit*, se manifiesta que éstos son precios de lanzamiento promocional y que pueden variar en cualquier momento.

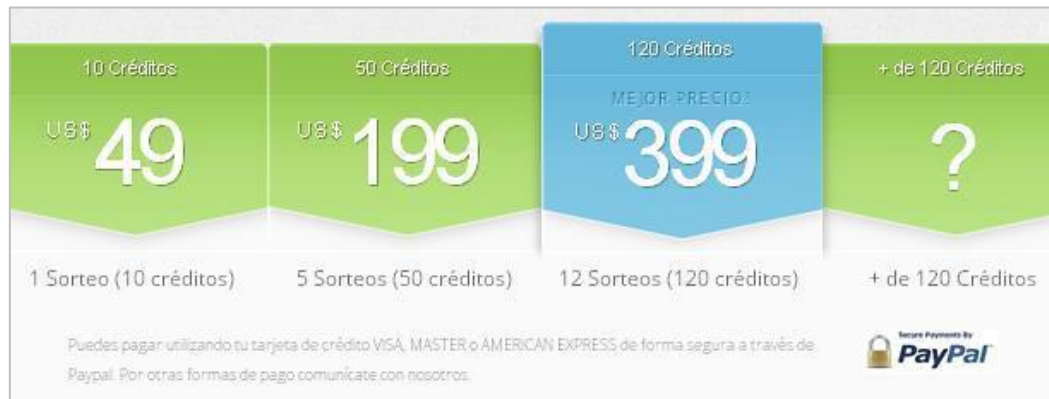


Figura 3.25: Sortwit. Planes y precios

Para comenzar un sorteo es preciso introducir el usuario o cuenta de correo electrónico, y contraseña usados en *Twitter*. Al hacer esto, se autoriza a *Sortwit* para que pueda utilizar la cuenta de *Twitter* del usuario que propone el sorteo, es decir, autorizar a esta aplicación para leer *tweets* de la cronología, ver a quién sigue el usuario y seguir a nuevos perfiles, actualizar el perfil y publicar *tweets* en nombre del usuario que desea realizar el sorteo. Por otro lado, la aplicación no tiene capacidad para acceder a mensajes directos ni a la propia contraseña de *Twitter*.

La siguiente pantalla que muestra la web, está representada en las figuras 3.26 y 3.27. En ella se solicitan todos los datos relacionados con la encuesta que se va a crear.

Bienvenido a Sortwit Carmen Ramirez Créditos: 0 Salir

Completa los datos para crear el sorteo
 A continuación debes completar el siguiente formulario para poder crear un sorteo. Todos los campos son requeridos.
 Para poder activar el sorteo debes disponer de **10 créditos**.

Sorteo: " " Nuevo: Usar un video »

Organizado por:
 Carmen Ramirez @cayma09

Validez:
 Desde: 2014-05-19 a 20 20
 Hasta: 2014-05-19 a 20 40

Datos de los participantes: Editar
 El usuario que participa via web deberá completar los siguientes datos de forma obligatoria:
☒ Nombre ☒ Email ☒ País
 Para agregar datos o hacer obligatorio el formulario de participación web haz click en "Editar".

Imagen del sorteo:
 IMAGEN DEL SORTEO AQUÍ
 (Se recomienda que el ancho de la misma sea de 600px)
 Ningún archivo seleccionado

Descripción del sorteo (opcional):

Figura 3.26: Sortwit. Creación de un sorteo I

Descripción del sorteo (opcional):

Bases y Condiciones

☒ Nombre ☒ Email ☒ País
 Para agregar datos o hacer obligatorio el formulario de participación web haz click en "Editar".

Reglas: Agregar regla
 El participante deberá cumplir al menos una de las reglas

Premios: Agregar premio

Figura 3.27: Sortwit. Creación de un sorteo II

3.4. JUSTIFICACIÓN DEL PROYECTO

Junto con los objetivos analizados en el capítulo anterior, se puede deducir la tabla 3.1 que justifica el desarrollo del presente proyecto. En ella se muestra una comparativa en la que se analizarán las siguientes soluciones: *Pollowers* y *Twtpoll*, A1 y A2 respectivamente. Además, se tendrán en cuenta los sistemas de gestión de sorteos siguientes: *Sortweet* y *Sortwit*, A3 y A4 respectivamente.

Finalmente, no se ha encontrado ningún sistema que gestione actividades del tipo respuesta libre.

Tabla 3.1: Análisis comparativo

Características	A1	A2	A3	A4	PFC
Gestión de votaciones	✓	✓	✗	✗	✓
Gestión de sorteos	✗	✗	✓	✓	✓
Gestión de respuestas libre	✗	✗	✗	✗	✓
Uso de <i>hashtags</i>	✗	✓	✓	✓	✓
Servicio gratuito	✓	✗	✗	✗	✓

4. RESTRICCIONES

4.1. INTRODUCCIÓN

En esta sección se describirán aquellos factores que limitan el proyecto. Las restricciones se contemplarán desde dos puntos de vista distintos, factores iniciales y factores estratégicos.

4.2. FACTORES INICIALES

La definición del problema y el cumplimiento de los objetivos descritos en sus correspondientes capítulos, forman las principales restricciones de este tipo que vienen impuestas por el propio proyecto, entre las que cabe destacar:

- La aplicación *SIGAPTWIT* será multiplataforma, por lo que podrá ser utilizada por cualquier usuario independientemente del sistema operativo que utilice.
- La aplicación *SIGAPTWIT*, se implementará a través de una página web que permitirá gestionar y controlar la misma en dicho entorno web.
- La aplicación se servirá de la API de *Twitter*, aprovechando los recursos que ésta ofrece.
- Los resultados obtenidos deben ser almacenados en una base de datos, siendo tratados posteriormente, para presentar resultados y estadísticas.

Una descripción más detallada del funcionamiento de *SIGAPTWIT* se puede consultar en la *Especificación de requisitos*, que corresponde al capítulo 6.

4.3. FACTORES ESTRATÉGICOS

Estos factores son identificados por las autoras del proyecto. Condicionan las diferentes propuestas alternativas en función de los recursos que se decidan utilizar, tanto a nivel de software como de hardware.

Seguidamente se exponen las restricciones que afectan al proyecto en concreto:

- Navegador web. El principal navegador web que se ha utilizado es *Google Chrome*, con su última versión actualizada, 39.0.2171.95 m, frente a otros navegadores como *Internet Explorer* o *Firefox*, ya que su interfaz es totalmente minimalista, incorpora el motor gráfico *WebKit*, considerado el más rápido y también *V8*, una versión especial de *JavaScript* que es tremendamente veloz. Aun así, el sistema está preparado para funcionar correctamente en cualquier navegador, tanto en versiones actualizadas como en versiones anteriores que cumplan con los estándares de *HTML*.
- Lenguajes de programación. Para el subsistema web se ha utilizado *HTML5*, *CSS*, *PHP* y *Javascript*. *HTML5* porque es un lenguaje predominante en la elaboración de sitios web. La versión utilizada está adaptada a las nuevas funcionalidades que se van generando dentro del desarrollo web. *CSS* porque pretende separar la estructura de un documento de su diseño. *PHP* debido a que es un lenguaje de programación diseñado para la elaboración de páginas webs dinámicas debido a su interacción con la base de datos. *Javascript* porque se integra con *HTML* y *PHP* para que la página web sea interactiva.
- Metodología de desarrollo del software. La metodología de desarrollo que se ha decidido utilizar es el *Paradigma de vida clásico*. Las razones que lo justifican es que ésta es la metodología que más se ha estudiado durante la carrera, y la que más conocen las autoras del proyecto.

- Sistema operativo del servidor. Para la elección del Sistema Operativo del servidor de pruebas se podría contar tanto con *Windows* como con *Linux*, descartando *MacOS* por el desconocimiento de este Sistema Operativo por parte de las autoras. Se ha decidido utilizar una distribución de *Linux*, ya que además de ser un sistema libre, es el predominante en los servidores webs, siendo también el más documentado en la red.
- Servidor de http. Es necesario que se disponga de un software servidor *http* con soporte *PHP*. Las opciones disponibles en un principio son *IIS (Internet Information Server)* y *Apache*. Se ha decidido utilizar *Apache*. La principal razón de la decisión se basa en que es el software de servidor *http* más utilizado en internet, además de ser gratuito.
- Sistema gestor de bases de datos. Para la elección del sistema de gestión de bases de datos se parte de dos opciones, *MySQL* y *Oracle*. Se decide utilizar *MySQL* puesto que es el sistema de gestión de bases de datos más popular en Internet.

5. RECURSOS

5.1. INTRODUCCIÓN

Este capítulo recoge información acerca de los tipos de recursos necesarios para el desarrollo del proyecto, y que son de tres tipos:

- Recursos Hardware
- Recursos Software
- Recursos Humanos

Los recursos tanto de Hardware como de Software que se utilizarán durante el proyecto se pueden dividir en las siguientes fases de la elaboración del proyecto: Elaboración de la documentación, Desarrollo e Implementación.

5.2. RECURSOS DE HARDWARE

Tanto para la *Elaboración de la documentación* como para el *Desarrollo del proyecto*, se utilizarán los ordenadores personales de cada alumna, con las características siguientes:

- Intel Core 2 Duo T6400 – 2.00 Ghz / Intel Core Dual i7 – 4510 a 2.0 Ghz
- 4 GB de RAM / 4 GB de RAM
- 250 GB de disco duro / 500 GB de disco duro

- Tarjeta Gráfica NVIDIA GeForce 9300M GS – 2029 MB de RAM / Gráfica NVIDIA GeForce GT 820M con 1GB VRAM

Igualmente, se hará uso de una impresora láser a color HP para la impresión de la documentación, así como de un dispositivo usb extraíble de 32 GB para el almacenamiento y transporte de la documentación.

5.3. RECURSOS DE SOFTWARE

El Sistema Operativo que se utilizará será *Microsoft Windows 7 Professional, Microsoft Windows XP Profesional SP 3* y *Ubuntu 12.10*, tanto para la elaboración como para el desarrollo del proyecto.

En la *Elaboración de la documentación*, se utilizará como editor de texto *Microsoft Office Word 2013*. Para la realización de los diferentes diagramas se utilizarán *Microsoft Office Visio 2007* y *DIA v0.96.1*.

Durante la etapa de *Desarrollo del proyecto*, se utilizarán las herramientas enumeradas a continuación:

- *Notepad++ v6.5.4*.
- *Eclipse Standard 4.3*.
- Plug-in para *Eclipse PyDev 1.5.1*

Para la *Implantación del proyecto* se utilizará el sistema operativo *Debian GNU/Linux 6*, por ser un sistema operativo libre, además del siguiente software:

- Servidor Web: Apache 2.
- PHP 5.
- Base de Datos: MySQL 5.
- phpMyAdmin 4.0.9.
- Python 2.7.3.
- API Oficial de *Twitter* 1.1.

5.4. RECURSOS HUMANOS

Las autoras del proyecto son las encargadas de llevar a cabo las tareas de análisis, diseño, programación y documentación, y son las alumnas de Ingeniería Técnica Informática en la especialidad de Gestión:

- Ana María Pérez Girona
- Carmen María Ramírez Ruiz

Los directores encargados de la coordinación del proyecto son:

- Dra. Cristina María Gámez Fernández: profesora Contratada Doctora del Área de Conocimiento de Filología Inglesa del Departamento de Filología Inglesa y Alemana, adscrita a la Escuela Politécnica Superior de la Universidad de Córdoba.
- D. Juan María Palomo Romero: colaborador Honorario del Área de Conocimiento de Proyectos de Ingeniería (Departamento de Ingeniería Rural) de la Universidad de Córdoba.

6. ESPECIFICACIÓN DE REQUISITOS

6.1. INTRODUCCIÓN

Los requisitos funcionales de un sistema describen lo que éste debe hacer. La captura de requisitos es un proceso extremadamente importante en el desarrollo de cualquier sistema, puesto que garantiza conocer de forma clara y sin ambigüedades, las necesidades del usuario final. Este capítulo informará acerca de los requisitos del proyecto fin de carrera *SIGAPTWIT*.

En este proyecto, el proceso de captura de requisitos se ha llevado a cabo de forma exhaustiva, utilizando para ello las siguientes técnicas:

- Introspección: las analistas se han situado en el lugar del usuario final y han definido una lista con las funcionalidades y restricciones que consideran útiles.
- Estudios de campo – documentación: las analistas han estudiado a fondo el dominio del proyecto, recopilando la mayor cantidad de información posible con el fin de entender completamente la necesidad planteada y conocer así las claves de su resolución

En los sucesivos apartados, se analizan en primer lugar, los requisitos funcionales que la aplicación resultante debe satisfacer, y pueden englobarse en cuatro apartados:

- Gestión de usuarios
- Gestión de votaciones
- Gestión de sorteos
- Gestión de respuesta libre

En segundo lugar, se enumerarán los requisitos no funcionales. Por último, se muestran los supuestos semánticos.

6.2. REQUISITOS FUNCIONALES

6.2.1. GESTIÓN DE USUARIOS

Este módulo es el encargado de gestionar los usuarios de la aplicación, es decir, de crear un nuevo usuario y de la eliminación del mismo del sistema.

Este usuario será el encargado de crear una de las posibles actividades (votación, sorteo o pregunta / respuesta libre) y además podrá eliminarla en un momento dado. También tendrá acceso a ver los resultados finales desde su propia cuenta de usuario y a decidir si los resultados de sus actividades, pueden ser públicos o únicamente visibles por él mismo.

6.2.2. GESTIÓN DE VOTACIONES

Este módulo es el encargado de permitir la creación y eliminación de una encuesta orientada a *Twitter* para poder conocer la opinión de varias personas de un determinado tema a través de esta red social. Mediante este módulo también se podrán visualizar los resultados de dichas encuestas.

Su funcionalidad consiste en crear una encuesta en el subsistema web y permitir que respondan a la misma los usuarios de *Twitter* entre varias opciones.

Se contabilizarán todas las respuestas válidas entre los usuarios participantes en la encuesta, desde la hora y fecha en la que la misma quede activa hasta la hora y fecha en la que finalice. Posteriormente, mostrará los resultados obtenidos en forma de porcentajes o bien de forma gráfica, de las encuestas que estén aún abiertas y de las que hayan finalizado. Dichos resultados pueden ser mostrados en tiempo real, sin necesidad de que una votación tenga que estar cerrada.

También existirá un acceso público para ver los resultados de las encuestas. Desde este enlace, cualquier usuario de la red puede comprobar los resultados de las encuestas publicadas por el usuario del subsistema web.

Las respuestas permitidas, como mínimo, serán dos, y a partir de ahí, el número de respuestas variará según estime oportuno el propio usuario. Éstas se responderán *twitteando* un *hashtag*, es decir, escribiendo en la cuenta de *Twitter* el símbolo # seguido por una de las opciones planteadas.

Un ejemplo ilustrativo, puede ser el siguiente:

Pregunta: “¿Qué titulaciones universitarias consideras que tienen más salidas profesionales?”

Respuestas posibles:

A) #titulacionADE

B) #titulacionINGENIERO

C) #titulacionDERECHO

D) #titulacionMEDICINA

La forma de responder a esta encuesta a través de *Twitter*, sería escribiendo un *tweet* en una cuenta de *Twitter*, en el que aparezca la respuesta (cadena de caracteres formada por una o varias palabras concatenadas) precedida por el carácter #, es decir, *twittear* un *hashtag* de los posibles aceptados como respuesta y que computen como válidos.

Si algún usuario *twittear* otra respuesta diferente a las respuestas posibles, por ejemplo, #TitulacionDerecho o #titulacionFARMACIA, dicha respuesta no será contabilizada y quedará fuera de las votaciones.

6.2.3. GESTIÓN DE SORTEOS

Este módulo es el responsable de permitir la creación de sorteos a través de *Twitter*.

Su funcionalidad consiste en crear un sorteo a través del subsistema web para premiar a los participantes en dicho sorteo. La forma de participar será *twitteando* el *hashtag* que indique el creador del sorteo (en el campo del subsistema web indicado para ello) de forma correcta y válida para que compute como respuesta participante. El ganador del sorteo se elegirá de forma aleatoria entre todos los que *twitteen* adecuadamente el *hashtag* sugerido una vez finalizado el sorteo, o también, puede existir más de un ganador, en este caso serían los n primeros en *twittear* de forma correcta, siendo n indicado por el usuario en el proceso de creación de un nuevo sorteo.

La aplicación *SIGAPTWIT* contabilizará todos los *tweets* correctos, realizará la elección del ganador/ganadores y mostrará los resultados. También será posible visualizar el número de usuarios que están participando en el sorteo antes de la finalización del mismo.

El sorteo se realizará únicamente teniendo en cuenta a los usuarios que respondan correctamente a la cuestión planteada.

Un ejemplo ilustrativo, puede ser el siguiente:

Sorteo: *“¿Quieres realizar el Curso de Programación de Google Glass y Google Watch que imparte la UCO? Si resultas ganador, el importe de la matrícula será gratuito.”*

Participa ahora con el hashtag #cursogratisUCO

Es preciso indicar el *hashtag* # con el cual responder a la pregunta planteada en el sorteo. Esto es algo imprescindible para que los usuarios sepan cómo participar en el mismo.

La forma de participar en este sorteo a través de *Twitter* sería la siguiente: el usuario debe escribir un *tweet* en su cuenta personal de *Twitter*, en el que aparezca la

respuesta (cadena de caracteres formada por una o varias palabras concatenadas) precedida por el carácter #, es decir, *twittear* el *hashtag* indicado.

Para participar en el sorteo anterior planteado → *twittear* *#cursogratisUCO* desde nuestra cuenta de *Twitter*.

La funcionalidad de este apartado, contabilizará todas las respuestas válidas de entre los usuarios participantes en el sorteo, y posteriormente se elegirá al ganador o ganadores, según se haya indicado en la creación del sorteo, y se mostrarán una vez finalizado el sorteo.

El sistema controla que un mismo usuario no pueda participar dos veces en el mismo sorteo.

Sólo se permite un *hashtag* como respuesta válida.

6.2.4. GESTIÓN DE PREGUNTAS / RESPUESTAS LIBRES

Este módulo es el encargado de gestionar la respuesta libre en *Twitter*. La respuesta libre va a consistir en un *hashtag* #, que será identificado de forma unívoca y a través del cual poder contabilizar las respuestas de los usuarios.

A través del subsistema web se planteará la cuestión en la que participar. Un ejemplo ilustrativo que refleja la respuesta libre, puede ser el siguiente:

Pregunta: “¿En qué ciudad te gustaría desarrollar tu carrera profesional?”

Para contestar a esta pregunta a través de *Twitter*, el usuario debe añadir su respuesta precedida por un *hashtag*. En este caso se identifica el *hashtag* como *#TrabajarEn* y a continuación la respuesta con la que desee aportar su opinión. Ejemplos válidos serían:

“*#TrabajarEn Córdoba*”

“*#TrabajarEn Barcelona o Madrid*”

“*#TrabajarEn fuera de mi país*”

Como se puede comprobar en los ejemplos anteriores, la repuesta libre va a ser todo lo que se escriba a partir del *hashtag* # que se indique para contestar a la

cuestión planteada. Si el usuario *twitteo* más de un *hashtag* en el mismo tweet, la respuesta considerada válida será todo lo que aparezca a continuación del *hashtag* # propuesto.

Se almacenará información acerca del usuario, la hora y fecha en la cual ha enviado el *tweet* con el *hashtag*, y la localización del mismo, suponiendo que el usuario tenga activa esta opción en *Twitter*.

6.3. REQUISITOS NO FUNCIONALES

Los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino que permiten especificar características de arquitectura lógica, infraestructura, interfaz gráfica, seguridad, rendimiento, disponibilidad, integración y portabilidad. Los requisitos no funcionales son detallados a continuación:

- Requisito 1: la aplicación debe ser intuitiva y eficaz, de forma que se puedan seleccionar parámetros y escoger opciones rápidamente y de forma robusta, informando al usuario de lo que ocurre en cada momento.
- Requisito 2: el sistema debe responder ante posibles errores ya sean por parte del usuario o del propio sistema.
- Requisito 3: redundancia mínima. Deben existir las mínimas copias posibles de un mismo dato, puesto que esto disminuye el espacio de almacenamiento y además dificulta el mantenimiento de los datos.
- Requisito 4: el tiempo de respuesta hombre-máquina debe ser mínimo para una mayor agilización de los datos.
- Requisito 5: todos los mensajes de error del sistema deben ser significativos, de manera que se incluya un texto descriptivo y una indicación de las acciones que el usuario debe efectuar para subsanar dicho error.
- Requisito 6: el sistema siempre deberá estar disponible.

6.4. SUPUESTOS SEMÁNTICOS

1. Cada usuario tiene un único identificador unívoco automático. Además de tener un nombre de usuario único en el sistema y una contraseña.
2. Cada usuario está asociado a una cuenta de correo electrónico.
3. Un usuario puede crear varias votaciones o incluso ninguna.
4. Cada votación tiene un identificador unívoco automático. Además de tener un nombre, que puede ser duplicado en la base de datos, una descripción, una fecha y hora de inicio y otra de finalización, las opciones que se tomarán como respuesta y un estado dentro del sistema.
5. Las opciones de respuesta en una votación siempre son mayor que uno. El usuario decidirá el número de opciones como respuestas hasta un máximo de diez.
6. Una votación puede tener uno de los siguientes tres estados: en espera, activa o finalizada.
7. Una votación puede ser respondida por ninguna persona o incluso por varias.
8. Un mismo usuario no puede participar más de una vez en la misma votación.
9. Toda votación crea un gráfico mostrando la cantidad de votos que recibe cada una de las opciones.
10. Un usuario puede crear varios sorteos o incluso ninguno.
11. Cada sorteo tiene un identificador unívoco automático. Además de tener un nombre, que puede ser duplicado en la base de datos, una descripción, una fecha y hora de inicio y otra de finalización, el número de premiados y el tipo de elección de ganadores, es decir, si se escogerán aleatoriamente o serán los n primeros en contestar, siendo n el número de ganadores indicado por el usuario en la creación del sorteo. Incluirá un *hashtag* único, que será el válido para poder participar en el sorteo, y un estado dentro del sistema.
12. Un sorteo tiene una respuesta única correcta.

13. Un sorteo puede tener uno de estos tres estados: en espera, activo o finalizado.
14. En cada sorteo pueden participar desde ninguna persona hasta varias personas.
15. Un mismo usuario no puede participar más de una vez en el mismo sorteo.
16. Todo sorteo muestra como resultado el/los ganador/es del mismo.
17. Un usuario puede crear varias respuestas libres o incluso ninguna.
18. Cada pregunta / respuesta libre tendrá un identificador unívoco automático. Además de tener un nombre, que puede ser duplicado en la base de datos, una descripción, una fecha y hora de inicio y otra de finalización, un campo para guardar la respuesta libre, el *hashtag* válido para poder participar en la respuesta libre, y un estado dentro del sistema.
19. Una pregunta / respuesta libre puede ser respondida por ningún usuario. O también puede ser respondida por varios usuarios diferentes.
20. Un mismo usuario no puede participar más de una vez en la misma pregunta / respuesta libre.
21. Toda pregunta / respuesta libre muestra los resultados de la pregunta / respuesta libre planteada.
22. Cualquier usuario de *Twitter* podrá participar en cualquier votación, sorteo y/o pregunta / respuesta libre.
23. Cualquier usuario registrado en el sistema podrá ver los resultados gráficos de sus actividades votación, sorteo y/o pregunta / respuesta libre.
24. Cualquier usuario no registrado en el sistema podrá ver los resultados públicos, de forma gráfica, de cualquier votación, sorteo y/o pregunta / respuesta libre.
25. Todo *hashtag* de cualquier actividad debe comenzar por # seguida del texto sin contener caracteres en blanco.

7. MODELO DE DATOS

7.1. INTRODUCCIÓN

En este capítulo se hará un análisis de la estructura de la base de datos, que será la encargada del procesamiento masivo de datos. Este análisis se centrará en la representación del conjunto de datos y la relación entre los mismos.

El modelo de datos que se utilizará para este análisis será el *Modelo Entidad/Relación* propuesto por *Peter Chen* (14).

7.2. ANÁLISIS DE LOS TIPOS DE ENTIDAD

A continuación, se muestra una descripción inicial de los tipos de entidad que pueden extraerse de la definición del problema real, y que servirán para identificar el correcto mantenimiento de la información en el sistema. La entidad presenta un objeto del mundo real con existencia independiente y que se identifica a partir de uno o varios atributos, que son las propiedades que describen a cada entidad.

Las entidades que se van a analizar corresponden al entorno en el cual se desarrolla el proyecto. Para este análisis se mostrarán, para cada tipo de entidad, los siguientes apartados:

- Descripción: indicará el papel representado por el tipo de entidad en el mundo real, así como las posibles dependencias existentes con otros tipos de entidad.
- Características: se mostrarán los siguientes datos sobre el tipo de entidad:
 - Nombre del tipo de entidad.
 - Atributo/s identificador/es principal/es.
 - Atributo/s identificador/es alternativo/s.
 - Atributo/s heredado/s.
 - Número de atributos.
- Atributos propios: se hará una descripción de cada uno de los atributos que forman el tipo de entidad, indicando su definición, dominio, así como el grado de obligatoriedad.
- Diagrama: representación gráfica del tipo de entidad.
- Ejemplo: se indicará un ejemplo de un posible caso real de ocurrencia del tipo de entidad.

Los atributos que se analizarán serán los más relevantes para el estudio del modelo de datos, obviando atributos secundarios que no influirán en el desarrollo de este proyecto.

7.2.1. TIPO DE ENTIDAD *USUARIO* *APLICACION*

- Descripción: cada uno de los usuarios que se registran en la plataforma web, y que tienen acceso para crear sorteos, votaciones o actividades de pregunta / respuesta libre.
- Características:
 - Nombre de la entidad: UsuarioAplicacion
 - Atributo/s identificador/es principal/es: id_usuario.
 - Atributo/s identificador/es alternativo/s: -.

- Atributo/s heredado/s: -.
- Número de atributos: 7.
- Atributos propios:
 - Id_usuario
 - Definición del atributo: es el código que identifica a la categoría unívocamente.
 - Definición del dominio: números naturales. Autoincremento.
 - Carácter: obligatorio.
 - Nombre
 - Definición del atributo: nombre de la categoría.
 - Definición del dominio: texto.
 - Carácter: obligatorio.
 - Apellidos
 - Definición del atributo: se utiliza como complemento del atributo nombre, para definir a cada uno de los usuarios de la aplicación.
 - Definición del dominio: texto.
 - Carácter: obligatorio.
 - Codigo_postal
 - Definición del atributo: código numérico necesario para ubicar a cada usuario de nuestra aplicación en una población o en las distintas zonas dentro de ella.
 - Definición del dominio: números naturales.
 - Carácter: obligatorio.
 - Usuario
 - Definición del atributo: nombre unívoco con el cual es identificado un usuario dentro de la aplicación.
 - Definición del dominio: caracteres alfanuméricos.
 - Carácter: obligatorio.
 - Contraseña

- Definición del atributo: se corresponde con un conjunto de caracteres, compuesto por números y letras, necesario para poder acceder a la aplicación.
- Definición del dominio: caracteres alfanuméricos.
- Carácter: obligatorio.
- E-mail
 - Definición del atributo: corresponde con una dirección de correo electrónico asociado a cada usuario de nuestra aplicación, mediante el cual mantener algún tipo de contacto en caso necesario.
 - Definición del dominio: caracteres alfanuméricos.
 - Carácter: obligatorio.
- Diagrama:

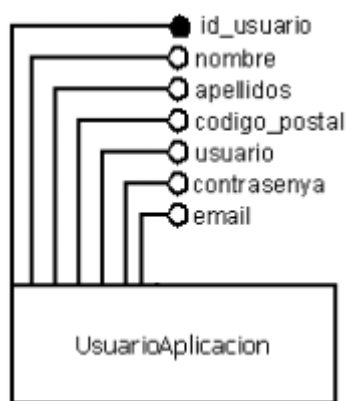


Figura 7.1: Ejemplo tipo de entidad *UsuarioAplicacion*

- Ejemplo:

Tabla 7.1: Ejemplo práctico tipo de entidad *UsuarioAplicacion*

UsuarioAplicacion	
Id_usuario	1
Nombre	Antonio
Apellidos	Pérez Campillo
Codigo_Postal	14920
Usuario	i14capei
Contrasenya	micontrasenya009
Email	i14capei@uco.es

7.2.2. TIPO DE ENTIDAD *TWEET*

- Descripción: se corresponde con cada uno de los mensajes que se envían a través de *Twitter* para poder participar en cada una de las actividades (sorteo, votación o respuesta libre) de la plataforma. Es un tipo de entidad débil por identificación respecto a la entidad *Actividad*, ya que no puede ser identificada por sí sola, es necesario identificarla con dicha entidad para poder identificarla.
- Características:
 - Nombre de la entidad: Tweet.
 - Atributo/s identificador/es principal/es: id_tweet.
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 5.
- Atributos propios:
 - Id_tweet
 - Definición del atributo: es el código que identifica al tweet unívocamente.
 - Definición del dominio: números naturales.
 - Carácter: obligatorio.
 - Id_usuario
 - Definición del atributo: es el nombre de usuario de la cuenta emisora del *tweet*.
 - Definición del dominio: carácter '@' seguido de una cadena de caracteres.
 - Carácter: obligatorio.
 - Contenido
 - Definición del atributo: este atributo contiene el mensaje del tweet.

- Definición del dominio: cadena de caracteres precedida por el signo #.
- Carácter: obligatorio.
- Fecha_hora_entrada
 - Definición del atributo: corresponde con la fecha y hora en la cual se publica un tweet.
 - Definición del dominio: es un dato de tiempo “date” o fecha. El formato usado es “YYYY-MM-DD HH:MM:SS”.
 - Carácter: obligatorio.
- Geolocalizacion
 - Definición del atributo: muestra la ubicación geográfica desde donde se ha publicado el tweet.
 - Definición del dominio: cadena de caracteres presentando el formato “Población, Provincia”.
 - Carácter: opcional.
- Diagrama:



Figura 7.2: Ejemplo tipo de entidad *Tweet*

- Ejemplo:

Tabla 7.2: Ejemplo práctico tipo de entidad *Tweet*

Tweet	
Id_tweet	1236567891612
Id_usuario	@antonio_zep
Contenido	#cuestionarioIngles
Fecha_hora_entrada	2014-10-22 17:50:00
Geolocalizacion	Villafranca de Córdoba, Córdoba

7.2.3. TIPO DE ENTIDAD *ACTIVIDAD*

- Descripción: cada una de las posibles acciones que pueden plantear los usuarios de la aplicación, y con las cuales, interactuar a través de *Twitter*. Esta entidad es débil por existencia respecto a la entidad *UsuarioAplicacion*, ya que es necesario que exista previamente un usuario para poder crear una actividad.
- Características:
 - Nombre de la entidad: Actividad.
 - Atributo/s identificador/es principal/es: id_actividad.
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 8.
- Atributos propios:
 - Id_actividad
 - Definición del atributo: es el código que identifica a la actividad unívocamente.
 - Definición del dominio: números naturales.
 - Carácter: obligatorio.
 - Titulo
 - Definición del atributo: nombre con el cual identificar una actividad.
 - Definición del dominio: caracteres alfanuméricos.
 - Carácter: obligatorio.
 - Descripcion
 - Definición del atributo: este atributo se utiliza para describir de forma más detallada en qué consiste la actividad.
 - Definición del dominio: caracteres alfanuméricos.
 - Carácter: obligatorio.

- Fecha_inicio
 - Definición del atributo: se corresponde con la hora y fecha exactas a partir de las cuales está activa una actividad.
 - Definición del dominio: es un dato de tiempo “date” o fecha. El formato usado es “YYYY-MM-DD HH:MM:SS”.
 - Carácter: obligatorio.
- Fecha_fin
 - Definición del atributo: se corresponde con la hora y fecha exactas a partir de las cuales una actividad deja de estar activa.
 - Definición del dominio: es un dato de tiempo “date” o fecha. El formato usado es “YYYY-MM-DD HH:MM:SS”.
 - Carácter: obligatorio.
- Cuestion
 - Definición del atributo: es utilizado para guardar el enunciado exacto de una actividad, y a través del cual los usuarios de *Twitter* pueden responder de forma correcta.
 - Definición del dominio: texto.
 - Carácter: obligatorio.
- Estado
 - Definición del atributo: característica de una actividad según si está en espera, activa o finalizada.
 - Definición del dominio: uno de los valores “2”, “1” o “0”. El valor “2” indica que la actividad está en espera de ejecutarse; el valor “1”, indica que la actividad está en ejecución; y el valor “0” indica que el estado de la actividad es finalizado.
 - Carácter: obligatorio.
- Publico
 - Definición del atributo: característica de una actividad que indica si los resultados son mostrados al público general, o únicamente son visibles por el creador de la actividad.
 - Definición del dominio: uno de los valores “0” o “1”. El valor “0” indica que la actividad únicamente puede ser visualizada

por el creador de la misma; el valor “1”, indica que la actividad también es visible para los usuarios no registrados en la aplicación.

- Carácter: obligatorio.

- Diagrama:

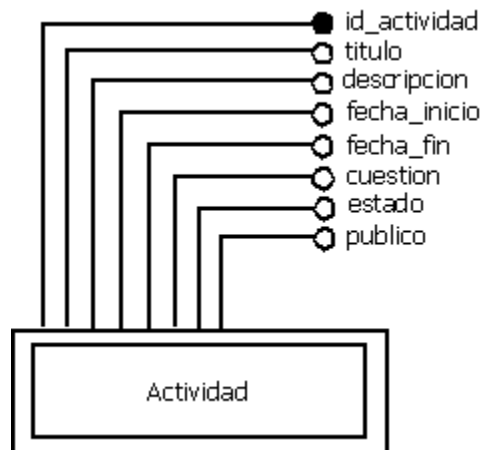


Figura 7.3: Ejemplo tipo de entidad *Actividad*

- Ejemplo:

Tabla 7.3: Ejemplo práctico tipo de entidad *Actividad*

Actividad	
Id_actividad	107
Titulo	Concurso para conseguir una plaza gratuita en el curso “AutoCAD para técnicos”.
Descripcion	Concurso destinado a promover la participación del alumnado en cursos impartidos por la UCO.
Fecha_inicio	2014-10-23 00:00:00
Fecha_fin	2014-10-26 00:00:00
Cuestion	¿Deseas realizar el curso “AutoCAD para técnicos” que imparte la UCO con matrícula gratuita? Participa con el hashtag #sorteoAutoCAD
Estado	2
Publico	1

- Generalización: la entidad *Actividad* es un supertipo de entidad que mantiene una relación jerárquica con los siguientes subtipos: *Votacion*, *Sorteo* y *RespuestaLibre*.
- Especialización: es una especialización exclusiva (especialización sin solapamiento) y total, es decir, no existirá una entidad *Actividad* que no sea de alguno de los tres subtipos planteados y además, solo pertenecerá a uno y sólo a uno de dichos subtipos.
- Atributo Tipo-actividad: este atributo tiene la función de clasificador de las entidades supertipo (*Actividad*) en alguno de los subtipos. Es decir, *tipo_actividad* puede tomar los siguientes valores: *Votacion*, *Sorteo* o *RespuestaLibre*.
- Diagrama jerárquico:

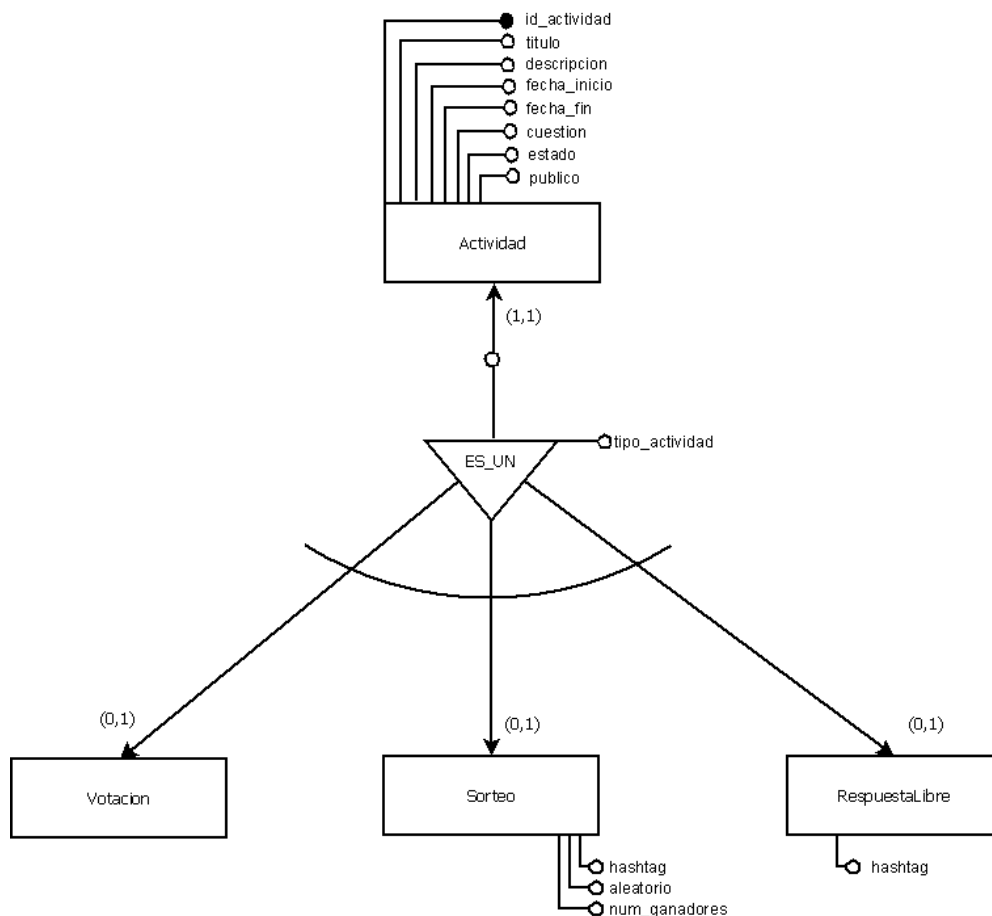


Figura 7.4: Diagrama jerárquico entidad *Actividad*

- Ejemplo:

Tabla 7.4: Ejemplo práctico tipo de entidad *Actividad*

Actividad	
Id_actividad	107
Titulo	Concurso para conseguir una plaza gratuita en el curso “AutoCAD para técnicos”.
Descripcion	Concurso destinado a promover la participación del alumnado en cursos impartidos por la UCO.
Fecha_inicio	2014-10-23 00:00:00
Fecha_fin	2014-10-26 00:00:00
Cuestion	¿Deseas realizar el curso “AutoCAD para técnicos” que imparte la UCO con matrícula gratuita? Participa con el hashtag #sorteoAutoCAD
Estado	2
Publico	1
Tipo	Sorteo
Hashtag	#sorteoAutoCAD
Aleatorio	1
Num_ganadores	2

7.2.4. TIPO DE ENTIDAD *VOTACIÓN*

- Descripción: esta entidad corresponde a un tipo de actividad específica de la aplicación, llamada *Votacion*. Es un subtipo que forma parte de la especialización del tipo de entidad *Actividad*.
- Características:
 - Nombre de la entidad: *Votacion*.
 - Atributo/s identificador/es principal/es: id_actividad
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 1.
- Atributos propios:
 - Id_actividad
 - Definición del atributo: es el código que identifica a la actividad unívocamente.

- Definición del dominio: números naturales.
- Carácter: obligatorio.
- Diagrama:

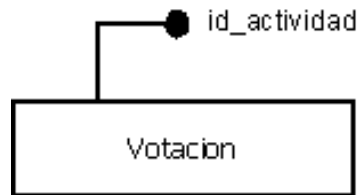


Figura 7.5: Ejemplo tipo de entidad *Votacion*

- Ejemplo:

Tabla 7.5: Ejemplo práctico tipo de entidad *Votacion*

Votacion	
Id_actividad	2010

7.2.5. TIPO DE ENTIDAD *OPCION*

- Descripción: esta entidad corresponde a cada una de las opciones posibles que se plantean para participar en una votación. Esta entidad es débil por existencia respecto a la entidad *Votacion*, ya que es necesario que exista una actividad tipo *votacion* para poder crear las opciones con las que participar.
- Características:
 - Nombre de la entidad: *Opcion*.
 - Atributo/s identificador/es principal/es: *id_actividad*.
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 2.
- Atributos propios:
 - *Id_opcion*
 - Definición del atributo: es el código que identifica a la opción unívocamente.

- Definición del dominio: números naturales. Autoincremento.
- Carácter: obligatorio.
- Hashtag
 - Definición del atributo: define cada una de las frases con las que se puede participar en una votación planteada.
 - Definición del dominio: caracteres alfanuméricos.
 - Carácter: obligatorio.
- Diagrama

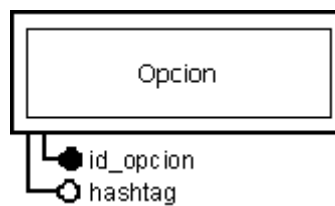


Figura 7.6: Ejemplo tipo de entidad *Opcion*

- Ejemplo:

Tabla 7.6: Ejemplo práctico tipo de entidad *Opcion*

Opcion	
Id_opcion	2000
Hashtag	#titulacionMEDICINA

7.2.6. TIPO DE ENTIDAD *SORTEO*

- Descripción: esta entidad corresponde a un tipo de actividad específica de la aplicación. Es un subtipo que forma parte de la especialización del tipo de entidad *Actividad*.
- Características:
 - Nombre de la entidad: Sorteo.
 - Atributo/s identificador/es principal/es: id_actividad.
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 4.

- Atributos propios:
 - Id_actividad
 - Definición del atributo: es el código que identifica a la actividad unívocamente.
 - Definición del dominio: números naturales.
 - Carácter: obligatorio.
 - Hashtag
 - Definición del atributo: define el hashtag para poder participar en un sorteo unívocamente.
 - Definición del dominio: caracteres alfanuméricos.
 - Carácter: obligatorio.
 - Aleatorio
 - Definición del atributo: identifica si el ganador o ganadores del sorteo, serán elegidos de forma aleatoria entre todos los participantes, o serán los n primeros en participar correctamente.
 - Definición del dominio: uno de los valores "0", "1".
 - Carácter: obligatorio.
 - Num_ganadores
 - Definición del atributo: identifica el número de ganadores posibles que tendrá como resultado el sorteo.
 - Definición del dominio: números naturales.
 - Carácter: obligatorio.
- Diagrama:

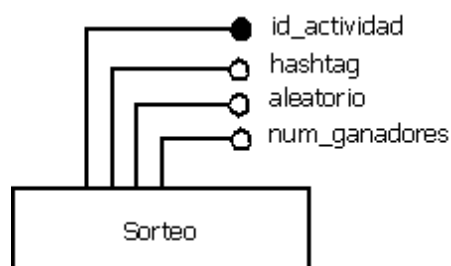


Figura 7.7: Ejemplo tipo de entidad *Sorteo*

- Ejemplo:

Tabla 7.7: Ejemplo práctico tipo de entidad *Sorteo*

Sorteo	
Id_actividad	23
Hashtag	#cursogratisUCO
Aleatorio	1
Num_ganadores	5

7.2.7. TIPO DE ENTIDAD *GANADOR*

- Descripción: esta entidad corresponde al usuario que resulta ganador de una actividad tipo sorteo, y que puede ser elegido o bien de forma aleatoria entre los que twitteen el hashtag correctamente, o bien de forma que ganan los n primeros en twittear el hashtag indicado de forma correcta.
- Características:
 - Nombre de la entidad: Ganador
 - Atributo/s identificador/es principal/es: id_ganador
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 1.
- Atributos propios:
 - Id_ganador
 - Definición del atributo: es el código que identifica a un ganador del tipo de entidad "Sorteo".
 - Definición del dominio: números naturales. Autoincremento.
 - Carácter: obligatorio.
- Diagrama:



Figura 7.8: Ejemplo tipo de entidad *Ganador*

- Ejemplo:

Tabla 7.8: Ejemplo práctico tipo de entidad *Ganador*

Ganador	
Id_ganador	12

7.2.8. TIPO DE ENTIDAD *RESPUESTALIBRE*

- Descripción: esta entidad corresponde a un tipo de actividad en el cual el usuario de *Twitter*, a través de un hashtag y a continuación un mensaje (*tweet*), participa aportando su opinión o criterio de forma libre sobre un tema concreto.
- Características:
 - Nombre de la entidad: RespuestaLibre.
 - Atributo/s identificador/es principal/es: id_actividad.
 - Atributo/s identificador/es alternativo/s: -.
 - Atributo/s heredado/s: -.
 - Número de atributos: 2.
- Atributos propios:
 - Id_actividad
 - Definición del atributo: es el código que identifica a la actividad unívocamente.
 - Definición del dominio: números naturales.
 - Carácter: obligatorio.
 - Hashtag

- Definición del atributo: define el hashtag para poder participar en una pregunta / respuesta libre de forma unívoca.
- Definición del dominio: caracteres alfanuméricos.
- Carácter: obligatorio.
- Diagrama:

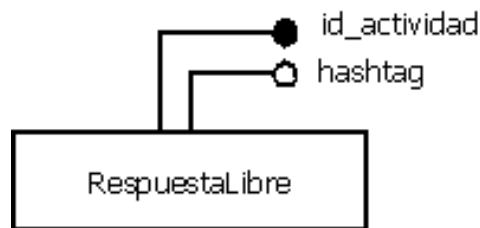


Figura 7.9: Ejemplo tipo de entidad *RespuestaLibre*

- Ejemplo:

Tabla 7.9: Ejemplo práctico tipo de entidad *RespuestaLibre*

RespuestaLibre	
Id_actividad	134
Hashtag	#trabajarEn

7.3. ANÁLISIS DE INTERRELACIONES

A continuación se analizarán las relaciones existentes entre las distintas entidades que se han descrito anteriormente. Para ello, se indicarán los siguientes apartados de cada relación:

- Descripción: se indicarán los tipos de entidad que participan en la interrelación, así como la explicación de la misma.
- Características: se mostrarán los siguientes datos sobre la interrelación:
 - Nombre del tipo de interrelación.
 - Grado del tipo de interrelación (número de entidades que participan en la interrelación).
 - Tipo de interrelación.

- Cardinalidad con la que cada tipo de entidad interviene en la interrelación.
 - Número de atributos pertenecientes a la interrelación.
- Diagrama: representación gráfica del tipo de interrelación.
- Ejemplo: tabla que representa un caso concreto.

A continuación se realizará el análisis de las interrelaciones del modelo conceptual de datos.

7.3.1. TIPO DE INTERRELACIÓN *USUARIOAPLICACION-ACTIVIDAD*

- Descripción: este tipo de interrelación manifiesta que los usuarios de nuestra aplicación *SIGAPTWIT*, proponen una serie de actividades. Es un tipo de interrelación débil por existencia que relaciona a una entidad fuerte (*UsuarioAplicacion*) con una entidad débil (*Actividad*), ya que una actividad no puede ser creada sin que exista previamente un usuario creado en la aplicación. De esta forma cada actividad (*Sorteo*, *Votacion* o *RespuestaLibre*) está asociada a un usuario, el cual las ha creado previamente.
- Características:
 - Nombre del tipo de interrelación: *U-A*
 - Grado del tipo de interrelación: 2.
 - Tipo de interrelación: 1:N.
 - Cardinalidad con la que interviene cada tipo de entidad:
 - Tipo de entidad *UsuarioAplicacion*: (0,n)
 - Tipo de entidad *Actividad*: (1,1)
 - Número de atributos: 0.
- Diagrama:

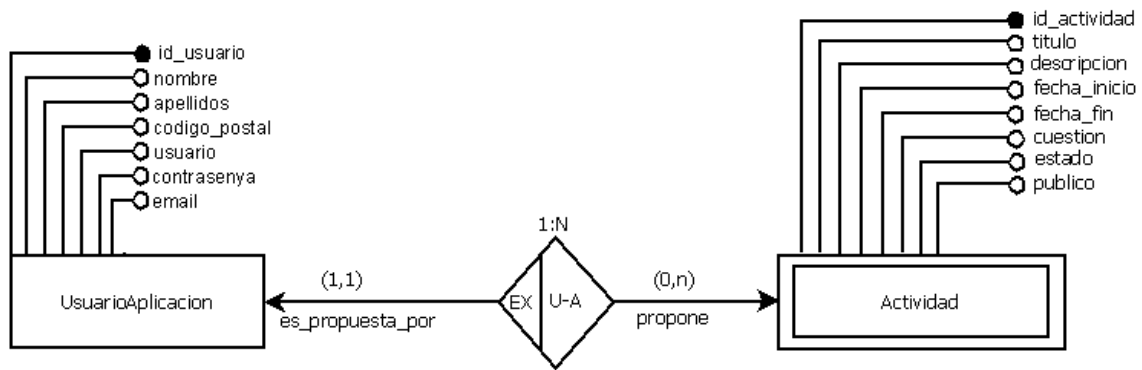


Figura 7.10: Ejemplo tipo de interrelación *UsuarioAplicacion-Actividad*

• Ejemplo:

Tabla 7.10: Ejemplo práctico tipo de interrelación *UsuarioAplicacion-Actividad*

UsuarioAplicacion		Actividad	
id_usuario	usuario	id_actividad	Titulo
1	i32topom	1	Sorteo Uco
1	i32topom	2	Votación profesorado
1	i32topom	3	Respuesta Libre Opciones Laborales
2	i42segar	4	Votación Equipos

7.3.2. TIPO DE INTERRELACIÓN *VOTACION-OPCION*

- Descripción: este tipo de interrelación indica que el tipo de entidad *Votacion*, tiene una serie de opciones posibles, que se encuentran en el tipo de entidad *Opcion*. Es un tipo de interrelación débil por existencia que relaciona a una entidad fuerte – *Votacion* – con una entidad débil – *Opcion* –, ya que no pueden existir opciones de una votación, sin estar creada previamente, la votación propiamente dicha.
- Características:
 - Nombre del tipo de interrelación: *V-O*.
 - Grado del tipo de interrelación: 2.
 - Tipo de interrelación: 1:N.

- Cardinalidad con la que interviene cada tipo de entidad:
 - Tipo de entidad *Votacion*: (1,n)
 - Tipo de entidad *Opcion*: (1,1)
- Número de atributos: 0.
- Diagrama:

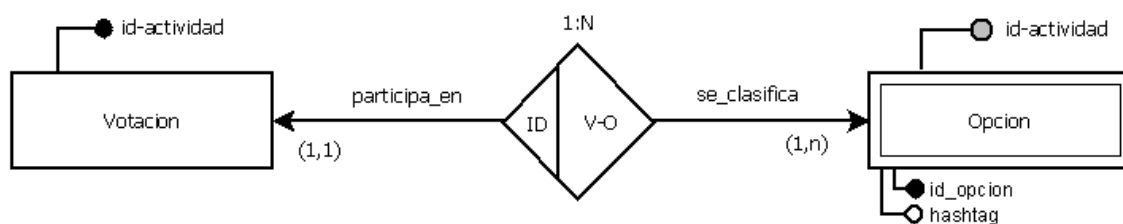


Figura 7.11: Ejemplo tipo de interrelación *Votacion-Opcion*

- Ejemplo:

Tabla 7.11: Ejemplo práctico tipo de interrelación *Votacion-Opcion*

Votacion	Opcion		
id_actividad	id_opcion	hashtag	id_actividad
1	1	#votacionUCO	1
1	2	#votacionProfesorado	1
1	3	#votacionAlumnado	1
2	4	#sorteoCURSO	2
2	5	#sorteoVIAJE	2

7.3.3. TIPO DE INTERRELACIÓN *SORTEO-GANADOR*

- Descripción: este tipo de interrelación indica que la actividad *Sorteo* tiene desde ninguno a varios ganadores, que se encuentran en el tipo de entidad *Ganador*. Es un tipo de interrelación débil por existencia que relaciona a una entidad fuerte –*Sorteo*– con una entidad débil –*Ganador*–, ya que no pueden existir uno o varios ganadores, sin que exista previamente un sorteo creado.
- Características:
 - Nombre del tipo de interrelación: S-G.

- Grado del tipo de interrelación: 2.
- Tipo de interrelación: 1:N.
- Cardinalidad con la que interviene cada tipo de entidad:
 - Tipo de entidad Sorteo: (0,n)
 - Tipo de entidad *Ganador*: (1,1)
- Número de atributos: 0.

• Diagrama:

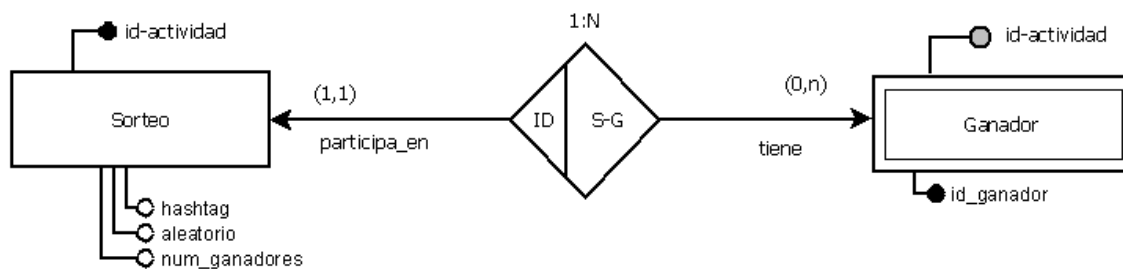


Figura 7.12: Ejemplo tipo de interrelación *Sorteo-Ganador*

• Ejemplo:

Tabla 7.12: Ejemplo práctico tipo de interrelación *Sorteo-Ganador*

Sorteo		Ganador	
id_actividad	hashtag	id_ganador	id_actividad
1	#sorteoUCO	1	1
1	#sorteoUCO	2	1
1	#sorteoUCO	3	1
2	#sorteoEntradasGRATIS	4	2
3	#sorteoRespuestaCorrecta	5	3

7.3.4. TIPO DE INTERRELACIÓN *ACTIVIDAD-TWEET*

- Descripción: este tipo de interrelación indica la relación entre los tipos de entidad *Actividad* y *Tweet*, de forma que uno o varios *tweets* están relacionados con una misma actividad. Es un tipo de interrelación débil por identificación. La debilidad del tipo de entidad *Tweet* es tanto por existencia como por identificación. De esta forma, una entidad *Tweet* no existirá si no

existe una entidad *Actividad* asociada con la cual esté relacionada a través del tipo de interrelación (A-T), y además, una entidad *Tweet* no puede ser identificada por sí sola, sino que es necesario identificar a la entidad *Actividad* con la cual está relacionada para poder diferenciarla del resto de las entidades *Tweet*.

- Características:

- Nombre del tipo de interrelación: A-T.
- Grado del tipo de interrelación: 2.
- Tipo de interrelación: 1:N.
- Cardinalidad con la que interviene cada tipo de entidad:
 - Tipo de entidad *Actividad*: (0,n)
 - Tipo de entidad *Tweet*: (1,1)
- Número de atributos: 0.

- Diagrama:

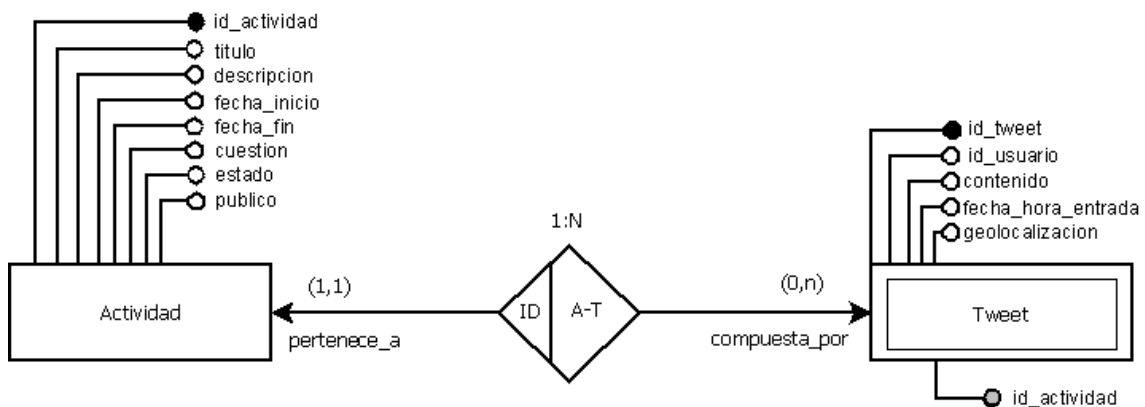


Figura 7.13: Ejemplo tipo de interrelación *Actividad-Tweet*

- Ejemplo:

Tabla 7.13: Ejemplo práctico tipo de interrelación *Actividad-Tweet*

Actividad	Tweet		
id_actividad	id_tweet	id_usuario	id_actividad
1	123456	ara_251	1
2	789101	antonio_zep	4

3	121314	cris_zamorano	2
4	151617	roberto_2010	5
5	181929	maitemmm	2

7.3.5. TIPO DE INTERRELACIÓN *TWEET-GANADOR*

- Descripción: este tipo de interrelación indica la relación entre los tipos de entidad *Tweet* y *Ganador*, de forma que un *tweet* puede resultar o no ganador de una actividad tipo *Sorteo*. Es un tipo de interrelación débil por identificación. La debilidad del tipo de entidad *Ganador* es tanto por existencia como por identificación. De esta forma, una entidad *Ganador* no existirá si no existe una entidad *Tweet* asociada con la cual esté relacionada a través del tipo de interrelación (*T-G*), y además, una entidad *Ganador* no puede ser identificada por sí sola, sino que es necesario identificar a la entidad *Tweet* con la cual está relacionada para poder diferenciarla del resto de las entidades del tipo *Ganador*.
- Características:
 - Nombre del tipo de interrelación: T-G.
 - Grado del tipo de interrelación: 2.
 - Tipo de interrelación: 1:1.
 - Cardinalidad con la que interviene cada tipo de entidad:
 - Tipo de entidad *Tweet*: (0,1)
 - Tipo de entidad *Ganador*: (1,1)
 - Número de atributos: 0.
- Diagrama:

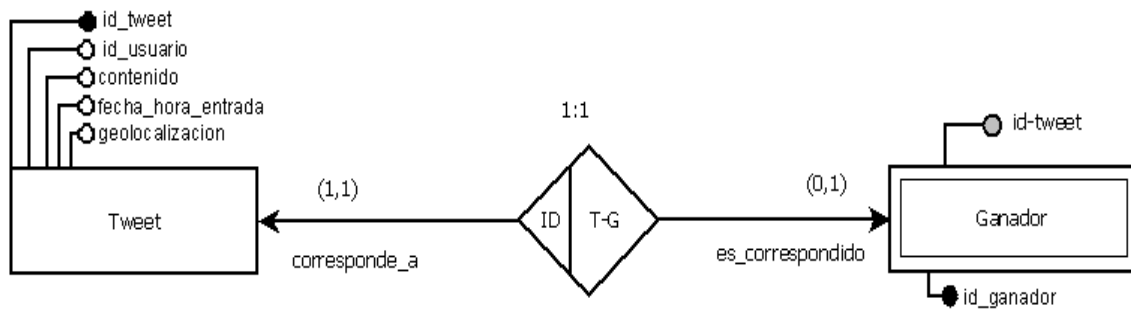


Figura 7.14: Ejemplo tipo de interrelación *Tweet-Ganador*

- Ejemplo:

Tabla 7.14: Ejemplo práctico tipo de interrelación *Tweet-Ganador*

Tweet	Ganador		
	id_ganador	id_tweet	id_sorteo
123456	1	123456	1
789101	2	789101	1
121314	3	209873	3
151617	4	100000	2
209873	5	222000	2

7.4. MODELO E-R

A continuación, se muestra el Modelo Entidad-Relación representado en la figura 7.15

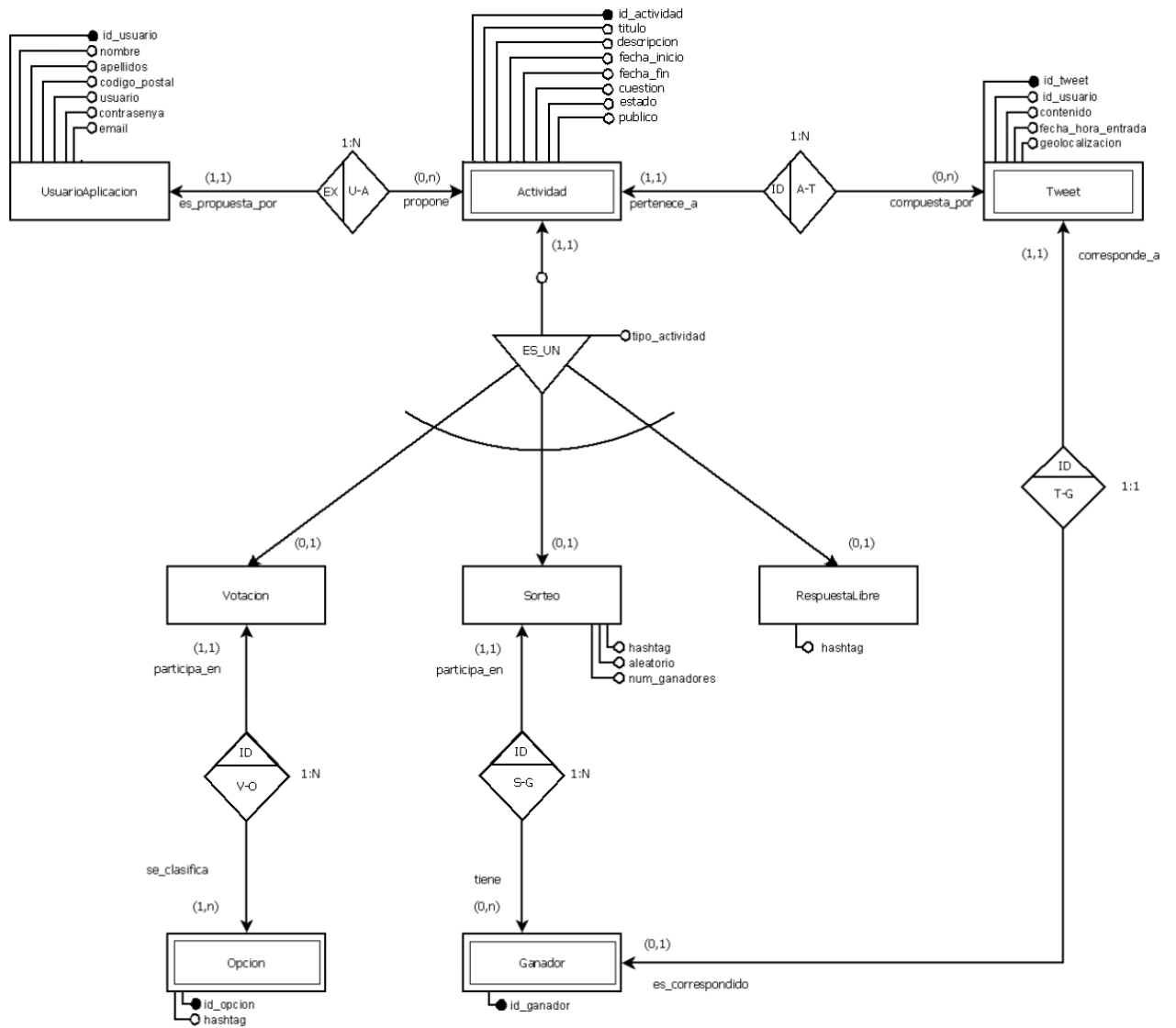


Figura 7.15: Modelo E-R

8. DESCRIPCIÓN FUNCIONAL

8.1. INTRODUCCIÓN

Los diagramas de casos de uso representan cómo un usuario (actor) opera con el sistema, además de la forma, tipo y orden en la que los elementos de dicho sistema interactúan entre sí (operaciones o casos de uso). Los casos de uso se emplean para capturar el comportamiento deseado del sistema, sin tener que especificar cómo se implementa ese comportamiento. Es decir, estos diagramas describen lo que hace el sistema desde el punto de vista de un observador externo, enfatizando el qué más que el cómo. Plantean escenarios en los que se muestra qué sucede cuando alguien interactúa con el sistema, proporcionando un resumen para una tarea u objetivo.

8.2. IDENTIFICACIÓN DE LOS ACTORES

A continuación se detalla el papel que juega cada uno de los actores dentro del sistema *SIGAPTWIT*, de acuerdo a los objetivos y los requisitos que se han expuesto en los capítulos anteriores. La plantilla que se va a utilizar es la siguiente:

Tabla 8.1: Plantilla para definir los actores

Código. Nombre del actor	
Descripción:	Descripción de cada uno de los actores que intervienen en la aplicación y el rol que representan.

8.2.1. USUARIO WEB

Tabla 8.2: Actor *Usuario web*

AC01.UsuarioWeb	
Descripción:	Usuario cuyo rol es crear y manejar una o varias de las actividades posibles de interacción social que permite la aplicación, esto es, un sorteo, una pregunta / respuesta libre o una votación. Este rol puede a su vez diferenciarse entre usuario web público, el que accede al sistema sin identificarse, y el privado, que está registrado en el sistema.

8.2.2. TWITTER

Tabla 8.3: Actor *Twitter*

AC02. Twitter	
Descripción:	Desempeña el rol de red social propiamente dicho, comportándose como un “sistema” independiente, y con el cual interactúa la aplicación.

8.2.3. TIEMPO

Tabla 8.4: Actor *Tiempo*

AC03. Tiempo	
Descripción:	Actor asociado a casos de uso que capturan una funcionalidad que debe ser lanzada en un momento específico.

8.3. ANÁLISIS DE LOS CASOS DE USO

Los casos de uso son requerimientos funcionales que describen, de una manera detallada, el comportamiento del sistema con los distintos actores que interactúan con él y no tienen por qué definir todos los requerimientos. Representan el hilo conductor del sistema o aplicación, además del carácter funcional de los objetivos.

Mediante los casos de uso se conduce el proceso a pesar de que el objetivo no será hacer una descripción absoluta del sistema, sino más bien una descripción de los aspectos funcionales más importantes del mismo. Esta descripción al comienzo del desarrollo del software, permite empezar a comprender el sistema y llegar a los puntos más relevantes.

La plantilla utilizada para este proceso, se muestra en la siguiente tabla:

Tabla 8.5: Plantilla para definir casos de uso

ID:	Número identificador del caso de uso
Breve descripción:	Descripción del caso de uso de forma concisa.
Actores principales:	Definición de los actores cuya función es relevante.
Actores secundarios:	Definición de actores que participan de forma complementaria.
Pre-Condiciones:	Conjunto de condiciones que deben cumplirse para que el caso de uso se realice de forma correcta.
Flujo principal:	Secuencia de pasos necesarios a seguir para la correcta realización del caso de uso.
Post-Condiciones:	Conjunto de resultados obtenidos como conclusión del caso de uso.

8.3.1. CASO DE USO 0. CONTEXTO DEL SISTEMA

Se pretende mostrar con este caso de uso una visión global de la funcionalidad que el sistema debe proporcionar a los distintos actores del mismo. En la figura 8.1 se muestra el diagrama de caso de uso 0, que representa el contexto del sistema.



Figura 8.1: Caso de uso 0. Contexto del sistema

En los siguientes apartados se detalla el funcionamiento de *SIGAPTWIT* mediante diagramas de casos de uso.

8.3.2. CASO DE USO 1. GESTIONAR SUBSISTEMA WEB

Este caso de uso está compuesto por:

- GestionarUsuario
- GestionarActividad

Dichas actividades se detallan a continuación, cada una de ellas con su correspondiente diagrama de caso de uso y diccionario de datos.

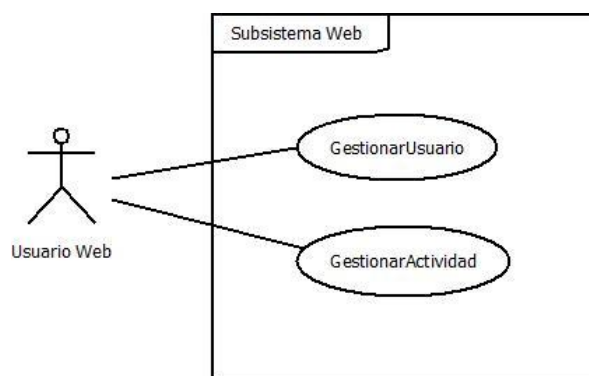


Figura 8.2: Caso de uso 1. *Gestionar Subsistema web*

8.3.2.1. Caso de uso 1.1. GestionarUsuario

Este caso de uso permite llevar a cabo las distintas tareas relacionadas con la gestión de usuarios.

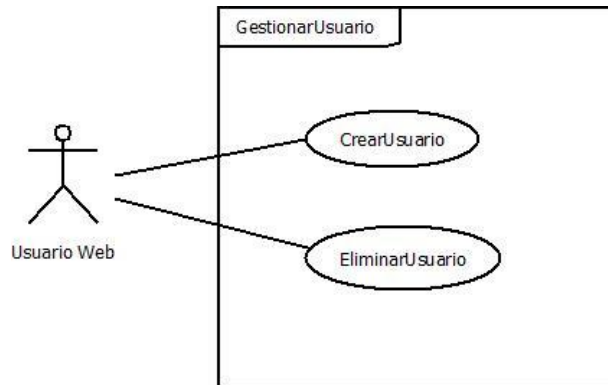


Figura 8.3: Caso de uso 1.1. *GestionarUsuario*

- Caso de uso 1.1.1. CrearUsuario

Tabla 8.6: Caso de uso 1.1.1. Crear usuario

ID:	1. CrearUsuario
Breve descripción:	Proceso mediante el cual el actor crea un nuevo usuario en el sistema.
Actores principales:	Usuario web.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El usuario no debe existir previamente en el sistema.
Flujo principal:	1. El usuario rellena el formulario de registro con sus datos. 2. El sistema verifica los datos.
Post-Condiciones:	1. El usuario queda registrado en la base de datos del sistema como nuevo usuario.

- Caso de uso 1.1.2. EliminarUsuario

Tabla 8.7: Caso de uso 1.1.2. Eliminar usuario

ID:	2. EliminarUsuario
Breve descripción:	Proceso mediante el cual el actor se da de baja en la aplicación.
Actores principales:	Usuario web.
Actores	Ninguno.

secundarios:	
Pre-Condiciones:	<ol style="list-style-type: none"> 1. El usuario debe existir previamente en el sistema. 2. El usuario debe haber iniciado sesión en la aplicación
Flujo principal:	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación correctamente. 2. El usuario confirma la eliminación de su cuenta.
Post-Condiciones:	<ol style="list-style-type: none"> 1. Todos los datos del usuario quedan eliminados del sistema.

8.3.2.2. Caso de uso 1.2. GestionarActividad

Este caso de uso permite llevar a cabo las distintas tareas relacionadas con la gestión de una actividad.

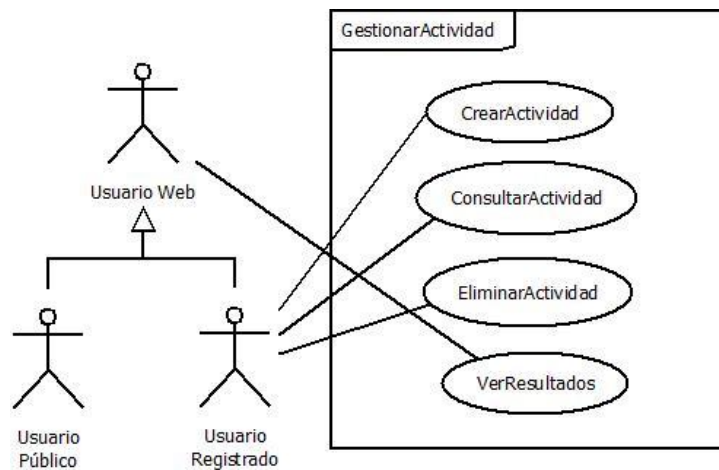


Figura 8.4: Caso de uso 1.2. *GestionarActividad*

- Caso de uso 1.2.1. CrearActividad

Tabla 8.8: Caso de uso 1.2.1. Crear actividad

ID:	3. CrearActividad
Breve descripción:	Proceso mediante el cual el actor crea una actividad de tipo votación, sorteo o pregunta / respuesta libre.
Actores principales:	Usuario registrado.
Actores secundarios:	Ninguno.
Pre-Condiciones:	<ol style="list-style-type: none"> 1. El usuario debe existir previamente en el sistema. 2. El usuario debe haber iniciado sesión en la aplicación
Flujo principal:	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación correctamente. 2. El usuario selecciona el tipo de actividad a crear.

	3. El usuario rellena el formulario de registro de la actividad.
Post-Condiciones:	1. La actividad queda creada.

- Caso de uso 1.2.2. ConsultarActividad

Tabla 8.9: Caso de uso 1.2.2. Consultar actividad

ID:	4. ConsultarActividad
Breve descripción:	Proceso mediante el cual el actor consulta los datos de una actividad, bien sea una votación, un sorteo o una pregunta / respuesta libre.
Actores principales:	Usuario registrado.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El usuario debe existir previamente en el sistema. 2. El usuario debe haber iniciado sesión en la aplicación.
Flujo principal:	1. El usuario accede a la aplicación correctamente. 2. El usuario selecciona el tipo de actividad a consultar.
Post-Condiciones:	Los datos de la actividad elegida son mostrados por pantalla.

- Caso de uso 1.2.3. EliminarActividad

Tabla 8.10: Caso de uso 1.2.3. Eliminar actividad

ID:	5. EliminarActividad
Breve descripción:	Proceso mediante el cual el actor elimina los datos de una actividad.
Actores principales:	Usuario registrado.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El usuario debe existir previamente en el sistema. 2. El usuario debe haber iniciado sesión en la aplicación.
Flujo principal:	1. El usuario accede a la aplicación correctamente. 2. El usuario selecciona el tipo de actividad a eliminar. 3. El usuario confirma la eliminación de la actividad.
Post-Condiciones:	Los datos de la actividad elegida son eliminados del sistema.

- Caso de uso 1.2.4. VerResultados

Tabla 8.11: Caso de uso 1.2.4. Ver resultados

ID:	6. VerResultados
Breve descripción:	Proceso mediante el cual el actor consulta los resultados de las actividades en ejecución o finalizadas.
Actores principales:	Usuario web.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. La actividad debe estar ejecutándose o finalizada. 2. Para actividades privadas, el usuario debe haber iniciado sesión correctamente en el sistema.
Flujo principal:	1. Si la actividad es pública: 1.1 El usuario accede al enlace de la página web donde se muestran los resultados públicos. 1.2 El usuario selecciona el tipo de actividad a consultar. 1.3 El usuario selecciona la opción de mostrar resultados. 2. Si la actividad es privada: 2.1 El usuario accede a la aplicación correctamente. 2.2 El usuario selecciona el tipo de actividad a consultar. 2.3 El usuario selecciona la opción de mostrar resultados.
Post-Condiciones:	Los datos de la actividad elegida son mostrados por el sistema.

8.3.3. CASO DE USO 2. GESTIONAR SUBSISTEMA CONTROLADOR

Este caso de uso está compuesto por tres operaciones, “ComprobarActividadesActivas”, “ContabilizarTweets” y “AlmacenarResultados”.

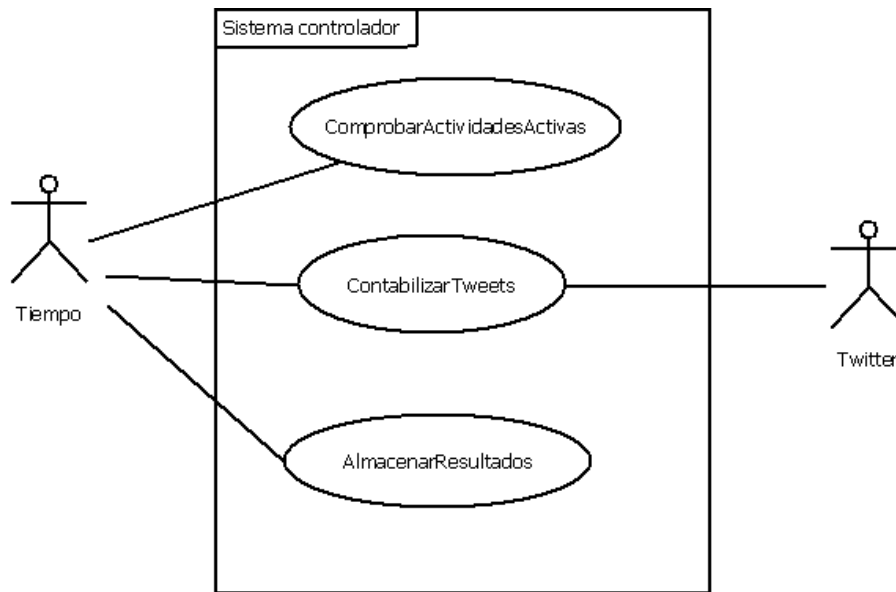


Figura 8.5: Caso de uso 2. *Gestionar Subsistema controlador*

8.3.3.1. Caso de uso 2.1. *ComprobarActividadesActivas*

Este caso de uso permite llevar a cabo las diferentes operaciones relacionadas con la comprobación de las actividades que deben comenzar a ejecutarse.

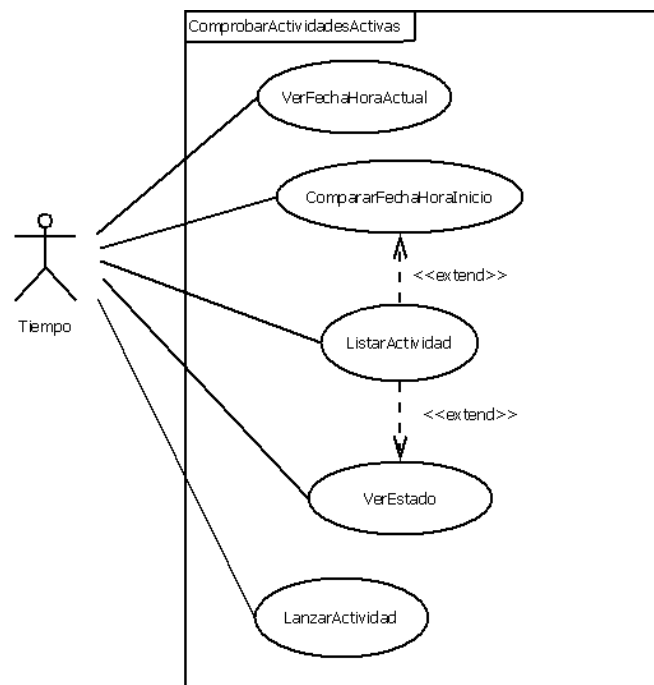


Figura 8.6: Caso de uso 2.1. *ComprobarActividadesActivas*

- Caso de uso 2.1.1. VerFechaHoraActual

Tabla 8.12: Caso de uso 2.1.1. Ver fecha hora actual

ID:	7. VerFechaHoraActual
Breve descripción:	Proceso mediante el cual el actor revisa la hora y fecha actuales.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema debe contar con un reloj.
Flujo principal:	1. El sistema captura la hora y fecha del reloj y la utiliza para una posterior comprobación.
Post-Condiciones:	1. El sistema dispone de la hora y fecha actuales.

- Caso de uso 2.1.2. CompararFechaHoraInicio

Tabla 8.13: Caso de uso 2.1.2. Comparar fecha hora inicio

ID:	8. CompararFechaHoraInicio
Breve descripción:	Proceso mediante el cual el actor revisa la fecha y hora actuales, y la fecha y hora que marca el inicio de una actividad concreta.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema debe disponer de la fecha y hora actuales. 2. El sistema debe tener acceso a la fecha y hora de inicio de una actividad.
Flujo principal:	1. El sistema compara si la fecha y hora actuales coinciden exactamente con la fecha y hora guardadas, para iniciar una actividad.
Post-Condiciones:	1. El sistema ha verificado que la hora y fecha de inicio de una actividad es igual, o no, a la hora y fecha actuales.

- Caso de uso 2.1.3. VerEstado

Tabla 8.14: Caso de uso 2.1.3. Ver estado

ID:	9. VerEstado
Breve descripción:	Proceso mediante el cual el actor inicia la ejecución de ver el estado que posee una actividad, en un momento dado. Este estado puede ser “En espera”, “Activa” o “Finalizada”.
Actores	Tiempo.

principales:	
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema puede obtener el estado de una actividad.
Flujo principal:	1. El sistema se conecta a la base de datos para ver el estado de una actividad.
Post-Condiciones:	1. El sistema guarda el estado de una actividad.

- Caso de uso 2.1.4. ListarActividad

Tabla 8.15: Caso de uso 2.1.4. Listar actividad

ID:	10. ListarActividad
Breve descripción:	Proceso mediante el cual el actor, tras comprobar que una actividad cumple con los requisitos de fecha, hora y estado, utiliza todos los datos que definen dicha actividad.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. La fecha y hora actuales coincidan con la fecha y hora de inicio de la actividad. 2. El estado de esa actividad debe ser “En espera”.
Flujo principal:	1. El sistema recoge de la base de datos, todos aquellos datos que necesita para lanzar la actividad.
Post-Condiciones:	1. La actividad queda preparada para iniciar su ejecución.

- Caso de uso 2.1.5.LanzarActividad

Tabla 8.16: Caso de uso 2.1.5. Lanzar actividad

ID:	11. LanzarActividad
Breve descripción:	Proceso mediante el cual el actor da la orden al sistema para que comience a ejecutarse la actividad.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. Los datos necesarios de una actividad deben estar recopilados.
Flujo principal:	1. El sistema realiza la petición de conexión a <i>Twitter</i> .

	2. El sistema se conecta a <i>Twitter</i> . 3. El sistema recopila la información que necesita para ejecutar su tarea de forma óptima.
Post-Condiciones:	1. La actividad queda en estado de ejecución durante el tiempo indicado.

8.3.3.2. Caso de uso 2.2. ContabilizarTweets

Este caso de uso permite llevar a cabo la operación de encontrar aquellos *tweets* que cumplan los requisitos que marca el *hashtag* de la actividad.

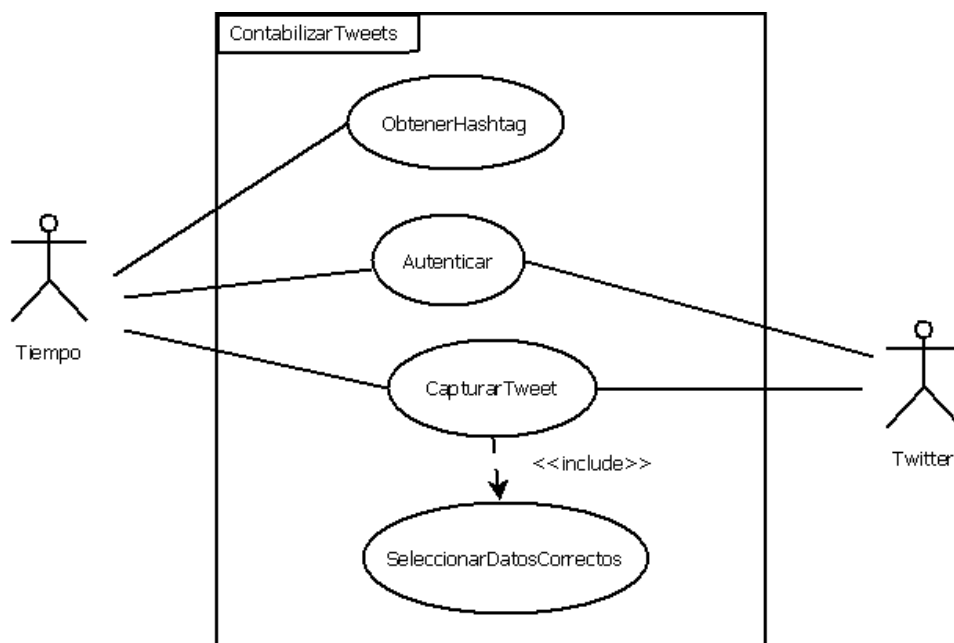


Figura 8.7: Caso de uso 2.2. ContabilizarTweets

El conjunto de casos de uso asociados a la contabilización de *tweets* referentes a una actividad es el siguiente:

- Caso de uso 2.2.1. ObtenerHashtag

Tabla 8.17: Caso de uso 2.2.1. Obtener hashtag

ID:	12. ObtenerHashtag
Breve descripción:	Proceso mediante el cual el actor inicia la acción de obtener el hashtag para poder lanzar una actividad.
Actores principales:	Tiempo.

Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema debe tener acceso a la base de datos. 2. El campo de la actividad dedicado para almacenar el hashtag esté relleno de forma correcta.
Flujo principal:	1. El sistema accede a la base de datos. 2. El sistema identifica una actividad activa. 3. El sistema almacena el hashtag correspondiente a esa actividad.
Post-Condiciones:	1. El hashtag queda almacenado.

- Caso de uso 2.2.2. Autenticar

Tabla 8.18: Caso de uso 2.2.2. Autenticar

ID:	13. Autenticar
Breve descripción:	Proceso mediante el cual es necesario autenticarse en <i>Twitter</i> , para poder obtener datos que nuestro sistema necesita para cumplir sus requisitos.
Actores principales:	<i>Twitter</i> .
Actores secundarios:	Tiempo.
Pre-Condiciones:	1. El sistema debe disponer de unas credenciales autorizadas por <i>Twitter</i> .
Flujo principal:	1. El sistema envía la petición de conexión a <i>Twitter</i> . 2. <i>Twitter</i> le solicita sus credenciales. 3. El sistema se conecta con sus credenciales. 4. <i>Twitter</i> permite la conexión.
Post-Condiciones:	1. El sistema queda autenticado por <i>Twitter</i> y puede comunicarse con él.

- Caso de uso 2.2.3. CapturarTweet

Tabla 8.19: Caso de uso 2.2.3. Capturar tweet

ID:	14. CapturarTweet
Breve descripción:	Proceso mediante el cual el sistema se mantiene a la espera de que un usuario envíe un tweet que cumple con las condiciones de la actividad planteada, almacena toda la información

	referente de ese tweet para un posterior uso.
Actores principales:	Ninguno.
Actores secundarios:	<i>Twitter.</i>
Pre-Condiciones:	1. Debe existir un tweet publicado que cumpla con el hashtag que propone la aplicación.
Flujo principal:	1. El sistema se mantiene comprobando hashtag de tweets publicados. 2. El sistema encuentra un hashtag igual que el indicado en la actividad. 3. El sistema recupera toda la información que acompaña a ese tweet y la almacena para un posterior tratamiento.
Post-Condiciones:	1. El tweet queda capturado y almacenado.

- Caso de uso 2.2.4. SeleccionarDatosCorrectos

Tabla 8.20: Caso de uso 2.2.4. Seleccionar datos correctos

ID:	15. SeleccionarDatosCorrectos
Breve descripción:	Proceso mediante el cual el sistema, una vez capturada toda la información acerca de un tweet válido, utiliza sólo aquellos datos relevantes y necesarios para cumplir con su funcionalidad.
Actores principales:	Tiempo.
Actores secundarios:	<i>Twitter.</i>
Pre-Condiciones:	1. Debe existir previamente un tweet válido publicado, del que capturar información.
Flujo principal:	1. El sistema almacena los datos de un tweet válido. 2. El sistema clasifica aquella información que es necesaria. 3. El sistema guarda la información relevante. 4. El resto de información del tweet es descartado.
Post-Condiciones:	1. Los datos del tweet quedan almacenados para un posterior tratamiento.

8.3.3.3. Caso de uso 2.3. AlmacenarResultados

Este caso de uso permite llevar a cabo las diferentes operaciones de inserción de resultados en la base de datos, para mostrarlos a través del subsistema web a la finalización de una actividad.

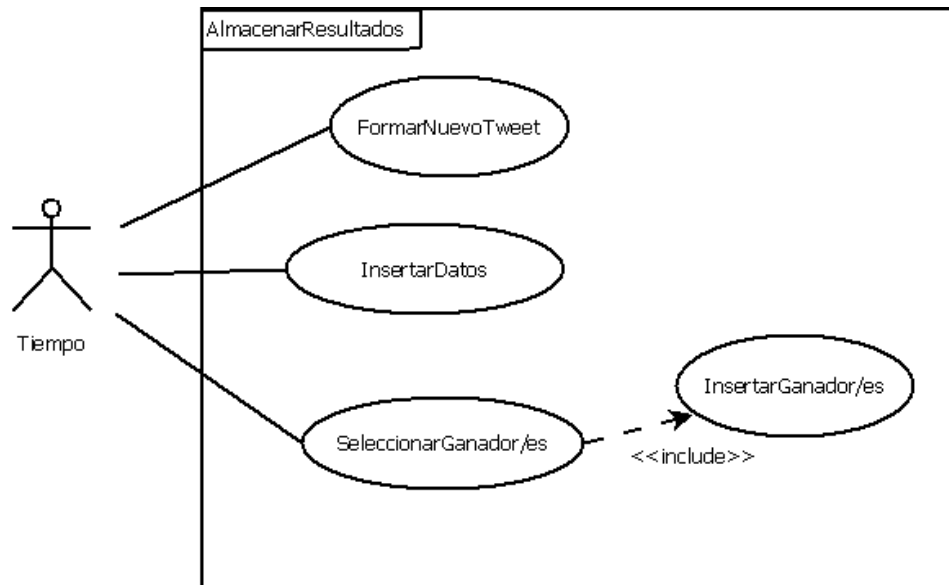


Figura 8.8: Caso de uso 2.3. *AlmacenarResultados*

El conjunto de casos de uso asociados al almacenamiento de resultados, es el siguiente:

- Caso de uso 2.3.1. FormarNuevoTweet

Tabla 8.21: Caso de uso 2.3.1. Formar nuevo tweet

ID:	16. FormarNuevoTweet
Breve descripción:	Proceso por el cual el sistema una vez seleccionados los datos que necesita para cumplir los requisitos, crea un nuevo tweet, preparado para insertarse en la base de datos.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema debe contar con información previa procedente de <i>Twitter</i> .
Flujo principal:	1. El sistema forma un tweet propio con el cual, más adelante, realizará una serie de funcionalidades.

Post-Condiciones:	1. El tweet que el sistema necesita queda realizado.
--------------------------	--

- Caso de uso 2.3.2. InsertarDatos

Tabla 8.22: Caso de uso 2.3.2. Insertar datos

ID:	17. InsertarDatos
Breve descripción:	Proceso mediante el cual el sistema inicia una conexión a la base de datos y almacena en la misma los datos del nuevo tweet.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema debe contar con un tweet formado, propio de la aplicación. 2. La conexión a la base de datos debe estar operativa.
Flujo principal:	1. El sistema registra en la base de datos la información referente al tweet.
Post-Condiciones:	1. Los datos del tweet permanecen registrados en la base de datos.

- Caso de uso 2.3.3. SeleccionarGanadores

Tabla 8.23: Caso de uso 2.3.3. Seleccionar ganadores

ID:	18. Seleccionar Ganador/es
Breve descripción:	Proceso mediante el cual una vez finalizada una actividad de tipo sorteo, el sistema selecciona un número de usuarios aleatorio o por orden de llegada, indicado previamente en la definición de la actividad, que resultarán los ganadores de dicha actividad tipo sorteo.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. La actividad sorteo debe estar finalizada. 2. Deben existir un número mayor que cero de participantes en el sorteo. 2. Debe haber sido especificado el proceso de selección de ganador/es.
Flujo principal:	1. El sistema conecta con la base de datos.

	2. El sistema selecciona todos los usuarios filtrados como correctos que han participado en el sorteo. 2. El sistema recoge aleatoriamente, o por orden de llegada, el número de usuarios ganadores indicado.
Post-Condiciones:	1. Los ganadores quedan seleccionados.

- Caso de uso 2.3.4. InsertarGanadores

Tabla 8.24: Caso de uso 2.3.4. Insertar ganadores

ID:	19. Insertar Ganador/es
Breve descripción:	Proceso mediante el cual el sistema se encarga de registrar en la base de datos aquel o aquellos usuarios que resulten ganadores de un sorteo, para su posterior tratamiento informativo en el subsistema web.
Actores principales:	Tiempo.
Actores secundarios:	Ninguno.
Pre-Condiciones:	1. El sistema debe haber seleccionado al menos un ganador. 2. El sistema debe haber conectado de forma correcta con la base de datos.
Flujo principal:	1. El sistema registra los ganadores asociados a un sorteo.
Post-Condiciones:	1. Los ganadores quedan insertados y registrados.

8.4. DIAGRAMAS DE SECUENCIA

Los diagramas de secuencia son utilizados para la descripción del comportamiento. Estos diagramas describen la colaboración existente entre los objetos que componen el sistema. Se trata de un diagrama de interacción que contiene detalles de implementación de escenarios, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos, organizado todo en torno al tiempo. Para una fácil comprensión de este tipo de diagramas, hay que tener en cuenta que la secuencia temporal avanza de arriba abajo en el diagrama, y que los objetos involucrados en las operaciones se listan de izquierda a derecha, manteniendo un orden de participación dentro de la secuencia de paso de mensajes.

Los diagramas de secuencia se obtienen de forma prácticamente automática a partir de los casos de uso. Por esta razón se describen en el mismo orden que los casos de uso expuestos anteriormente para facilitar su comprensión. A continuación se exponen y describen los diagramas de secuencia resultantes:

8.4.1. SUBSISTEMA WEB

Para el subsistema web, existen dos diagramas de secuencia: *GestionarUsuario* y *GestionarActividad*, dentro de los cuales existen varias operaciones diferentes.

8.4.1.1. Diagrama de secuencia: *GestionarUsuario*

Se distinguirán dos operaciones diferentes en la gestión de usuarios:

- CrearUsuario

El diagrama de secuencia que se muestra en la figura 8.9 muestra la interacción de mensajes entre los objetos que participan en el CU *CrearUsuario*. El usuario web selecciona la opción “Registrar” de la interfaz gráfica web. A continuación, se le muestra el formulario en el que debe introducir los datos del nuevo usuario. Una vez introducidos, el usuario envía el formulario con el botón correspondiente. Seguidamente, el sistema almacena dicha información en la base de datos.

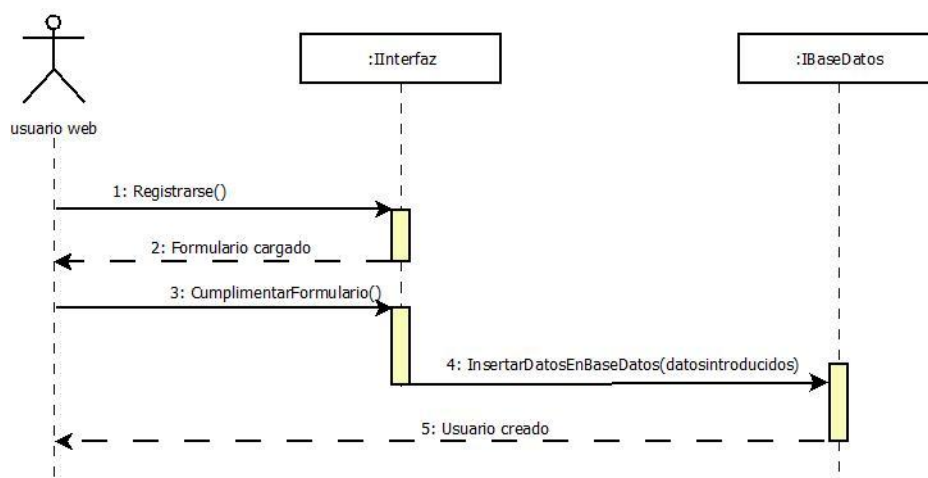


Figura 8.9: Diagrama de secuencia *CrearUsuario*

- EliminarUsuario

El diagrama de secuencia que se muestra en la figura 8.10 muestra la interacción de mensajes entre los objetos que participan en el CU *EliminarUsuario*. El usuario web, una vez identificado correctamente en el sistema, selecciona la opción “Desactivar cuenta” de la interfaz gráfica web. A continuación, se muestra una ventana para confirmar la eliminación del usuario. Seguidamente, el sistema elimina dicho usuario de la base de datos.

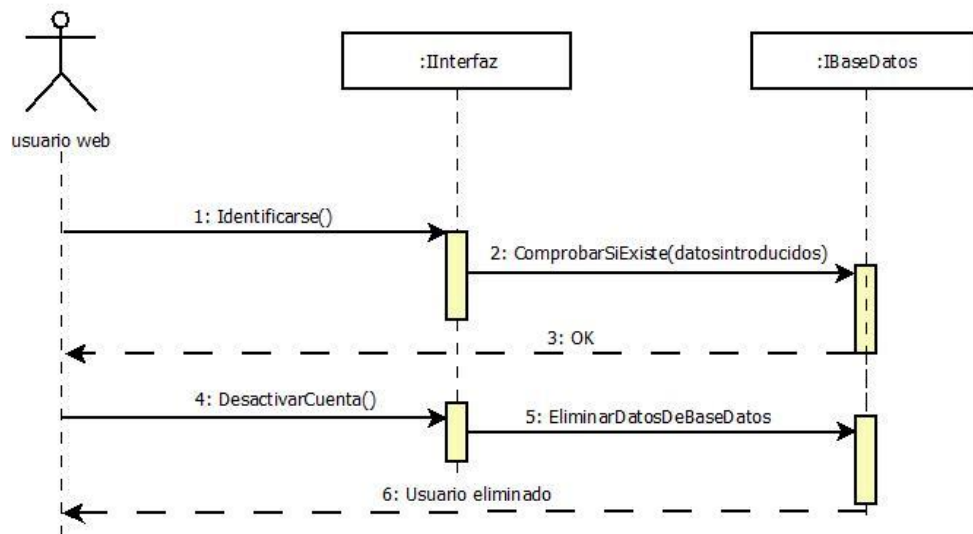


Figura 8.10: Diagrama de secuencia *EliminarUsuario*

8.4.1.2. Diagrama de secuencia: GestionarActividad

Se distinguirán dos operaciones diferentes en la gestión de actividades:

- CrearActividad

El diagrama de secuencia que se muestra en la figura 8.11 muestra la interacción de mensajes entre los objetos que participan en el CU *CrearActividad*. El usuario web, una vez identificado correctamente en el sistema, selecciona la actividad a crear. A continuación, se le muestra el formulario en el que debe introducir los datos de la nueva actividad. Una vez introducidos, el usuario envía el formulario con el botón correspondiente. Seguidamente, el sistema almacena dicha información en la base de datos.

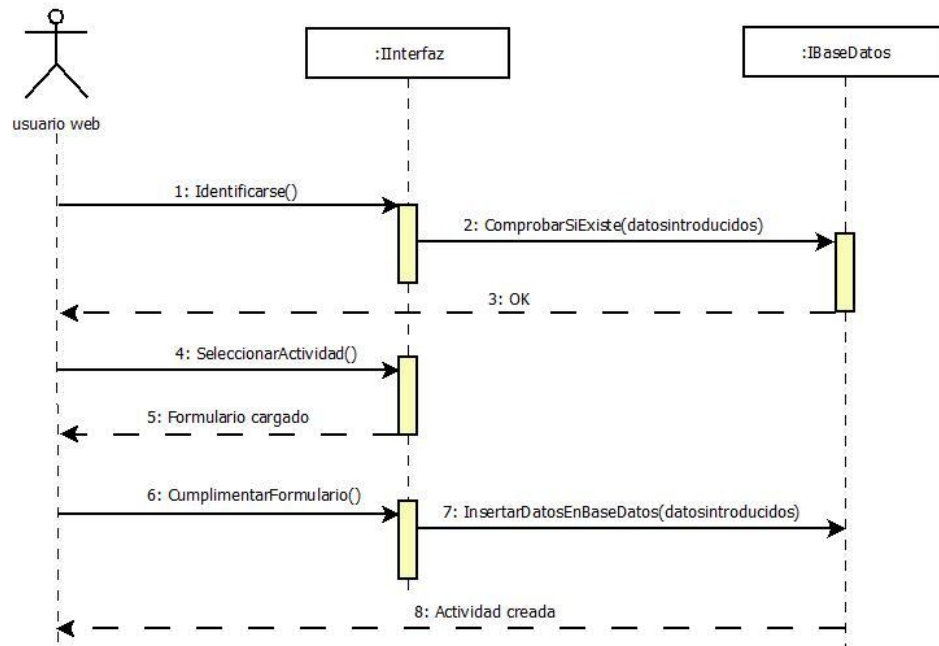


Figura 8.11: Diagrama de secuencia *CrearActividad*

- **EliminarActividad**

El diagrama de secuencia que se muestra en la figura 8.12 muestra la interacción de mensajes entre los objetos que participan en el CU *EliminarActividad*. El usuario web, una vez identificado correctamente en el sistema, selecciona el tipo de actividad a eliminar. A continuación, se le muestra los datos de las actividades. Una vez mostradas, el usuario elimina la actividad con el botón correspondiente. Seguidamente, el sistema elimina dicha actividad de la base de datos.

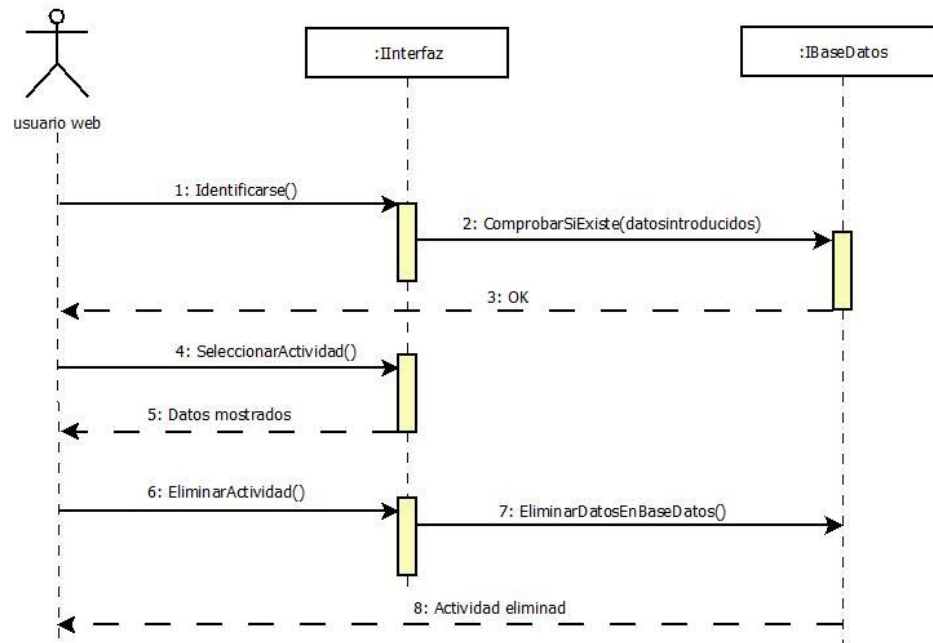


Figura 8.12: Diagrama de secuencia *EliminarActividad*

8.4.2. SUBSISTEMA CONTROLADOR

Para el subsistema web, existen dos diagramas de secuencia: *GestionarUsuario* y *GestionarActividad*, dentro de los cuales existen varias operaciones diferentes.

8.4.2.1. Diagrama de secuencia: *ComprobarActividadesActivas*

Se distinguirán tres operaciones diferentes en la gestión de comprobar actividades activas:

- *CompararFechaHoralInicio*

El diagrama de secuencia que se muestra en la figura 8.13 muestra la interacción de mensajes entre los objetos que participan en el CU *CompararFechaHoralInicio*. En este caso el rol desempeñado por el actor *Tiempo* es el que desencadena la acción de establecer conexión con la base de datos buscando actividades que deban iniciar su ejecución en ese mismo instante. Una vez finalizada esta acción se realiza la desconexión de la base de datos quedando comprobada la fecha y hora de inicio de una actividad.

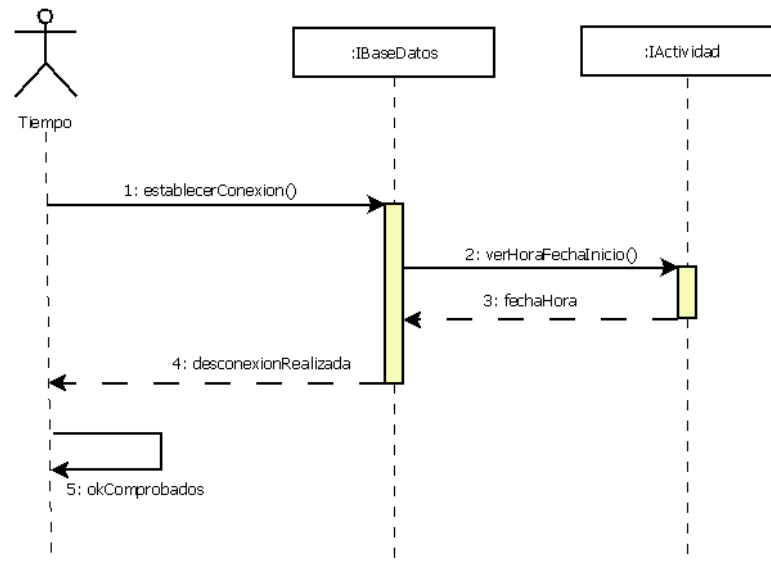


Figura 8.13: Diagrama de secuencia *CompararFechaHoraInicio*

- VerEstado

El diagrama de secuencia que se muestra en la figura 8.14 muestra la interacción de mensajes entre los objetos que participan en el CU *VerEstado*. El actor *Tiempo* es el encargado de desencadenar la acción de conexión con la base de datos y comprobar el estado de una actividad, obtener dicha información y finalmente realizar la desconexión de la base de datos.

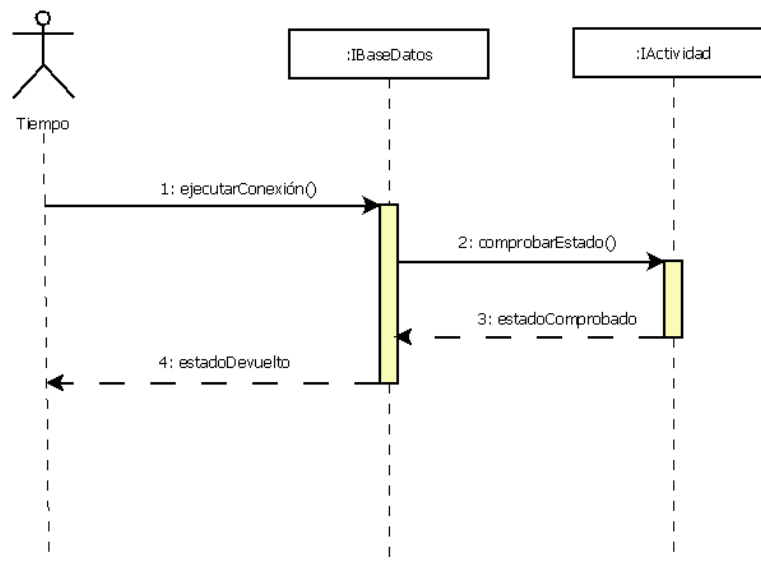


Figura 8.14: Diagrama de secuencia *VerEstado*

- ListarActividad

El diagrama de secuencia que se muestra en la figura 8.15 muestra la interacción de mensajes entre los objetos que participan en el CU *ListarActividad*. El actor *Tiempo* se encarga de iniciar la acción de establecer la conexión con la base de datos y obtener todos los datos de una actividad que va a ejecutarse. A continuación se realiza la desconexión de la base de datos, quedando los datos obtenidos para su posterior tratamiento por parte de la aplicación.

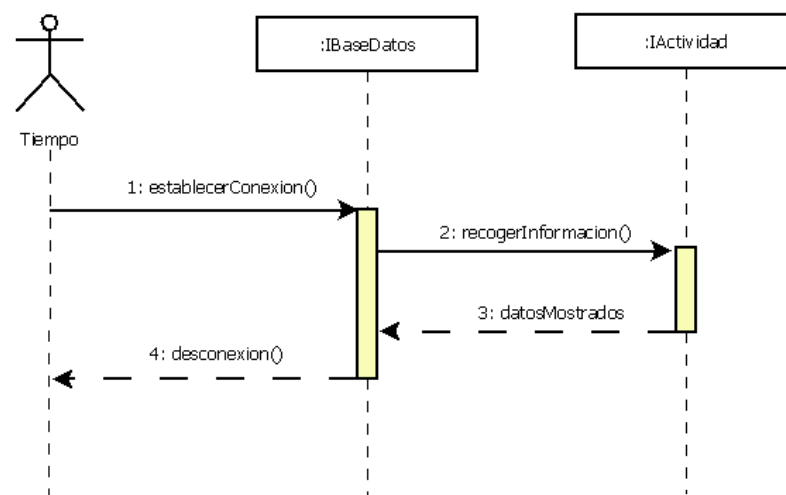


Figura 8.15: Diagrama de secuencia *ListarActividad*

8.4.2.2. Diagrama de secuencia: ContabilizarTweets

Se distinguirán tres operaciones diferentes en contabilización de *tweets*.

- ObtenerHashtag

El diagrama de secuencia que se muestra en la figura 8.16 muestra la interacción de mensajes entre los objetos que participan en el CU *ObtenerHashtag*. El actor *Tiempo* es el responsable de lanzar la acción de conectar con la base de datos y obtener de la misma los datos referentes al *hashtag* o los *hashtag* de una actividad con los cuales la aplicación llevará a cabo su funcionamiento. Una vez obtenidos se realiza el proceso de desconexión de la base de datos.

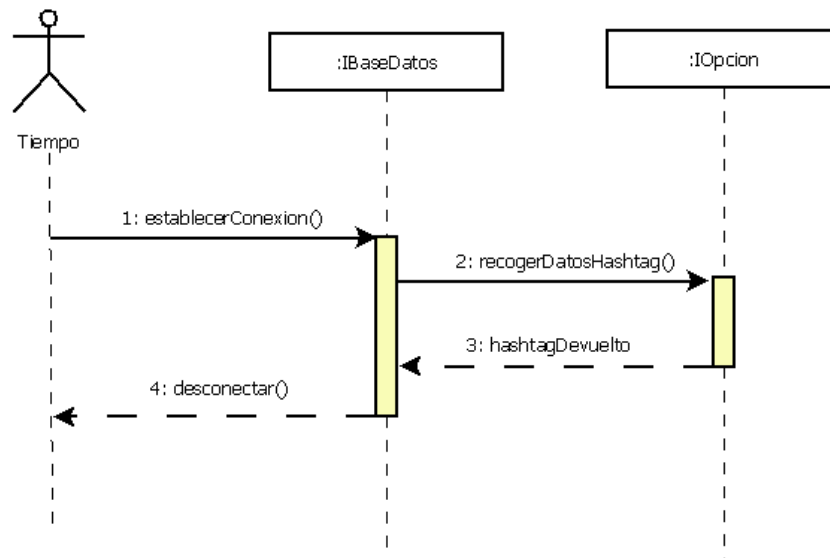


Figura 8.16: Diagrama de secuencia *ObtenerHashtag*

- Autenticar

El diagrama de secuencia que se muestra en la figura 8.17 muestra la interacción de mensajes entre los objetos que participan en el CU *Autenticar*. El actor *Tiempo* es el encargado de iniciar la acción de enviar los datos de la actividad que se va a ejecutar a un objeto de tipo *hilo*, que a su vez es el encargado de expandir un nuevo *hilo* que realiza la autenticación de usuario de tipo desarrollador para que *Twitter* permita la obtención de información que necesita el sistema controlador.

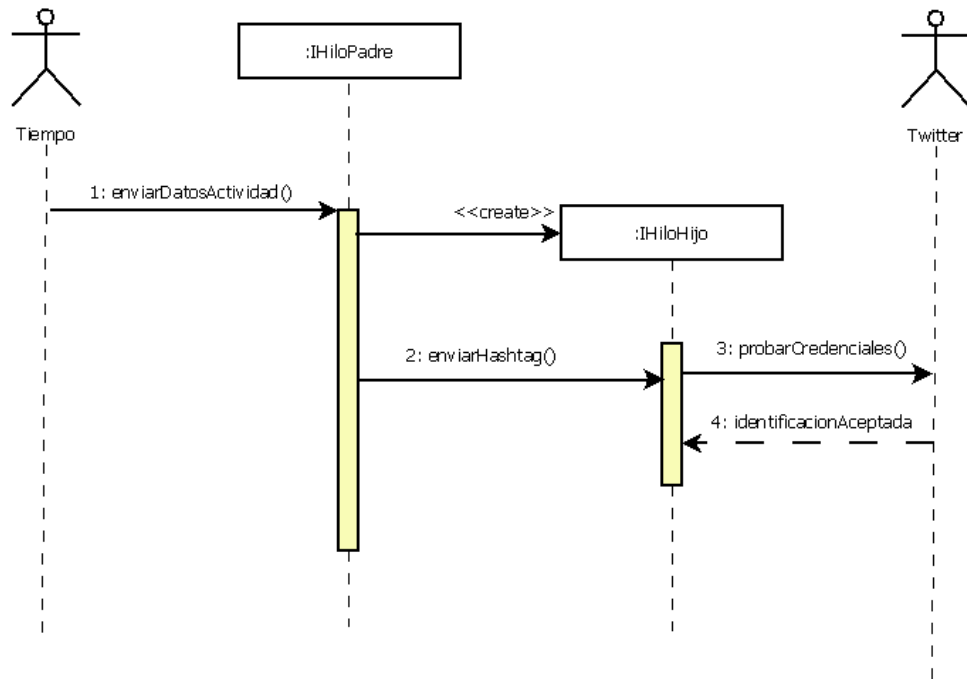


Figura 8.17: Diagrama de secuencia *Autenticar*

- CapturarTweet

El diagrama de secuencia que se muestra en la figura 8.18 muestra la interacción de mensajes entre los objetos que participan en el CU *CapturarTweet*. El actor *Tiempo* es el encargado de iniciar la acción de envío del *hashtag* de una actividad mediante un objeto de tipo *hilo* a otro de este mismo tipo, que a su vez se encarga comprobar *tweets* y obtener toda la información útil de aquellos que necesita el sistema controlador.

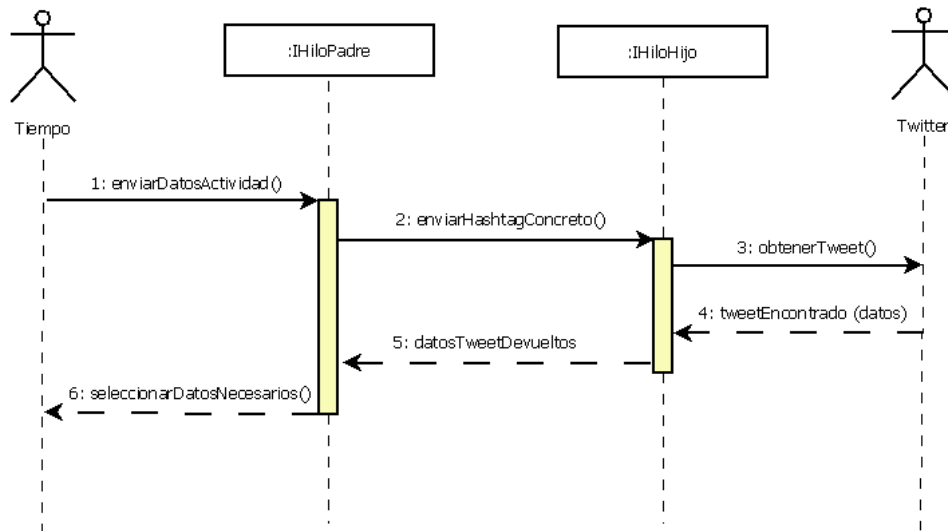


Figura 8.18: Diagrama de secuencia *CapturarTweet*

8.4.2.3. Diagrama de secuencia: *AlmacenarResultados*

Se distinguirán cuatro operaciones diferentes para el almacenamiento de los resultados.

- *FormarNuevoTweet*

El diagrama de secuencia que se muestra en la figura 8.19 muestra la interacción de mensajes entre los objetos que participan en el CU *FormarNuevoTweet*. El actor *Tiempo* es el encargado de iniciar la acción de enviar los datos de una actividad activa a un objeto de tipo *hilo*. Éste a su vez envía el *hashtag* a un objeto del mismo tipo solicitando únicamente la información del *tweet* con ese *hashtag* que necesita el sistema controlador. Dicha información queda enviada.

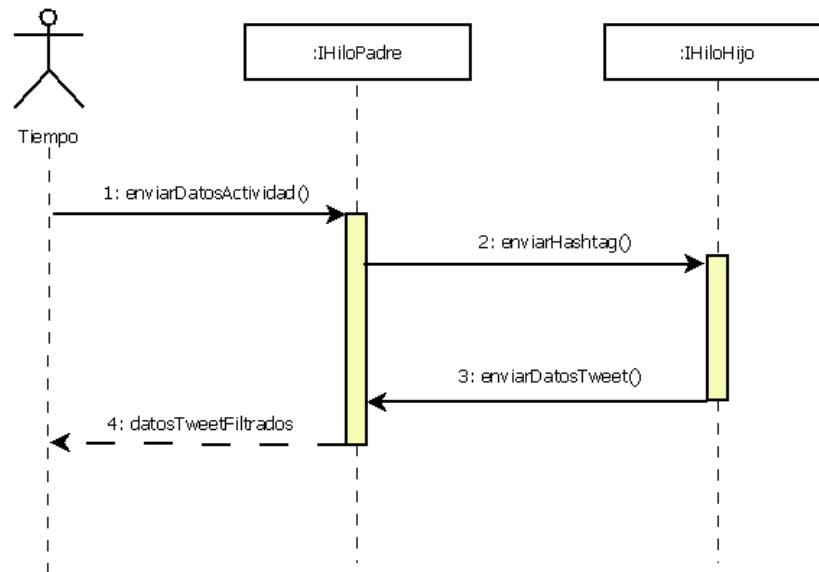


Figura 8.19: Diagrama de secuencia *FormarNuevoTweet*

- InsertarDatos

El diagrama de secuencia que se muestra en la figura 8.20 muestra la interacción de mensajes entre los objetos que participan en el CU *InsertarDatos*. El actor *Tiempo* es el encargado de iniciar la acción mediante el objeto de tipo *hilo* que inicia la conexión con la base de datos e inserta únicamente los datos necesarios del *tweet* en ella. De esta forma queda insertado el nuevo *tweet*, que incluye solo información relevante para el sistema controlador, habiéndose descartado el resto de información no relevante. A continuación se realiza la desconexión de la base de datos.

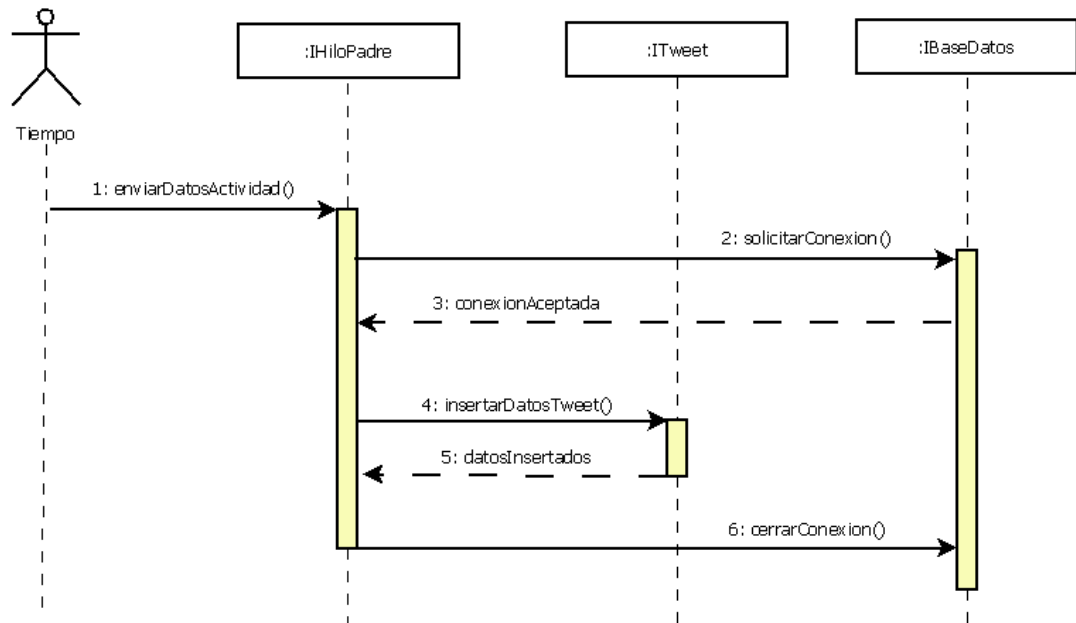


Figura 8.20: Diagrama de secuencia *InsertarDatos*

- SeleccionarGanadores

El diagrama de secuencia que se muestra en la figura 8.21 muestra la interacción de mensajes entre los objetos que participan en el CU *SeleccionarGanadores*. El actor *Tiempo* es el encargado de iniciar la acción de envío de los datos de la actividad al objeto *hilo* para que éste realice la conexión con la base de datos. A continuación se obtienen todos los participantes de la actividad para que el sistema controlador realice la operación de selección de ganadores.

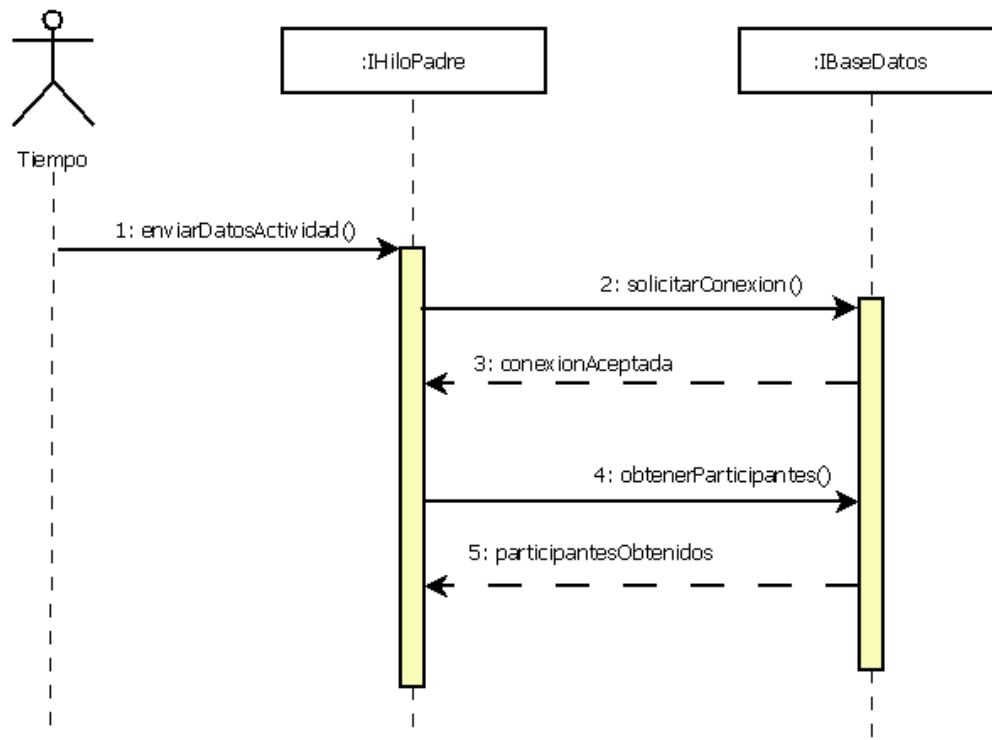


Figura 8.21: Diagrama de secuencia *SelecionarGanadores*

- InsertarGanador

El diagrama de secuencia que se muestra en la figura 8.22 muestra la interacción de mensajes entre los objetos que participan en el CU *InsertarGanador*. El actor *Tiempo* es el encargado de iniciar la acción de procesar los datos de los participantes obtenidos e insertar el resultado en la base de datos, que son los ganadores de la actividad. A continuación el objeto *hilo* se encarga de cerrar la conexión con la base de datos y eliminar finalmente el proceso.

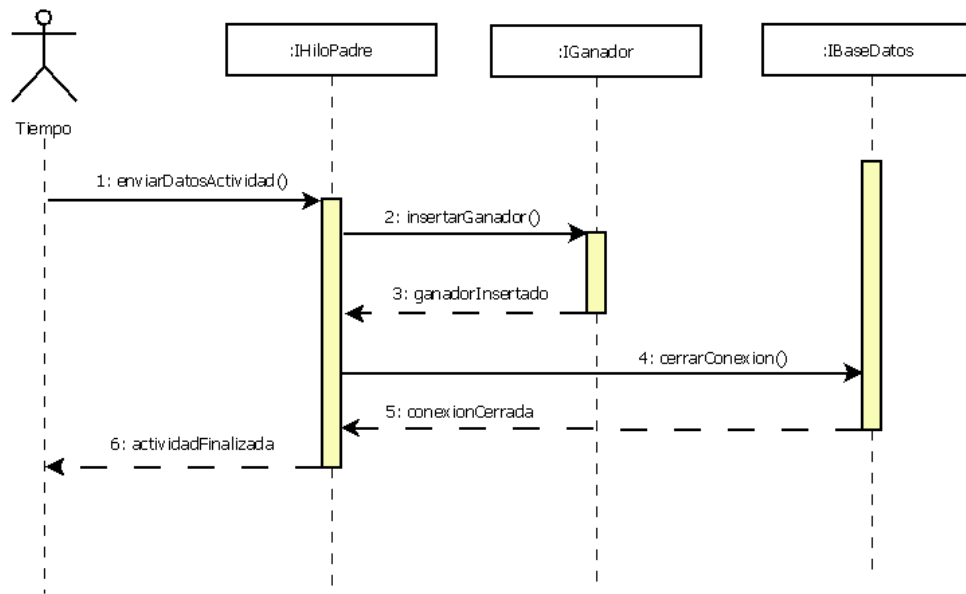


Figura 8.22: Diagrama de secuencia *InsertarGanador*

8.5. DIAGRAMAS DE ESTADO

Un diagrama de estado es una técnica que se utiliza para describir el comportamiento de un sistema. Describen todos los estados posibles en los que puede estar un objeto y la manera en que cambia el estado de dicho objeto, mostrando el flujo de control entre estados. Se consideran útiles para modelar el ciclo de vida de un objeto.

En los siguientes apartados se han realizado los diagramas de estados sólo para aquellos objetos que presentan un comportamiento relativamente interesante como ayuda para una mejor comprensión del subsistema controlador.

8.5.1. SUBSISTEMA CONTROLADOR

El subsistema controlador consta de 2 diagramas de estados, los cuales se detallan a continuación. Son los siguientes:

- Diagrama de estado: ciclo de vida de una actividad
- Diagrama de estado: contabilización de tweets
- Diagrama de estado: gestión de resultados

8.5.1.1. Diagrama de estado: Ciclo de vida de una actividad

En la siguiente figura 8.23 se muestra como una actividad permanece en espera de iniciar su ejecución hasta que el paso de cierto período de tiempo (cierta hora y fecha concreta) inicia la transición de la misma al estado activo. La siguiente transición es causada por un evento de tipo condicional que puede tomar el valor verdadero o falso. En el caso que sea falso, se termina el proceso de la actividad. En el caso que sea verdadero, la actividad comunica con *Twitter*. Se produce una nueva transición en la cual la actividad recoge *tweet* válidos. A continuación se procesa la información de ese *tweet* y almacena información. Este estado permanece hasta que un evento de paso de un período de tiempo inicia una transición de estado final en la que se termina el proceso.

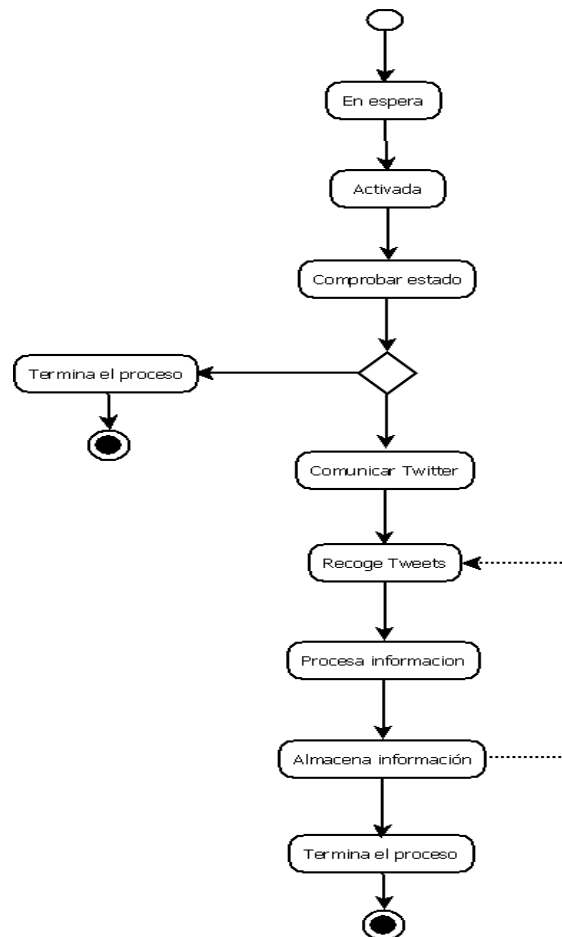


Figura 8.23: Diagrama de estado. *Ciclo de vida de una actividad*

8.5.1.2. Diagrama de estado: Contabilización de tweets

En la siguiente figura 8.24 se muestra como el objeto *tweet* cambia su estado a través de una indicación de otro objeto del modelo. La siguiente transición es causada por un evento de tipo comprobación. A continuación, se produce una transición de tipo condicional que puede tomar el valor verdadero o falso. Si toma el valor falso, se descarta información acerca del objeto y se termina el proceso. Si el valor es verdadero, el objeto cambia de estado por un evento y pasa a revisar la actividad a la que pertenece el objeto. La siguiente transición produce que se inserte la información relevante del objeto en la base de datos, y a continuación, finaliza el proceso.

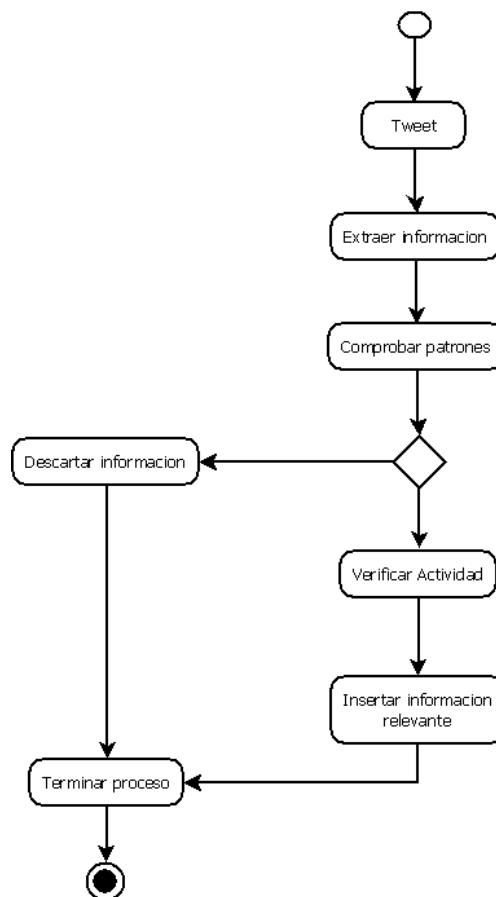


Figura 8.24: Diagrama de estado. *Contabilización de tweets*

8.5.1.3. Diagrama de estado: Gestión de resultados

En la siguiente figura 8.25 se muestra como el objeto se inicia con una conexión a *Twitter*. Mediante un evento cambia su estado y pasa a realizar una comprobación de *hashtag*. La siguiente transición es causada por un evento de tipo condicional que

puede tomar el valor verdadero o falso. En el caso que sea falso, se descarta el *tweet* que contiene dicho *hashtag*. En el caso que sea verdadero, se procesan los datos relativos al *tweet*. El siguiente estado es insertar estos datos en la base de datos y se finaliza dicha inserción. La siguiente transición es causada por un nuevo evento de tipo condicional relacionado con el tipo de actividad que puede tomar el valor verdadero o falso. En el caso que sea falso, se termina el proceso de la actividad. En el caso que sea verdadero, se pasa al estado de seleccionar los ganadores de entre todos los participantes; a continuación se insertan en base de datos y se termina el proceso.

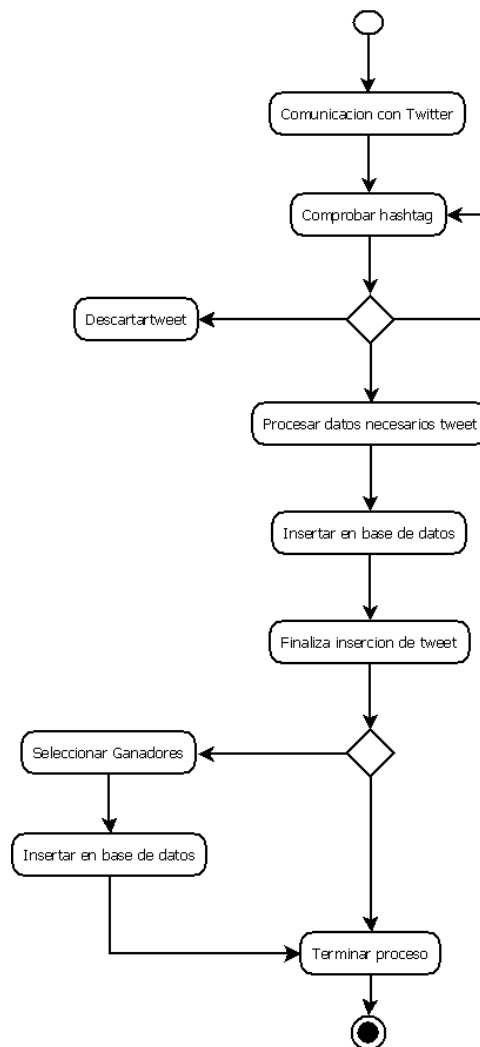


Figura 8.25: Diagrama de secuencia. *Almacenar resultados*

9. ESPECIFICACIÓN DE REQUISITOS DE LA INTERFAZ

9.1. INTRODUCCIÓN

En este capítulo se describirán las características que deberá poseer la interfaz que utilizarán los usuarios del sistema *SIGAPTWIT*. Los requisitos especificados en este análisis darán como resultado el *Diseño de la Interfaz* que se detallará en el capítulo 14.

La interfaz es una de las partes más importantes de un Sistema Informático, ya que es lo que los usuarios ven del mismo y con la cual interactúan, llevando a cabo la relación entre el usuario y el software.

La interfaz debe resultar lo más sencilla y simple posible para facilitar la comodidad del usuario y poder realizar todas las operaciones que desee sin que resulte tedioso ni complicado. También es importante que sea atractiva y no aburra al usuario con el manejo de la misma.

Al tratarse de una aplicación web, la interfaz debe cargarse lo más rápido posible para no desesperar al usuario con largas esperas, evitando presentar contenidos pesados.

La legibilidad, el tamaño y color de las fuentes de texto, también deben ser tomadas en cuenta, para facilitar la lectura de los textos que aparecen en la aplicación y debiendo estar a su vez, bien contrastados con el color de fondo.

Por último, para facilitar al usuario el entendimiento de los diferentes elementos que aparecerán, se designarán a estos con nombres autodescriptivos. Del mismo modo, en los formularios y campos de edición, se indicará de algún modo sugerente, los campos que deben ser obligatoriamente rellenados para poder realizar las operaciones correspondientes.

9.2. CARACTERÍSTICAS DE LA INTERFAZ

Como se ha citado anteriormente, el Sistema *SIGAPTWIT* utiliza como base un sistema web. Es por ello que el diseño de la interfaz estará optimizado y diseñado específicamente para navegadores web.

En el diseño de la página inicial, aparecerán tres opciones bien definidas:

- Acceso Público: para ver los resultados de las actividades sin necesidad de ser un usuario registrado en la aplicación.
- Formulario de acceso: para acceder a la aplicación mediante un formulario que consta de dos campos editables, en el que el usuario deberá introducir sus datos (usuario y contraseña) para posteriormente, poder acceder a una siguiente página que presentará todas las operaciones disponibles a las que un usuario registrado tiene acceso.
- Formulario de registro: formulario que es necesario rellenar con una serie de campos que especifican algunos datos del usuario, y a través del cual una vez creado, permite al usuario tener la cualidad de usuario registrado con las consecuentes utilidades que ello implica.

A continuación se describirán los distintos elementos que podrá encontrarse un usuario navegando a través de *SIGAPTWIT*:

- Menú: este elemento permite al usuario elegir una opción de las que se presentan. Los menús se utilizarán para la navegación a través de las páginas web. Un menú está formado, principalmente, por un conjunto de enlaces web.
- Formulario: es un conjunto de elementos que permiten al usuario, entre otras funciones, introducir datos en el sistema a través del subsistema web. Los elementos que pueden contener los formularios son los siguientes:
 - *Botón*: es el elemento que permite la interacción del usuario con el subsistema web, y que lleva a cabo una acción en el momento en el que el usuario pulsa el botón. Esta acción, por ejemplo, puede ser un enlace a otra página web.
 - *Campo de texto*: Se representará gráficamente en forma de rectángulo. Se usará para que el usuario introduzca en ellos los valores que van a tomar los campos de la base de datos, o para mostrar la información contenida en estos campos para su posterior edición. Los campos que muestren un asterisco significa que deben rellenarse obligatoriamente.
 - *Área de texto*: Este tipo de elemento es similar al anterior, pero con la diferencia de disponer de varias líneas para la introducción de caracteres.
 - *Lista desplegable*: En este tipo de elemento se presentan una serie de opciones en una lista desplegable, y que permite la elección de una de las mismas.
 - *Botones de radio*: Se presenta una lista de opciones junto con un botón redondo con el fondo blanco junto a cada opción. Se permite elegir una única opción de la lista. Al pulsar en el botón de una opción, éste se rellenará con un punto negro. Si se cambia de opción, la opción anteriormente elegida se desmarcará.

- *Botón de formulario:* Una vez el usuario haya rellenado el formulario, pulsará este botón, que efectuará la acción de enviar los datos del formulario. Antes de enviar los datos, el formulario es validado comprobando que los datos introducidos son los permitidos, y que se han cumplimentado también los elementos obligatorios. La etiqueta del botón suele ser *Crear*.

A continuación se muestra una figura donde se expone la estructura que deberá tener la pantalla inicial de la aplicación. Esta figura incluye el título de la aplicación, descripción de sus siglas, un formulario de acceso para usuarios ya registrados, un formulario de registro para nuevos usuarios, y un acceso público que sólo permite mostrar información acerca de las actividades.

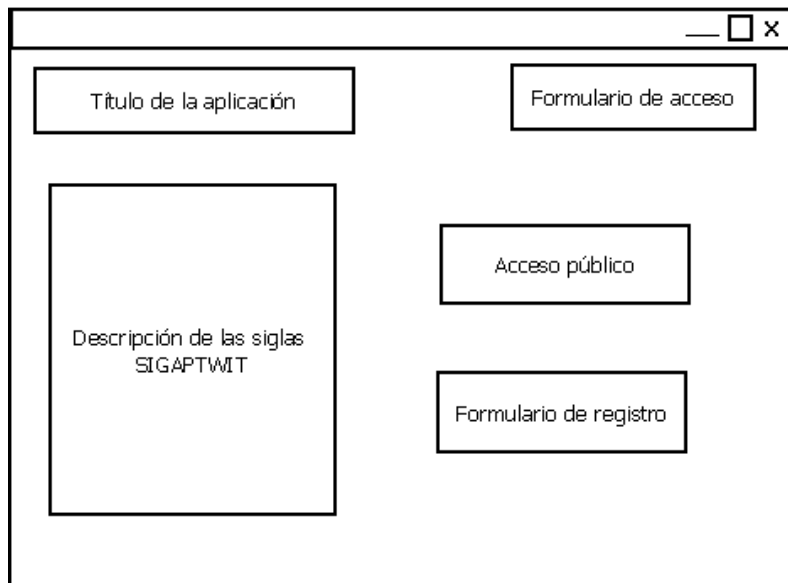


Figura 9.1: Estructura de la ventana inicial

Si se clikea en la ventana de acceso público, aparece una nueva pantalla en la cual se muestra el título de la aplicación, el formulario de acceso, los tres tipos de actividades Votaciones, Sorteos y Respuesta Libre, con sus respectivos menús que son nombrados de la siguiente forma: “Activas” que define aquellas actividades que están ejecutándose en ese mismo momento de consulta y muestra resultados; y “Cerradas”, para mostrar aquellas actividades que ya han finalizado, mostrando sus resultados finales.

En las figura 9.2 y 9.3, se muestra lo anteriormente expuesto citando como ejemplo una actividad de tipo *Respuesta Libre*, pero ocurre exactamente igual para el resto de actividades.

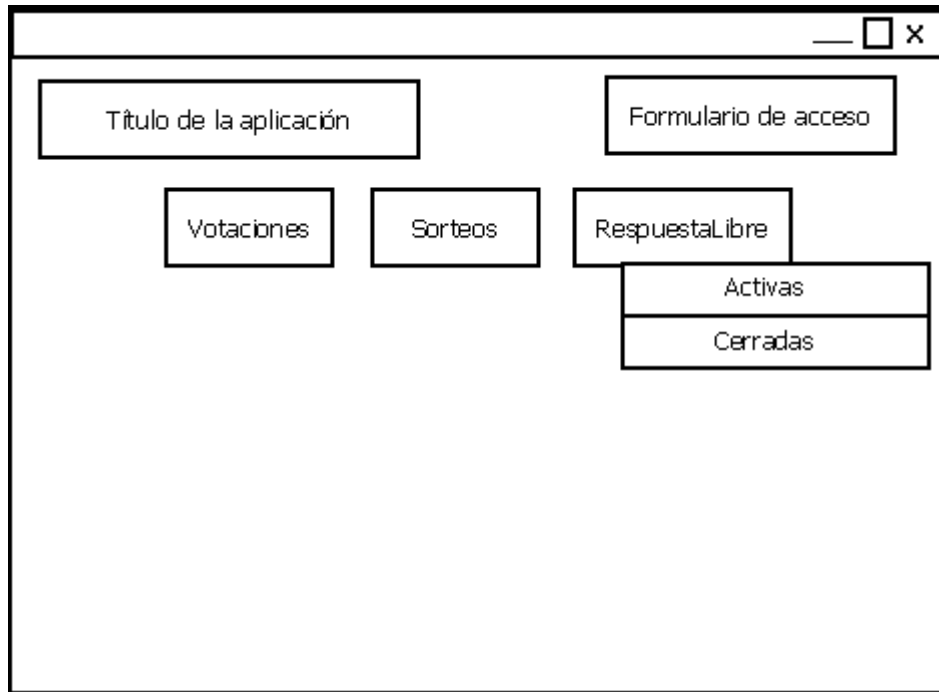


Figura 9.2: Estructura de la ventana de acceso público

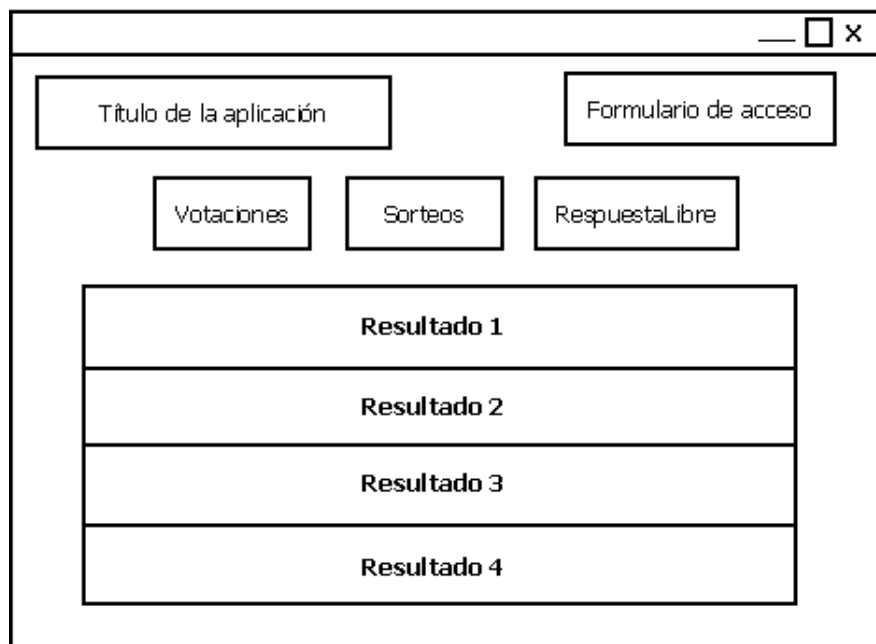


Figura 9.3: Estructura de la ventana de resultados públicos

Si previamente un usuario se ha registrado en la aplicación de forma correcta, y se identifica a través del formulario de acceso, el usuario se encuentra en la parte de acceso privado de la aplicación. En ella aparece una nueva pantalla en la cual se muestra el título de la aplicación, el usuario registrado, un botón para abandonar la ventana de acceso privado, un botón para poder darse de baja de la aplicación el propio usuario, y los tres tipos de actividades *Votaciones*, *Sorteos* y *Preguntas / Respuestas libres*, con sus respectivos menús.

El menú de cada actividad presenta las opciones “Nuevo ...” (y a continuación la actividad: “Nueva votación” o “Nuevo Sorteo” o “Nueva Pregunta”) y “Ver actividades”. Esta última opción, incluye un submenú con las siguientes alternativas: “En espera”, para aquellas actividades que están preparadas para ejecutarse; “En ejecución” que define aquellas actividades que están ejecutándose en ese mismo momento de consulta; y “Cerradas”, para mostrar aquellas actividades que ya han finalizado. En la siguiente figura se muestra de forma visual lo expuesto en el párrafo anterior.

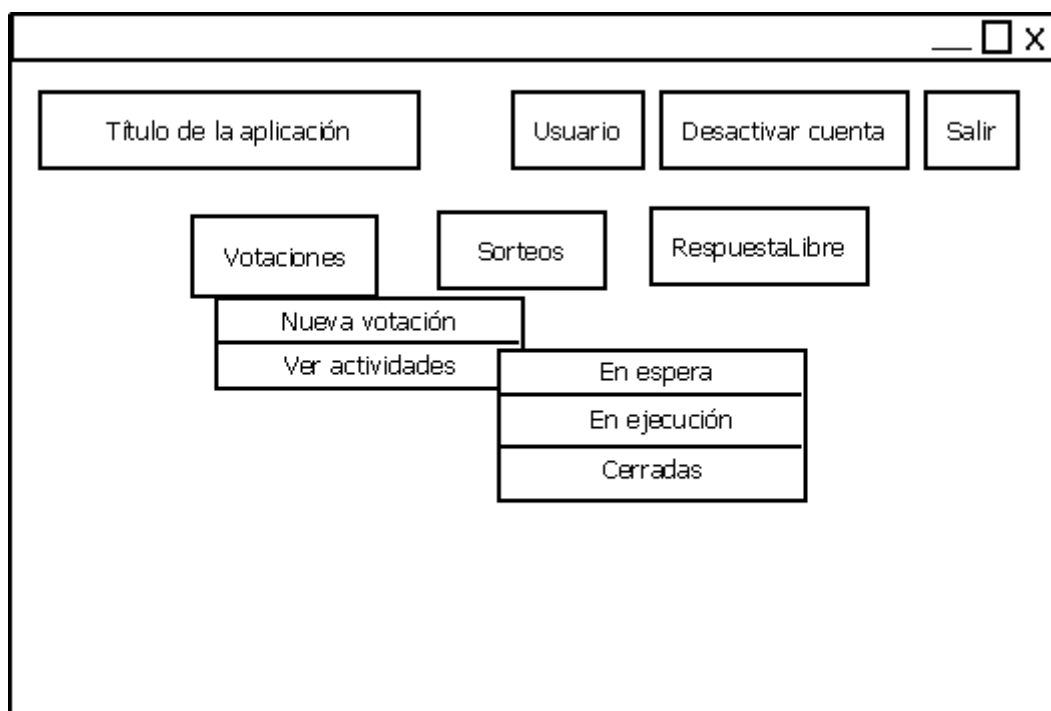


Figura 9.4: Estructura de la ventana de actividades privadas

En la ventana de acceso privado, si se hace clic sobre *Votaciones*, se tiene acceso a dos opciones como se ha comentado anteriormente. Para la opción “Nueva Votacion”, aparece un formulario de registro para la actividad Votación, con sus respectivos campos y áreas de texto, y sus mensajes de error correspondientes, en el caso de que el usuario cometiera algún error en el registro. En la figura 9.5 se muestra lo que se ha descrito de forma visual.

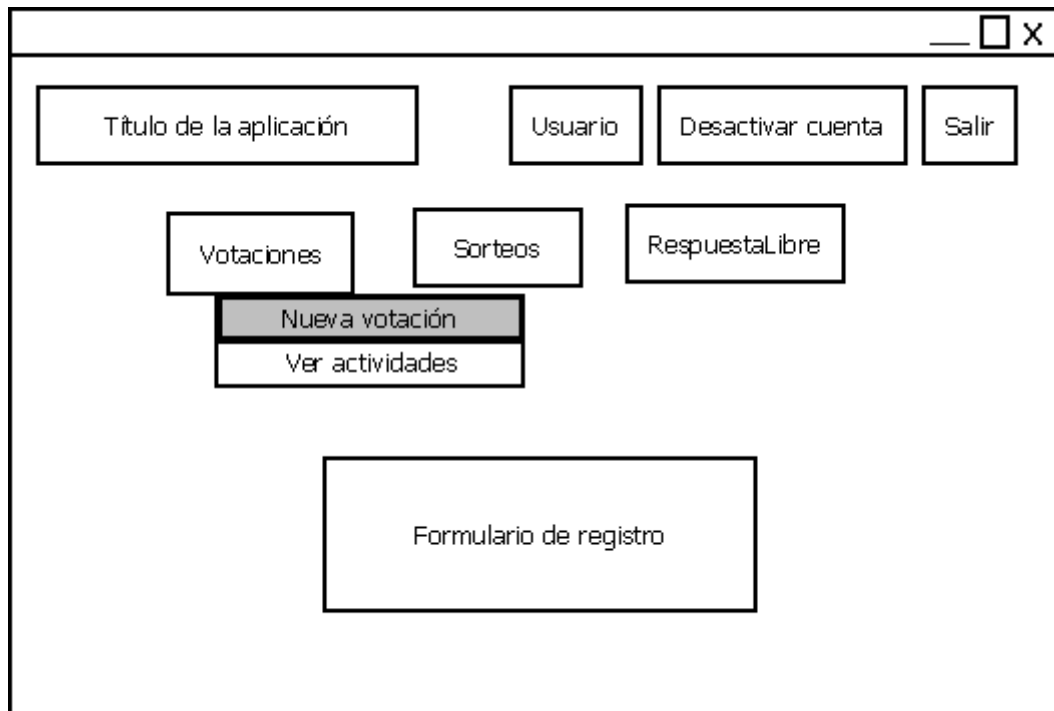


Figura 9.5: Estructura de la ventana para crear nueva actividad

En las figuras siguientes, se muestra el submenú de un tipo de actividad, que incluye como ya se ha comentado anteriormente, las alternativas “En espera”, “En ejecución” y “Cerradas”. Estas opciones son exactamente iguales para el resto de actividades, aunque en las figuras se ha optado por elegir como ejemplo una actividad de tipo *Votación*.

- Si la opción elegida es “En espera” (Figura 9.6): se listan en una ventana todas aquellas actividades del usuario que aún no han iniciado su ejecución, con sus respectivas características. El usuario registrado tiene la posibilidad de eliminar las actividades creadas por él que permanezcan en este estado y en este listado.

- Si la opción elegida es “En ejecución” (Figura 9.7): se listan en una ventana todas aquellas actividades del usuario que están ejecutándose en ese momento, con sus respectivas características. El usuario registrado, al igual que el usuario de acceso público, tiene la posibilidad de visualizar los resultados de una actividad que aparezca en el listado, en ese mismo instante.
- Si la opción elegida es “Cerradas” (Figura 9.8): se listan en una ventana todas aquellas actividades del usuario que ya han finalizado, con sus respectivas características. El usuario registrado tiene la opción de eliminar aquellas actividades creadas por él que ya hayan finalizado. Dicho usuario, desde esta ventana también tiene la posibilidad de visualizar los resultados finales de una actividad.

Por último, existirán varios tipos de mensajes que se le presentarán al usuario al realizar una operación:

- Mensaje informativo: este tipo de mensaje se presentará cuando una operación haya sido realizada con éxito.
- Mensaje de error: se mostrará cuando una operación no se haya podido realizar por algún motivo, por alguna acción incorrecta por parte del usuario (datos introducidos equivocadamente).

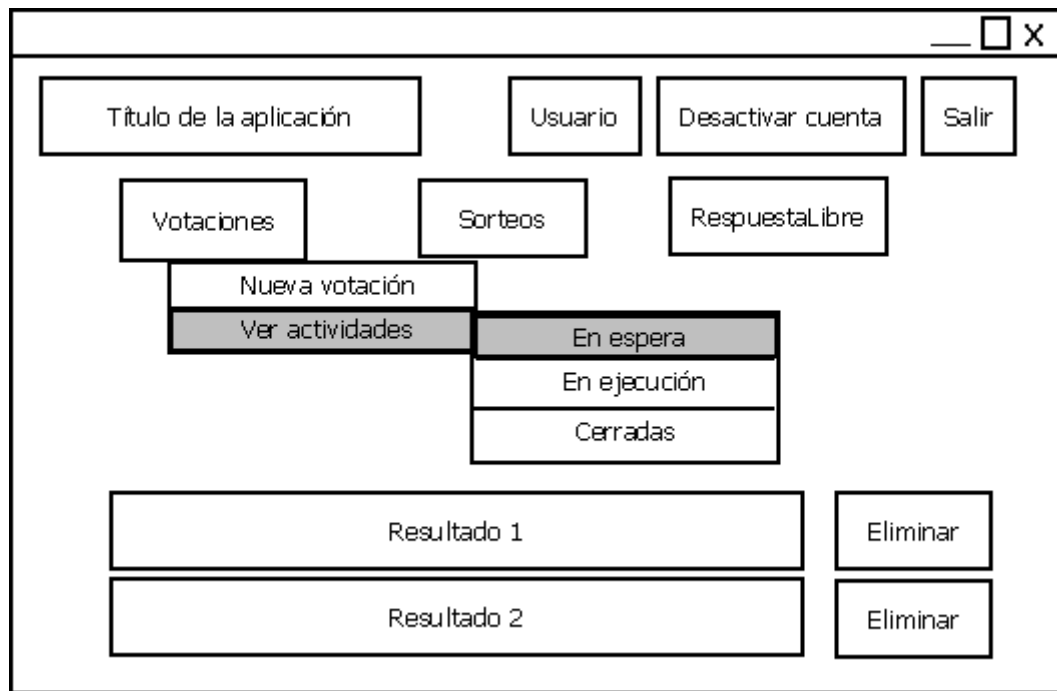


Figura 9.6: Estructura de la ventana para mostrar resultados de actividades en espera

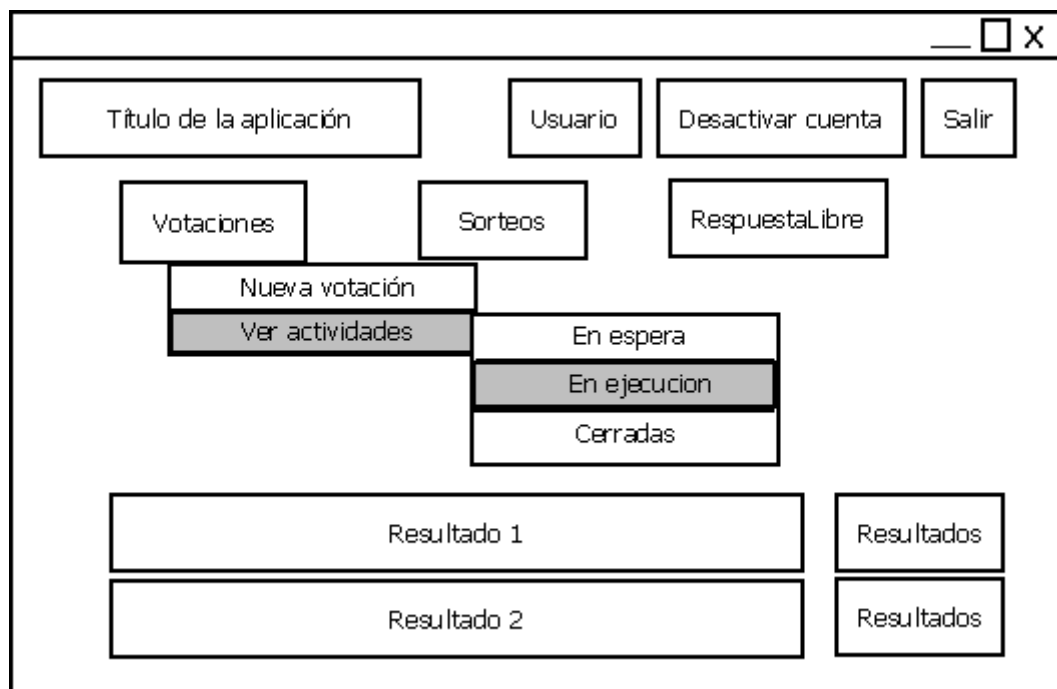


Figura 9.7: Estructura de la ventana para mostrar resultados de actividades en ejecución

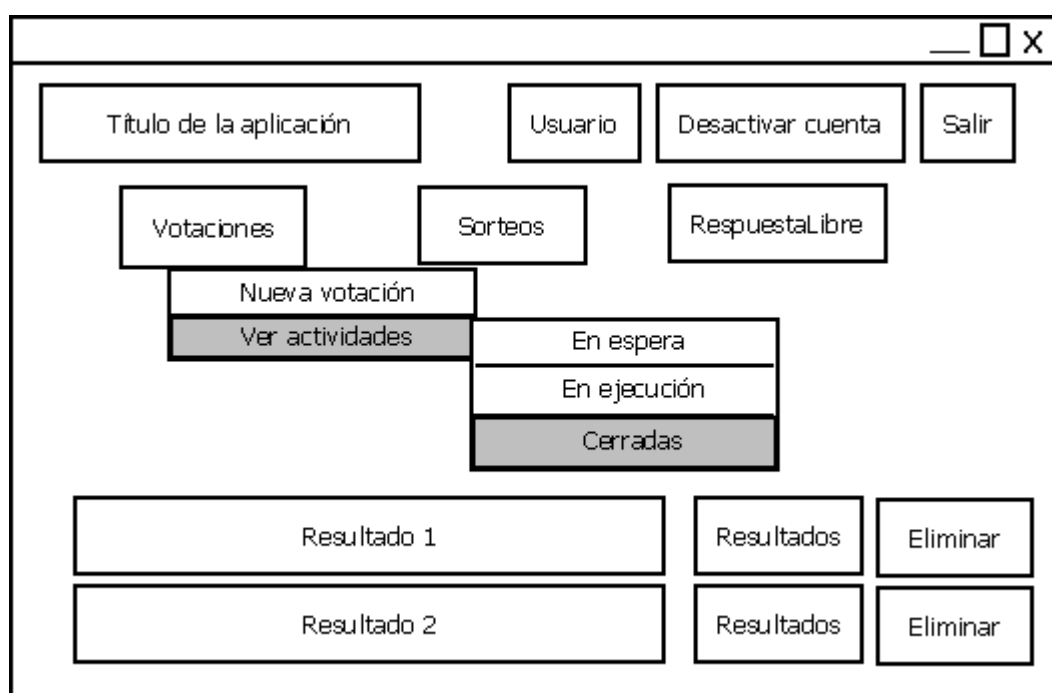


Figura 9.8: Estructura de la ventana para mostrar resultados de actividades cerradas

10. DISEÑO DE DATOS

10.1. INTRODUCCIÓN

El diseño de datos es de gran importancia en aplicaciones que hacen un uso masivo de la información. Este es el caso del proyecto *SIGAPTWIT*, ya que una de sus características más importantes es el almacenamiento de datos y el tratamiento de los mismos. Por ello, es necesario un buen diseño de datos, que evite la redundancia en la información y que tenga una gran eficacia en el manejo de los mismos, utilizando a su vez la menor cantidad de recursos posibles.

A continuación se explicará el proceso a seguir en el *Diseño de datos*.

En primer lugar, se realizará la transformación del *Modelo Conceptual* de la base de datos, descrita en el *capítulo 7 Modelo de datos*, al *Modelo Relacional* en el que las entidades y sus interrelaciones se transformarán en tablas.

Posteriormente se procederá a normalizar el *Modelo Relacional* mediante una serie de normas llamadas *Reglas de normalización* (15). El resultado de esta normalización satisfará la forma normal de *Boyce-Codd* que, por regla general, asegura que el esquema resultante es consistente.

A continuación, se realizará una optimización del *Modelo Relacional*, y en último lugar, se efectuará una transformación del *Modelo Relacional* final a la definición sintáctica de las tablas (16).

10.2. MODELO RELACIONAL

La transformación del *Modelo Conceptual* al *Modelo Relacional* se realizará a través de la aplicación de las reglas de transformación (*RTECAR*).

Para cada tabla creada se especificará:

- Nombre de la tabla.
- Nombre de la/s entidad/es e interrelación/es de la/s que proviene.
- Atributos que componen la tabla:
 - Clave principal.
 - CLAVE ALTERNA.
 - **Clave foránea** (17).
 - Resto de atributos.

A continuación se pasará a detallar las distintas tablas del *Modelo Relacional*.

10.2.1. TABLA USUARIOAPLICACION

El tipo de entidad *UsuarioAplicacion* se transformará en la tabla *UsuarioAplicacion*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

- UsuarioAplicacion (id-usuario, nombre, apellidos, codigo_postal, usuario, contrasenya, email)

10.2.2. TABLA TWEET

El tipo de entidad *Tweet* se transformará en la tabla *Tweet*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

Como se indica en el apartado 10.2.3 *Tabla Actividad*, en la interrelación *Votacion – Tweet* del tipo 1:N, la entidad *Tweet* que interviene con cardinalidad

máxima muchos, participa de forma parcial, por lo tanto, y por aplicación de la regla RTECAR-3.1, el identificador del tipo de entidad que participa con cardinalidad máxima uno (*Votacion*) pasará como *clave foránea* al tipo de entidad que participa con cardinalidad máxima muchos (*Tweet*). Este atributo se llamará *id_votacion*.

También indicado en el apartado 10.2.3 *Tabla Actividad*, en la interrelación *Sorteo – Tweet* del tipo 1:N, la entidad *Tweet* interviene con cardinalidad máxima muchos y participa de forma parcial, por lo tanto, y por aplicación de la regla RTECAR-3.1, el identificador del tipo de entidad que participa con cardinalidad máxima uno (*Sorteo*) pasará como *clave foránea* al tipo de entidad que participa con cardinalidad máxima muchos (*Tweet*). Este atributo se llamará *id_sorteo*.

Una vez más, como se ha indicado en el apartado 10.2.3 *Tabla Actividad*, en la interrelación *RespuestaLibre – Tweet* del tipo 1:N, la entidad *Tweet* que interviene con cardinalidad máxima muchos, participa de forma parcial, por lo tanto, y por aplicación de la regla RTECAR-3.1, el identificador del tipo de entidad que participa con cardinalidad máxima uno (*RespuestaLibre*) pasará como *clave foránea* al tipo de entidad que participa con cardinalidad máxima muchos (*Tweet*). Este atributo se llamará *id_respuestalibre*.

- *Tweet* (*id-tweet*, *id_usuario*, *contenido*, *fecha_hora_entrada*, *geolocalizacion*, *id_sorteo*, *id_respuestalibre*, *id_votacion*)

10.2.3. TABLA ACTIVIDAD

La tabla *Actividad* se forma a partir del tipo de entidad del mismo nombre tomando los atributos de este tipo de entidad, por aplicación de la regla RTECAR-1.

- *Actividad* (*id-actividad*, *titulo*, *descripcion*, *fecha_inicio*, *fecha_fin*, *cuestion*, *estado*, *publico*)

Es conveniente y necesaria la eliminación de las relaciones jerárquicas como paso previo al proceso de transformación de los esquemas conceptuales a relacionales. En nuestro caso, por aplicación de la regla PRTECAR-3, se elimina el supertipo de entidad. De esta forma se transfieren todos los atributos del supertipo a cada uno de los subtipos, y cada uno de los tipos de interrelación que mantiene el

supertipo de entidad serán considerados para cada uno de los subtipos, además manteniéndose los tipos de interrelación en los que intervengan cada uno de los subtipos de entidad. El atributo cualificador *tipo_actividad*, será desestimado. Como el tipo de interrelación jerárquica es exclusivo, los subtipos intervendrán de forma parcial en los tipos de interrelación transferidos desde el supertipo.

Como aplicación de lo anteriormente expuesto, la tabla *Actividad* es eliminada, transfiriendo sus atributos al resto de subtipos de entidad, a partir de los cuales se formarán sus correspondientes tablas.

10.2.4. TABLA VOTACION

El tipo de entidad *Votacion* se transformará en la tabla *Votacion*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

Como se ha indicado en el apartado 10.2.3 *Tabla Actividad*, la interrelación jerárquica pasará a ser una relación 1:N entre las entidades *UsuarioAplicacion* y *Votacion* (RTECAR-3.1). En la transformación de este tipo de interrelación, el identificador de la entidad que participa de forma completa (*UsuarioAplicacion*) será una *clave foránea* en la entidad que participa con cardinalidad máxima muchos (*Votacion*).

- *Votacion* (id-votacion, titulo, descripcion, fecha-inicio, fecha-fin, cuestion, estado, publico, id_usuario)

10.2.5. TABLA OPCION

El tipo de entidad *Opcion* se transformará en la tabla *Opcion*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

En la interrelación *Opcion – Votacion* del tipo 1:N, ambas entidades participan de forma total, por lo tanto, y por aplicación de la regla RTECAR-3.1, el identificador del tipo de entidad que participa con cardinalidad máxima uno (*Votacion*) pasará como *clave foránea* al tipo de entidad que participa con cardinalidad máxima muchos (*Opcion*). Este atributo se llamará *id_votacion*.

- Opcion (id-opcion, hashtag, **id_votacion**)

10.2.6. TABLA SORTEO

El tipo de entidad *Sorteo* se transformará en la tabla *Sorteo*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

Como se ha indicado en el apartado 10.2.3 *Tabla Actividad*, la interrelación jerárquica pasará a ser una relación 1:N entre las entidades *UsuarioAplicacion* y *Sorteo* (RTECAR-3.1). En la transformación de este tipo de interrelación, el identificador de la entidad que participa de forma completa (*UsuarioAplicacion*) será una *clave foránea* en la entidad que participa con cardinalidad máxima muchos (*Sorteo*).

- Sorteo (id-sorteo, titulo, descripcion, fecha-inicio, fecha-fin, cuestion, estado, publico, hashtag, aleatorio, num-ganadores, **id_usuario**)

10.2.7. TABLA GANADOR

El tipo de entidad *Ganador* se transformará en la tabla *Ganador*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

En la interrelación *Sorteo – Ganador* del tipo 1:N, la entidad *Ganador* participa de forma parcial, por lo tanto, y por aplicación de la regla RTECAR-3.1, el identificador del tipo de entidad que participa con cardinalidad máxima uno (*Sorteo*) pasará como *clave foránea* al tipo de entidad que participa con cardinalidad máxima muchos (*Ganador*). Este atributo se llamará *id_sorteo*.

En la interrelación *Tweet – Ganador* del tipo 1:1, la entidad *Tweet* participa de forma completa en el tipo de interrelación, y la entidad *Ganador* participa de forma parcial, por ello, por aplicación de la regla RTECAR-2.2, el identificador del tipo de entidad que participa de forma total (*Tweet*), pasa como *clave alterna y foránea* al otro tipo de entidad (*Ganador*). Este atributo se llamará *id_tweet*.

- Ganador (id-ganador, **id_sorteo**, **ID_TWEET**)

10.2.8. TABLA RESPUESTALIBRE

El tipo de entidad *RespuestaLibre* se transformará en la tabla *RespuestaLibre*, manteniendo el número y tipo de los atributos presentes en la entidad (regla RTECAR-1).

Como se ha indicado en el apartado 10.2.3 *Tabla Actividad*, la interrelación jerárquica pasará a ser una relación 1:N entre las entidades *UsuarioAplicacion* y *RespuestaLibre* (RTECAR-3.1). En la transformación de este tipo de interrelación, el identificador de la entidad que participa de forma completa (*UsuarioAplicacion*) será una *clave foránea* en la entidad que participa con cardinalidad máxima muchos (*RespuestaLibre*).

- RespuestaLibre (id-rl, titulo, descripcion, fecha-inicio, fecha-fin, cuestion, estado, publico, hashtag, id_usuario)

10.3. NORMALIZACIÓN DEL MODELO RELACIONAL

A continuación se normalizará el *Modelo Relacional* mediante la aplicación de las *reglas de normalización*. Estas reglas persiguen los siguientes objetivos: la eliminación de redundancias superfluas; aumentar el desempeño de las operaciones de actualización de la base de datos; representar de forma coherente los objetos y relaciones; y garantizar la fiabilidad de las interrogaciones sobre la información mantenida en la base de datos.

El modelo se normalizará hasta satisfacer la *Formal Normal* de *Boyce-Codd* (FNBC) (18), ya que se asegura que el modelo poseerá los objetivos que se pretenden conseguir con esta normalización.

Para cumplir con la FNBC, todos los dominios subyacentes de la relación deben contener valores atómicos, además cada *determinante funcional* debe ser una clave candidata de la relación.

Como ninguno de los atributos de las distintas relaciones es compuesto, se parte con la primera restricción cumplida.

10.3.1. TABLA USUARIOAPLICACION

Las *dependencias funcionales* (19) que existen están formadas entre los atributos que no forman parte de la clave principal, y el atributo que forma la clave principal, es decir, entre los atributos no primos y los primos. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

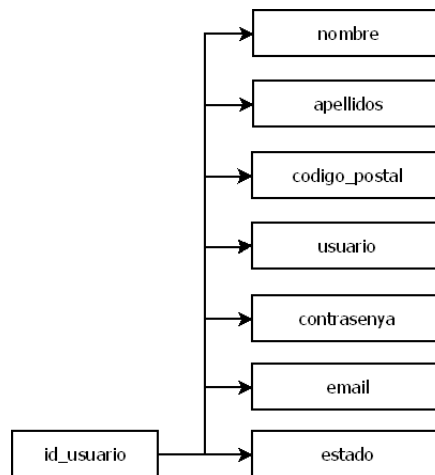


Figura 10.1: Dependencias funcionales de la tabla *UsuarioAplicacion*

10.3.2. TABLA TWEET

Las *dependencias funcionales* que existen están formadas entre los atributos que no forman parte de la clave principal, y el atributo que forma la clave principal, es decir, entre los atributos no primos y los primos. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

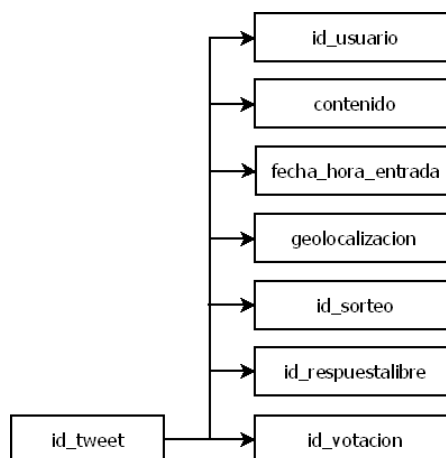


Figura 10.2: Dependencias funcionales de la tabla *Tweet*

10.3.3. TABLA VOTACION

Las *dependencias funcionales* que existen están formadas entre los atributos que no forman parte de la clave principal, y el atributo que forma la clave principal, es decir, entre los atributos no primos y los primos. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

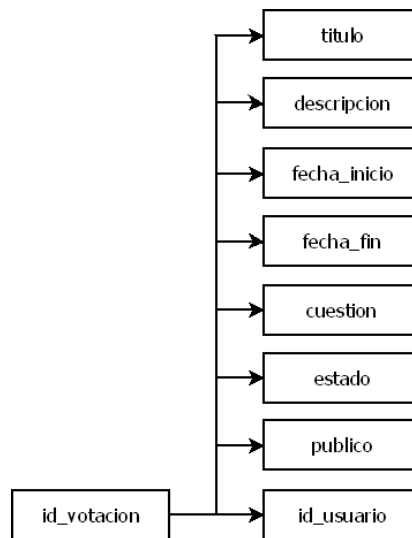


Figura 10.3: Dependencias funcionales de la tabla *Votacion*

10.3.4. TABLA OPCION

Las *dependencias funcionales* que existen están formadas entre los atributos que no forman parte de la clave principal, y el atributo que forma la clave principal, es decir, entre los atributos no primos y los primos. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

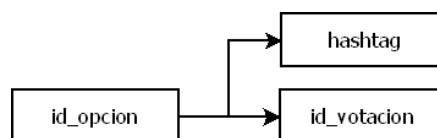


Figura 10.4: Dependencias funcionales de la tabla *Opcion*

10.3.5. TABLA SORTEO

Las *dependencias funcionales* que existen están formadas entre los atributos que no forman parte de la clave principal, y el atributo que forma la clave

principal, es decir, entre los atributos no primos y los primos. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

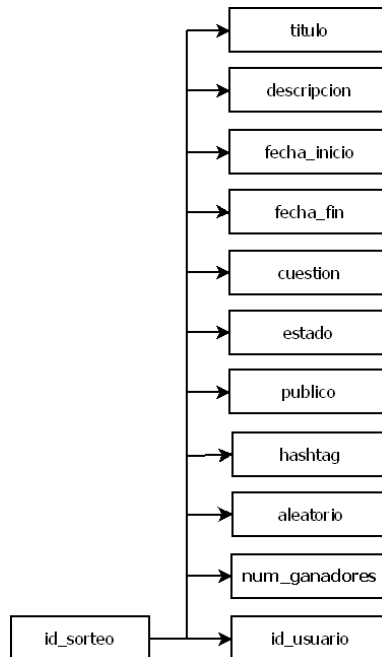


Figura 10.5: Dependencias funcionales de la Tabla Sorteo

10.3.6. TABLA GANADOR

En este caso, los determinantes funcionales son claves candidatas (clave principal y clave alterna) de la relación. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

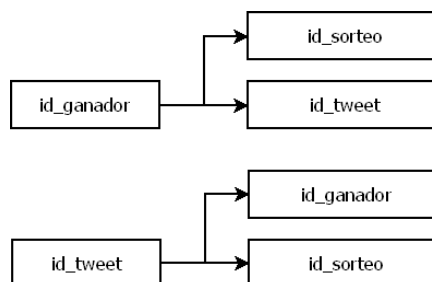


Figura 10.6: Dependencias funcionales de la tabla *Ganador*

10.3.7. TABLA RESPUESTALIBRE

Las *dependencias funcionales* que existen están formadas entre los atributos que no forman parte de la clave principal, y el atributo que forma la clave

principal, es decir, entre los atributos no primos y los primos. Por lo tanto, se cumple la *Forma Normal de Boyce-Codd*.

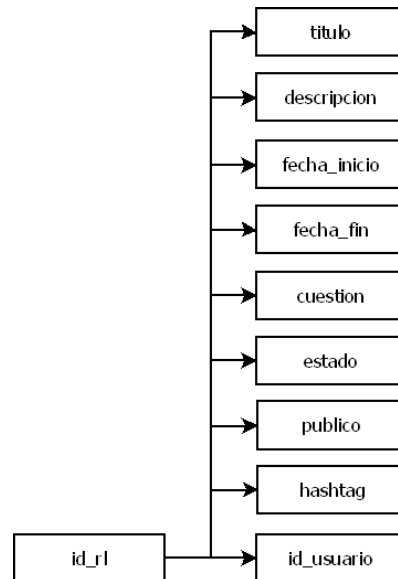


Figura 10.7: Dependencias Funcionales de la tabla *RespuestaLibre*

10.4. ESQUEMA RELACIONAL

El Esquema Relacional resultante quedaría de la forma siguiente:

- UsuarioAplicacion (id_usuario, nombre, apellidos, codigo_postal, usuario, contrasenya, email)
- Votacion (id-votacion, titulo, descripcion, fecha-inicio, fecha-fin, cuestion, estado, publico, **id_usuario**)
- Sorteo (id-sorteo, titulo, descripcion, fecha-inicio, fecha-fin, cuestion, estado, publico, hashtag, aleatorio, num-ganadores, **id_usuario**)
- RespuestaLibre (id-rl, titulo, descripcion, fecha-inicio, fecha-fin, cuestion, estado, publico, hashtag, **id_usuario**)
- Opcion (id-opcion, hashtag, **id_votacion**)
- Tweet (id-tweet, id-usuario, contenido, fecha-hora-entrada, geolocalizacion, **id_sorteo**, **id_respuestalibre**, **id_votacion**)
- Ganador (id-ganador, **id_sorteo**, **ID_TWEET**)

10.5. DIAGRAMA RELACIONAL

En este apartado se muestra el Diagrama Relacional, cuya leyenda es también incluida para una mejor inteligibilidad.

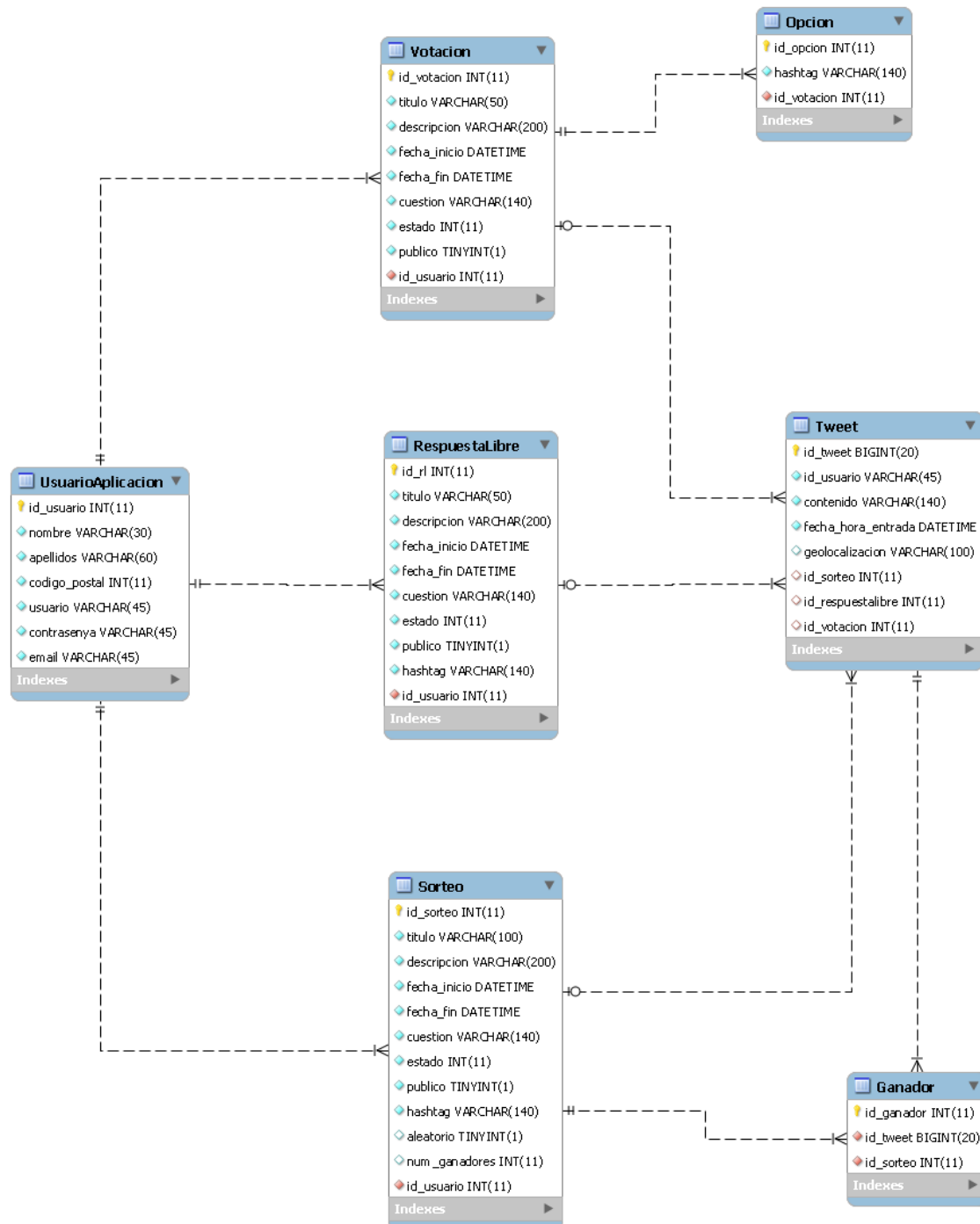


Figura 10.8: Diagrama Relacional

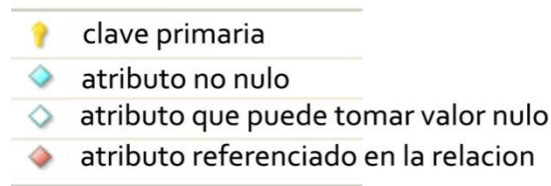


Figura 10.9: Leyenda del Diagrama Relacional

10.6. DEFINICIÓN SINTÁCTICA DE LAS TABLAS

A continuación se presenta el conjunto de sentencias que crearán la estructura de tablas que definirán a la base de datos del sistema. Esto es llamado *Definición sintáctica de las tablas* (20) .

```
CREATE SCHEMA `dbname` DEFAULT CHARACTER SET latin1 ;
```

/ -- Inicialmente, se borran las tablas por si éstas han sido creadas con anterioridad. Las tablas deben ser borradas teniendo en cuenta las relaciones entre ellas. Como norma general es conveniente borrarlas en el orden inverso a como se crean. --*/*

```
DROP TABLE `Ganador`;
DROP TABLE `Tweet`;
DROP TABLE `Opcion`;
DROP TABLE `Votacion`;
DROP TABLE `RespuestaLibre`;
DROP TABLE `Sorteo`;
DROP TABLE `UsuarioAplicacion`;
```

```
START TRANSACTION;
```

```
USE bd_sigaptwit;
```

/ -----Tabla UsuarioAplicacion -----*/*

```
CREATE TABLE IF NOT EXISTS `UsuarioAplicacion` (
  `id_usuario` INT(11) NOT NULL AUTO_INCREMENT,
  `nombre` VARCHAR(30) NOT NULL,
  `apellidos` VARCHAR(60) NOT NULL,
  `codigo_postal` INT(11) DEFAULT NULL,
  `usuario` VARCHAR(45) NOT NULL,
  `contrasena` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
```

```
PRIMARY KEY (`id_usuario`));
```

```
/* -----Tabla Sorteo -----*/
```

```
CREATE TABLE IF NOT EXISTS `Sorteo` (
  `id_sorteo` INT(11) NOT NULL AUTO_INCREMENT,
  `titulo` VARCHAR(50) NOT NULL,
  `descripcion` VARCHAR(200) DEFAULT NULL,
  `fecha_inicio` DATETIME NOT NULL,
  `fecha_fin` DATETIME NOT NULL,
  `cuestion` VARCHAR(140) NOT NULL,
  `estado` INT(11) NOT NULL,
  `publico` TINYINT(1) NOT NULL,
  `hashtag` VARCHAR(140) NOT NULL,
  `aleatorio` TINYINT(1) NULL DEFAULT NULL,
  `num_ganadores` INT(11) NULL DEFAULT NULL,
  `id_usuario` INT(11) NOT NULL,
  PRIMARY KEY (`id_sorteo`),
  INDEX `fk_idusuariosorteo_idusuario_idx` (`id_usuario` ASC),
  CONSTRAINT `fk_idusuariosorteo_idusuario`
    FOREIGN KEY (`id_usuario`)
    REFERENCES `UsuarioAplicacion` (`id_usuario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

```
/* -----Tabla RespuestaLibre -----*/
```

```
CREATE TABLE IF NOT EXISTS `RespuestaLibre` (
  `id_rl` INT(11) NOT NULL AUTO_INCREMENT,
  `titulo` VARCHAR(50) NOT NULL,
  `descripcion` VARCHAR(200) DEFAULT NULL,
  `fecha_inicio` DATETIME NOT NULL,
  `fecha_fin` DATETIME NOT NULL,
  `cuestion` VARCHAR(140) NOT NULL,
  `estado` INT(11) NOT NULL,
  `publico` TINYINT(1) NOT NULL,
  `hashtag` VARCHAR(140) NOT NULL,
  `id_usuario` INT(11) NOT NULL,
  PRIMARY KEY (`id_rl`),
  INDEX `fk_idusuariorl_idusuario_idx` (`id_usuario` ASC),
  CONSTRAINT `fk_idusuariorl_idusuario`
    FOREIGN KEY (`id_usuario`)
    REFERENCES `UsuarioAplicacion` (`id_usuario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

```
/* -----Tabla Votacion -----*/
```

```
CREATE TABLE IF NOT EXISTS `Votacion` (
```

```

`id_votacion` INT(11) NOT NULL AUTO_INCREMENT,
`titulo` VARCHAR(50) NOT NULL,
`descripcion` VARCHAR(200) DEFAULT NULL,
`fecha_inicio` DATETIME NOT NULL,
`fecha_fin` DATETIME NOT NULL,
`cuestion` VARCHAR(140) NOT NULL,
`estado` INT(11) NOT NULL,
`publico` TINYINT(1) NOT NULL,
`id_usuario` INT(11) NOT NULL,
PRIMARY KEY (`id_votacion`),
INDEX `fk_idusuariovotacion_idusuario_idx` (`id_usuario` ASC),
CONSTRAINT `fk_idusuariovotacion_idusuario`
    FOREIGN KEY (`id_usuario`)
    REFERENCES `UsuarioAplicacion` (`id_usuario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);

```

/ -----Tabla Opcion -----*/*

```

CREATE TABLE IF NOT EXISTS `Opcion` (
    `id_opcion` INT(11) NOT NULL AUTO_INCREMENT,
    `hashtag` VARCHAR(140) NOT NULL,
    `id_votacion` INT(11) NOT NULL,
    PRIMARY KEY (`id_opcion`),
    INDEX `fk_idvotacionopcion_idvotacion_idx` (`id_votacion`
ASC),
    CONSTRAINT `fk_idvotacionopcion_idvotacion`
        FOREIGN KEY (`id_votacion`)
        REFERENCES `Votacion` (`id_votacion`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION);

```

/ -----Tabla Tweet -----*/*

```

CREATE TABLE `Tweet` (
    `id_tweet` BIGINT(20) NOT NULL,
    `id_usuario` VARCHAR(45) NOT NULL,
    `contenido` VARCHAR(140) NOT NULL,
    `fecha_hora_entrada` DATETIME NOT NULL,
    `geolocalizacion` VARCHAR(100) NULL,
    `id_sorteo` INT(11) NULL,
    `id_respuestalibre` INT(11) NULL,
    `id_votacion` INT(11) NULL,
    PRIMARY KEY (`id_tweet`),
    INDEX `fk_idsorteotweet_idsorteo_idx` (`id_sorteo` ASC),
    INDEX `fk_idrltweet_idrl_idx` (`id_respuestalibre` ASC),
    INDEX `fk_idvotaciontweet_idvotacion_idx` (`id_votacion` ASC),
    CONSTRAINT `fk_idsorteotweet_idsorteo`

```

```

        FOREIGN KEY (`id_sorteo`)
        REFERENCES `Sorteo` (`id_sorteo`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,
    CONSTRAINT `fk_idrltweet_idrl`
        FOREIGN KEY (`id_respuestalibre`)
        REFERENCES `RespuestaLibre` (`id_rl`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,
    CONSTRAINT `fk_idvotaciontweet_idvotacion`
        FOREIGN KEY (`id_votacion`)
        REFERENCES `Votacion` (`id_votacion`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION);

/* -----Tabla Ganador -----*/

CREATE TABLE IF NOT EXISTS `Ganador` (
    `id_ganador` INT(11) NOT NULL AUTO_INCREMENT,
    `id_tweet` BIGINT(20) NOT NULL,
    `id_sorteo` INT(11) NOT NULL,
    PRIMARY KEY (`id_ganador`),
    INDEX `fk_idtweetganador_idtweet_idx` (`id_tweet` ASC),
    INDEX `fk_idsorteoganador_idsorteo_idx` (`id_sorteo` ASC),
    CONSTRAINT `fk_idtweetganador_idtweet`
        FOREIGN KEY (`id_tweet`)
        REFERENCES `Tweet` (`id_tweet`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,
    CONSTRAINT `fk_idsorteoganador_idsorteo`
        FOREIGN KEY (`id_sorteo`)
        REFERENCES `Sorteo` (`id_sorteo`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION);

COMMIT;

```


11. DISEÑO DE PAQUETES

11.1. INTRODUCCIÓN

Modelar sistemas software conlleva manejar una cantidad considerable de elementos de modelado (clases, interfaces, componentes, nodos, relaciones y diagramas). A partir de un cierto tamaño, es necesario agrupar aquellos elementos relacionados entre sí de acuerdo a algún criterio.

En *UML* (21) las abstracciones que permiten organizar un modelo se llaman paquetes. Los diagramas de paquetes *UML* se suelen usar para mostrar la organización de alto nivel de un proyecto de software. Los paquetes son muy prácticos a la hora de recopilar clases relacionadas para que se les pueda hacer referencia de forma más concisa en diagramas de vista de nivel superior de la arquitectura del proyecto. En un proyecto de grandes dimensiones, los paquetes son una manera eficaz de organizar y documentar subproyectos reutilizables. También es común usar los paquetes para representar otros sistemas y subsistemas que interactúan con el proyecto que están modelando.

La aplicación está constituida por dos subsistemas: el subsistema web y el subsistema controlador. De esta forma, el proyecto presenta los siguientes paquetes:

- SIGAPTWIT_Web
 - SIGAPTWIT_Web::estilos
 - SIGAPTWIT_Web::javascrips
 - SIGAPTWIT_Web::preguntas_respuesta_libre
 - SIGAPTWIT_Web::sorteos
 - SIGAPTWIT_Web::usuarios
 - SIGAPTWIT_Web::votaciones
- SIGAPTWIT_Control

11.2. PAQUETE SIGAPTWIT_WEB

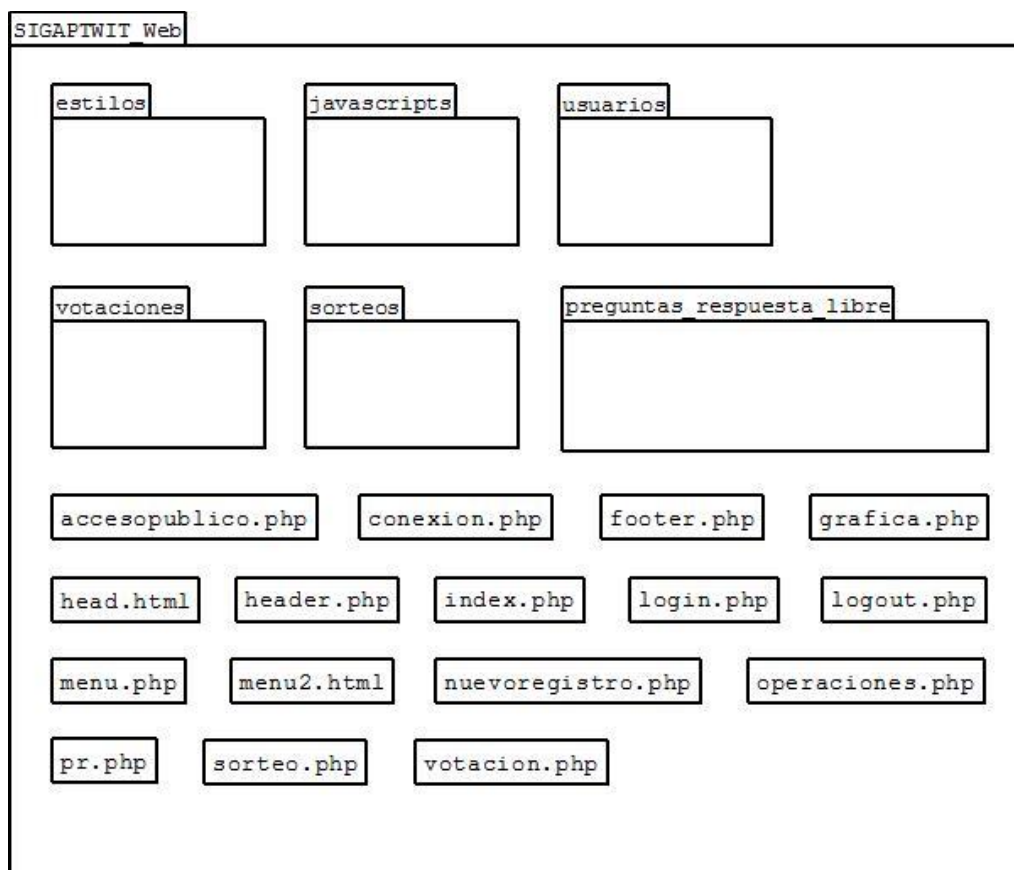


Figura 11.1: Paquete SIGAPTWIT_Web

En el paquete SIGAPTWIT_Web se pueden observar, a su vez, seis paquetes diferentes y dieciséis clases. Los seis paquetes son “estilos”, “javascripts”, “usuarios”, “votaciones”, “sorteos” y “preguntas_respuesta_libre”, que se detallarán más adelante. A continuación se explican cada una de las clases:

- accesopublico.php: archivo programado en php en el que se especifica todo lo relacionado con los datos públicos de las actividades.
- conexion.php: archivo programado en php en el que se definen las variables correspondientes para la conexión a la base de datos.
- footer.php: archivo programado en php en el que se programa el pie de página de la web.
- grafica.php: archivo programado en php en el que se programa las gráficas a mostrar con los resultados.
- head.html: archivo programado en html en el que se encuentran todos los *link* a distintos archivos.
- header.php: archivo programado en php en el que se programa la cabecera de la web.
- index.php: archivo principal programado en php.
- login.php: archivo programado en php en el que se programa todo lo relacionado con el login de un usuario para entrar a la aplicación web.
- logout.php: archivo programado en php en el que se programa la salida del usuario logeado de la aplicación web.
- menu.php: archivo programado en php en el que se programa el menú para un usuario registrado y logeado en la aplicación.
- menu2.html: archivo programado en html en el que se programa el menú para cualquier usuario no registrado en la aplicación.
- nuevoregistro.php: archivo programado en php en el que se programa cómo se crea un nuevo usuario en la web.

- operaciones.php: archivo programado en php en el que se programa la eliminación de las actividades.
- pr.php: archivo programado en php en el que se programa todo lo relacionado con las preguntas / respuestas libres.
- sorteo.php: archivo programado en php en el que se programa todo lo relacionado con los sorteos.
- votacion.php: archivo programado en php en el que se programa todo lo relacionado con las votaciones.

11.2.1. PAQUETE SIGAPTWIT_WEB::ESTILOS



Figura 11.2: Paquete *estilos*

Este paquete contiene los siguientes elementos:

- menu.css: hoja de estilos para el menú a mostrar a un usuario registrado y logeado en la aplicación.
- menu2.css: hoja de estilos para el menú a mostrar a un usuario no registrado en la aplicación.
- style.css: hoja de estilos general de la aplicación.

11.2.2. PAQUETE SIGAPTWIT_WEB::JAVASCRIPTS

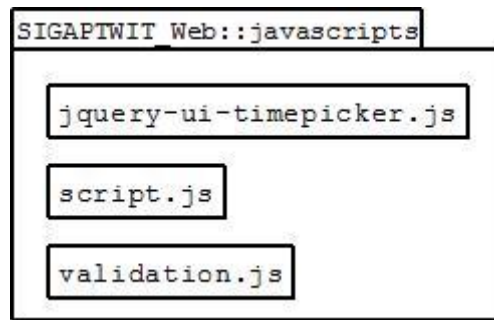


Figura 11.3: Paquete *javascripts*

Este paquete contiene los siguientes elementos:

- jquery-ui-timepicker.js: archivo de javascripts con funciones para la utilidad del calendario al crear una nueva actividad.
- script.js: archivo de javascripts para la creación de las multi-respuestas a la hora de crear una nueva actividad de tipo votación.
- validation.js: archivo de javascripts con funciones para la comprobación de datos.

11.2.3. PAQUETE SIGAPTWIT_WEB::PREGUNTAS_RESPUESTA_LIBRE

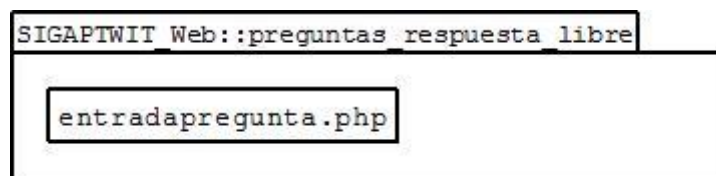


Figura 11.4: Paquete *preguntas_respuesta_libre*

Este paquete contiene solo un elemento:

- entradapregunta.php: archivo programado en php para insertar en la base de datos una nueva actividad de tipo pregunta / respuesta libre.

11.2.4. PAQUETE SIGAPTWIT_WEB::SORTEOS

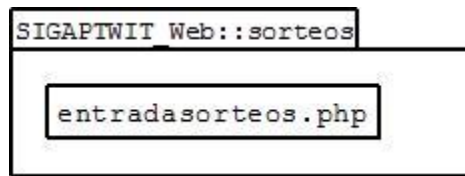


Figura 11.5: Paquete *sorteos*

Este paquete contiene solo un elemento:

- entradasorteos.php: archivo programado en php para insertar en la base de datos una nueva actividad de tipo sorteo.

11.2.5. PAQUETE SIGAPTWIT_WEB::USUARIOS

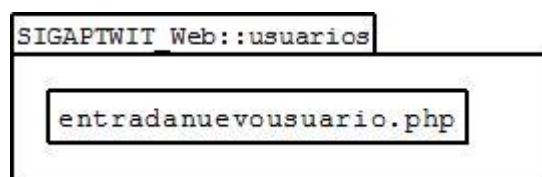


Figura 11.6: Paquete *usuarios*

Este paquete contiene solo un elemento:

- entradanuevousuario.php: archivo programado en php para insertar en la base de datos un nuevo usuario.

11.2.6. PAQUETE SIGAPTWIT_WEB::VOTACIONES

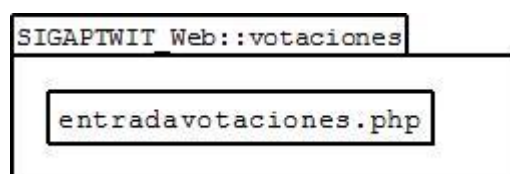


Figura 11.7: Paquete *votaciones*

Este paquete contiene solo un elemento:

- entradavotaciones.php: archivo programado en php para insertar en la base de datos una nueva actividad de tipo votación.

11.3. PAQUETE SIGAPTWIT_CONTROL

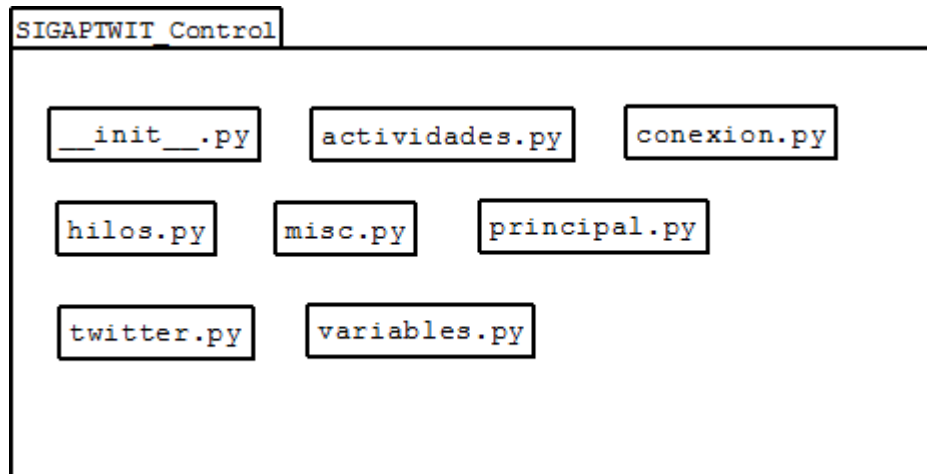


Figura 11.8: Paquete *SIGAPTWIT_Control*

En el subsistema controlador, un paquete es una carpeta que contiene archivos “.py” y debe incluir un archivo de inicio llamado `__init__.py`. Este archivo no es necesario que contenga ninguna instrucción. En el paquete `SIGAPTWIT_Control` se pueden observar los siguientes archivos: `__init__.py`, `actividades.py`, `conexion.py`, `hilos.py`, `misc.py`, `principal.py`, `twitter.py` y `variables.py`.

A continuación se procede a explicar cada uno de ellos:

- `__init__.py`: El principal uso de `__init__.py` es inicializar paquetes de Python. Este archivo es necesario para poder importar desde main de ese directorio. La función de añadir los paquetes importados en el `__init__` sirve para configurar el comando de importación global.
- `actividades.py`: archivo programado en *Python* para manejar las actividades que contiene la aplicación y que ya se han explicado anteriormente.
- `conexion.py`: archivo programado en *Python* para poder interactuar con la base de datos de la aplicación.

- hilos.py: archivo programado en *Python* que permite la ejecución simultánea de procesos, en este caso, la concurrencia de actividades.
- misc.py: archivo programado en *Python* que contiene funciones varias necesarias para el desarrollo del software del subsistema controlador.
- principal.py: archivo programado en *Python* que incluye la información de inicio de la aplicación.
- twitter.py: archivo programado en *Python* que contiene clases y funciones necesarias para interactuar con *Twitter*.
- variables.py: archivo que contiene aquellas variables que pueden cambiar de contenido según la instalación y ejecución de la aplicación.

12. DISEÑO DE CLASES

12.1. INTRODUCCIÓN

En este capítulo se detallan las clases que será necesario utilizar e implementar. Para cada clase se especificará el componente al que pertenece y la capa arquitectónica, así como las relaciones existentes con las demás clases del sistema. Estas clases se extraen tanto de los diagramas de casos de uso y los diagramas de secuencia como de la propia definición del problema. Todas se tendrán en cuenta en el modelo de objetos. En cada una de las clases, que se describen a continuación, aparecen los siguientes apartados.

- Nombre de la clase: nombre asignado a la clase para su identificación. Este nombre debe ser descriptivo, de manera que proporcione una idea acerca del funcionamiento general del elemento al cual pertenece.
- Descripción general de la clase: indica cuál es el objetivo principal de la clase. Se comenta de forma breve el objetivo fundamental de la misma, así como toda aquella información que pudiera resultar de interés en esta etapa.
- Variables miembro de la clase: se analizan todos aquellos atributos que definen la clase y almacenan la información que ésta necesita para desarrollar la funcionalidad que se le ha asignado de manera correcta. Para

cada una de estas variables se indican sus principales características a destacar.

- Métodos de la clase: se muestran todos aquellos métodos definidos por la clase, es decir, aquellas funciones en las que puede estructurarse la funcionalidad de la misma.

12.2. SUBSISTEMA CONTROLADOR

La siguiente sección detalla las clases existentes de forma individual que forman parte del subsistema de control.

Tabla 12.1: Tabla de la clase *Conexion*

Nombre:	Conexion
Descripción:	Esta clase es la encargada de interactuar con la base de datos. Se puede considerar una parte importante del sistema, ya que implementa métodos de suma importancia.
Variables miembro:	<ul style="list-style-type: none"> ▪ datos: almacena los datos de conexión a la base de datos. ▪ cursor: utilizada para obtener información de la base de datos. ▪ conn: es una instancia de la clase MySQLdb utilizada para crear la conexión a la base de datos MySQL.
Métodos:	<ul style="list-style-type: none"> ▪ def __init__ (): es el constructor de la clase. ▪ conectar (): realiza una conexión a la base de datos. ▪ desconectar (): realiza una desconexión de la base de datos. ▪ insertar (): inserta un registro nuevo en la base de datos. ▪ consultar (): realiza una consulta a la base de datos. ▪ comprobarUsuarioRepetido (): este método permite comprobar si un usuario ha participado anteriormente en una actividad concreta. ▪ actualizar (): permite realizar una modificación en un registro de la base de datos. ▪ obtenerActivas (): este método accede a la base de datos en busca de las actividades que deben comenzar a ejecutarse. ▪ obtenerOpcionesVotacion (): este método es el encargado de obtener de la base de datos todas las opciones con las cuales es posible participar en una actividad de tipo votación. ▪ obtenerGanadores (): método mediante el cual se insertan en la base de datos, los registros correspondientes a el/los usuario/usuarios ganadores en una actividad de tipo sorteo.

Conexion
+datos +cursor +conn
- __def__init() +conectar() +desconectar() +insertar() +consultar() +comprobarUsuarioRepetido() +actualizar() +obtenerActivas() +obtenerOpcionesVotacion() +obtenerGanadores()

Figura 12.1: Clase Conexion

Tabla 12.2: Tabla de la clase Actividad

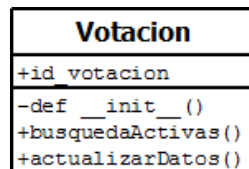
Nombre:	Actividad
Descripción:	Esta clase es la encargada de almacenar los datos comunes referentes a un tipo de actividad.
Variables miembro:	<ul style="list-style-type: none"> ▪ titulo: almacena un nombre identificativo de una actividad. ▪ descripcion: variable de tipo cadena de caracteres que almacena una descripción sobre el tema concreto de la actividad. ▪ fecha_inicio: variable que guarda la hora y la fecha exactas en las que empieza la ejecución de una actividad. ▪ fecha_fin: variable que guarda la fecha y la hora exactas en la que finaliza la ejecución de una actividad. ▪ cuestion: variable de tipo cadena que almacena de forma exacta y concisa la definición de una actividad. ▪ estado: variable numérica que almacena el estado en el cual permanece la actividad. ▪ id_usuario: variable que guarda el código identificativo del usuario el cual ha planteado la actividad de tipo votación.
Métodos	<ul style="list-style-type: none"> ▪ def __init__(): es el constructor de la clase.

Actividad
+titulo +descripcion +fecha_inicio +fecha_fin +cuestion +estado +id_usuario +def __init__()

Figura 12.2: Clase Actividad

Tabla 12.3: Tabla de la clase *Votacion*

Nombre:	Votacion
Descripción:	Esta clase es la encargada de almacenar los datos referentes a una actividad de tipo votación.
Variables miembro:	<ul style="list-style-type: none"> ▪ id_votacion: almacena el número de identificación de una actividad votación dentro de la aplicación.
Métodos:	<ul style="list-style-type: none"> ▪ def __init__(): es el constructor de la clase. ▪ busquedaActivas(): método mediante el cual se buscan aquellas actividades de tipo votación, que están listas para dar comienzo a su ejecución. ▪ actualizarDatos(): método mediante el cual se cambia el estado de una actividad que estaba en espera de ser iniciada, al estado de estar en ejecución.

**Figura 12.3:** Clase *Votacion***Tabla 12.4:** Tabla de la clase *Sorteo*

Nombre:	Sorteo
Descripción:	Esta clase es la encargada de almacenar los datos referentes a una actividad de tipo sorteo.
Variables miembro:	<ul style="list-style-type: none"> ▪ id_sorteo: almacena el número de identificación de un sorteo dentro de la aplicación. ▪ hashtag: variable que almacena el <i>hashtag</i> correcto con el que se puede participar en un sorteo dentro del período en el que la actividad permanezca en ejecución. ▪ aleatorio: variable de tipo booleano que indica si se elegirá/elegirán el/los ganadores aleatoriamente de entre todos los usuarios que participen de forma correcta en el sorteo. ▪ num_ganadores: variable de tipo numérico que almacena el número de ganadores resultantes de una actividad de tipo sorteo.
Métodos:	<ul style="list-style-type: none"> ▪ def __init__(): es el constructor de la clase. ▪ busquedaActivas(): método mediante el cual se buscan aquellas actividades de tipo sorteo, que están listas para dar comienzo a su ejecución. ▪ actualizarDatos(): método mediante el cual se cambia el estado de

	una actividad de tipo sorteo que estaba en espera de ser iniciada, al estado de estar en ejecución.
--	---

Sorteo
+id_sorteo +hashtag +aleatorio +num_ganadores
-def __init__() +busquedaActivas() +actualizarDatos()

Figura 12.4: Clase *Sorteo*Tabla 12.5: Tabla de la clase *Ganador*

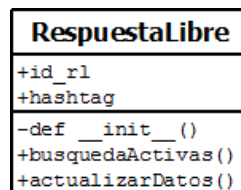
Nombre:	Ganador
Descripción:	Esta clase es la encargada de almacenar los datos referentes a un usuario que ha resultado ganador de una actividad de tipo sorteo.
Variables miembro:	<ul style="list-style-type: none"> ▪ id_ganador: variable numérica que almacena un código identificativo asociado a cada registro en la base de datos, de tipo ganador. ▪ id_tweet: variable que guarda el código numérico del <i>tweet</i> con el cual un usuario ha participado en la actividad de tipo sorteo y ha sido seleccionado como ganador. ▪ id_sorteo: almacena el número de identificación de la actividad tipo sorteo a la cual pertenece un ganador.
Métodos:	<ul style="list-style-type: none"> ▪ def __init__(): es el constructor de la clase. ▪ nuevoObjeto(): este método crea un nuevo objeto la clase ganador. ▪ obtenerGanador(): este método permite seleccionar los ganadores resultantes de una actividad de tipo sorteo, y pueden ser seleccionados de forma aleatoria o seleccionando los <i>n</i> primeros usuarios en participar en el sorteo.

Ganador
+id_ganador +id_tweet +id_sorteo
-def __init__() +nuevoObjeto() +obtenerGanador()

Figura 12.5: Clase *Ganador*

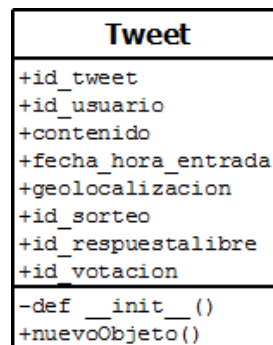
Tabla 12.6: Tabla de la clase *RespuestaLibre*

Nombre:	RespuestaLibre
Descripción:	Esta clase permite crear un objeto que almacena información de una actividad de tipo Respuesta Libre.
Variables miembro:	<ul style="list-style-type: none"> ▪ id_rl: variable de tipo entero que almacena el código único e identificativo de una actividad pregunta / respuesta libre. ▪ hashtag: variable que almacena el <i>hashtag</i> correcto con el que se puede participar en una pregunta / respuesta libre dentro del período en el que la actividad permanezca en ejecución.
Métodos:	<ul style="list-style-type: none"> ▪ def <code>__init__()</code>: es el constructor de la clase. ▪ <code>busquedaActivas()</code>: método mediante el cual se buscan aquellas actividades de tipo pregunta / respuesta libre, que están listas para dar comienzo a su ejecución. ▪ <code>actualizarDatos()</code>: método mediante el cual se cambia el estado de una actividad que estaba en espera de ser iniciada, al estado de estar en ejecución.

**Figura 12.6:** Clase *RespuestaLibre***Tabla 12.7:** Tabla de la clase *Tweet*

Nombre:	Tweet
Descripción:	Esta es la clase encargada de guardar los datos referentes a un <i>tweet</i> y de insertarlos en la base de datos.
Variables miembro:	<ul style="list-style-type: none"> ▪ id_tweet: variable de tipo entero que almacena el código único e identificativo de un <i>tweet</i>. ▪ id_usuario: variable que almacena el nombre de usuario en la red social <i>Twitter</i>. ▪ contenido: variable que almacena el en una cadena de caracteres, el contenido del <i>tweet</i> completo publicado por el usuario de <i>Twitter</i>. ▪ fecha_hora_entrada : variable que guarda la hora y fecha exactas en las que se ha publicado un <i>tweet</i>. ▪ geocalizacion: variable que almacena una cadena de texto con la provincia y municipio desde donde se envió el <i>tweet</i>. ▪ id_sorteo: variable que almacena el código único e identificativo de

	<p>una actividad de tipo sorteo a la cual está vinculado el <i>tweet</i>.</p> <ul style="list-style-type: none"> ▪ <code>id_respuestalibre</code>: variable que almacena el código único e identificativo de una actividad de tipo pregunta / respuesta libre a la cual está vinculado el <i>tweet</i>. ▪ <code>id_votacion</code>: variable que almacena el código único e identificativo de una actividad de tipo votación a la cual está vinculado el <i>tweet</i>.
Métodos:	<ul style="list-style-type: none"> ▪ <code>def __init__()</code>: es el constructor de la clase. ▪ <code>nuevoObjeto()</code>: este método crea un nuevo objeto la clase <i>tweet</i>.

Figura 12.7: Clase *Tweet*Tabla 12.8: Tabla de la clase *HiloPadre*

Nombre:	HiloPadre
Descripción:	Esta clase permite comprobar las actividades que comienzan a ejecutarse, y una vez esto sucede, crea un hilo de comunicación con <i>Twitter</i> . Esta clase recibe información de ese hilo creado, acerca de un <i>tweet</i> , la procesa y almacena en base de datos.
Variables miembro:	<ul style="list-style-type: none"> ▪ <code>actividad</code>: variable numérica que almacena el código unívoco que identifica a una actividad. ▪ <code>tipoActividad</code>: cadena de caracteres que almacena el tipo de actividad al que pertenece la misma. ▪ <code>conexion</code>: variable que recibe un objeto de tipo <i>Conexion</i>, para poder conectar a la base de datos. ▪ <code>hiloHijo</code>: variable que almacena un objeto de la clase <i>HiloHijo</i>. ▪ <code>arrayOpciones</code>: array que almacena todas las opciones con las que se puede participar en una actividad de tipo <i>votación</i>.
Métodos:	<ul style="list-style-type: none"> ▪ <code>def __init__()</code>: es el constructor de la clase. ▪ <code>run()</code>: este método es el encargado de comprobar si existen en base de datos, actividades listas para comenzar a ejecutarse. ▪ <code>informacionTweet()</code>: este método recibe toda la información acerca de un <i>tweet</i>, la procesa y la inserta de forma correcta en la base de

	datos.
--	--------

HiloPadre
+actividad +tipoActividad +conexion +hiloHijo +arrayOpciones
-def __init__() +run() +informacionTweet()

Figura 12.8: Clase *HiloPadre*Tabla 12.9: Tabla de la clase *HiloHijo*

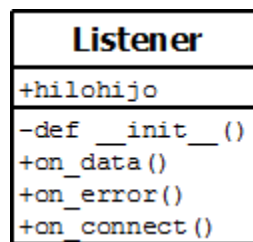
Nombre:	HiloHijo
Descripción:	Esta clase, que hereda de la clase <i>Thread</i> , es la encargada de filtrar los <i>tweets</i> , y cada vez que obtenido uno válido, envía toda la información concerniente a éste a un objeto de la clase <i>HiloPadre</i> .
Variables miembro:	<ul style="list-style-type: none"> ▪ hashtag: variable que almacena el <i>hashtag</i> que esta clase debe encontrar en <i>Twitter</i>. ▪ padre: variable que almacena un objeto de tipo <i>HiloPadre</i>, con el cual se comunica esta clase. ▪ <i>TwitterStream</i>: almacena un objeto de tipo <i>streaming</i>, con el cual se realiza la autenticación en <i>Twitter</i>. ▪ arrayOpciones: variable de tipo <i>array</i>, que almacena todos los <i>hashtag</i> que se deben filtrar de una actividad.
Métodos:	<ul style="list-style-type: none"> ▪ def __init__(): es el constructor de la clase. ▪ obtenerTweet(): este método comprueba si el <i>hashtag</i> de la actividad coincide con el <i>hashtag</i> del <i>tweet</i> filtrado, y si es así, envía toda la información de ese <i>tweet</i> al objeto <i>HiloPadre</i>. ▪ run(): este método inicializa el filtrado del <i>hashtag</i> y muestra un mensaje cuando finaliza su ejecución.

HiloHijo
+hashtag +padre +twitterStream +arrayOpciones
-def __init__() +obtenerTweet() +run()

Figura 12.9: Clase *HiloHijo*

Tabla 12.10: Tabla de la clase *Listener*

Nombre:	Listener
Descripción:	Esta clase permite crear un objeto que hereda de la clase <i>StreamListener</i> , y permite estar a la escucha de <i>tweets</i> vía <i>streaming</i> .
Variables miembro:	<ul style="list-style-type: none"> ▪ <i>hilohijo</i>: variable que almacena el objeto de la clase <i>HiloHijo</i>.
Métodos:	<ul style="list-style-type: none"> ▪ <code>def __init__()</code>: es el constructor de la clase. ▪ <code>on_data()</code>: este método permite leer toda la información perteneciente a un <i>tweet</i>, desde un fichero de texto, y la transforma a una variable de tipo diccionario. ▪ <code>on_error()</code>: este método devuelve un mensaje de error si no se ha podido establecer la comunicación <i>streaming</i> con <i>Twitter</i>. ▪ <code>on_connect()</code>: este método muestra un mensaje indicando qué hashtag han comenzado a filtrarse de entre todos los <i>tweets</i> posibles enviados a la red social.

**Figura 12.10:** Clase *Listener*

13. DISEÑO ARQUITECTÓNICO

13.1. DIAGRAMA DE DESPLIEGUE

El diagrama de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. En la figura 13.1 se muestra el diagrama de despliegue para la aplicación *SIGAPTWIT*.

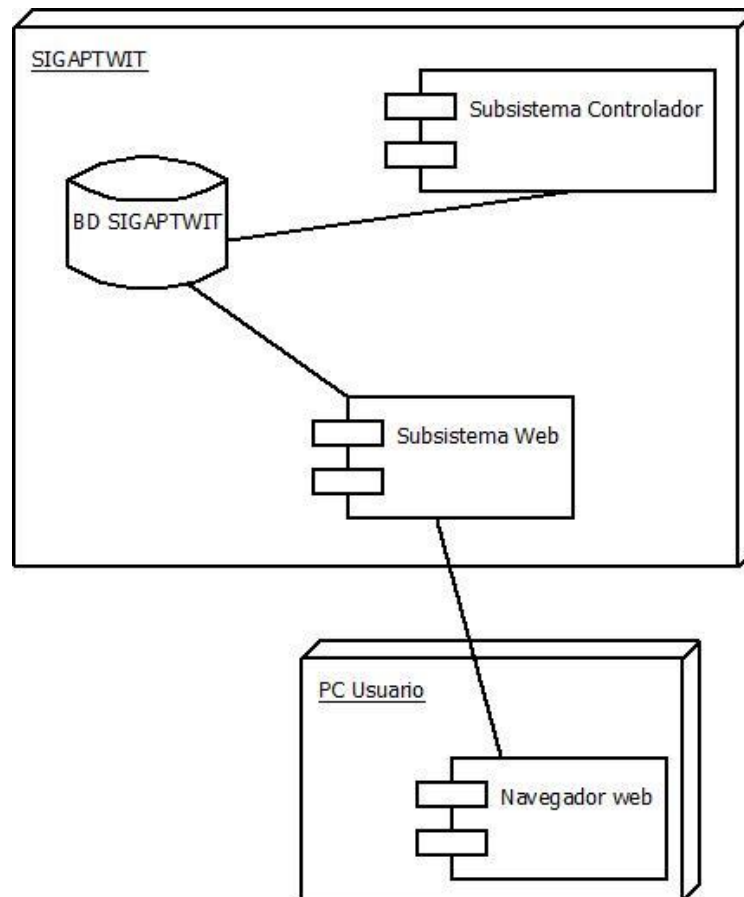


Figura 13.1: Diagrama de despliegue

13.2. DESCRIPCIÓN DE LOS NODOS

A continuación se describen los nodos mostrados en el diagrama de despliegue.

Tabla 13.1: Nodo: *PC Usuario*

Nodo: PC Usuario	
Descripción	Equipo informático del usuario que, mediante el navegador web, utilizará el sistema <i>SIGAPTWIT</i> .
Componentes	Navegador web
Dependencias	<u>Nodo</u> : <i>SIGAPTWIT</i> <u>Componente</u> : Subsistema Web

Tabla 13.2: Nodo: *SIGAPTWIT*

Nodo: SIGAPTWIT	
Descripción	Sistema Informático para la Gestión de Aplicaciones de Participación social a través de <i>Twitter</i> .
Componentes	Subsistema Controlador Subsistema Web

Dependencias	<u>Nodo</u> : PC Usuario <u>Componente</u> : Navegador Web
---------------------	---

13.3. DESCRIPCIÓN DE LOS COMPONENTES

A continuación se describen los componentes, ubicados en los distintos nodos, que tienen relevancia en el presente proyecto (22).

Tabla 13.3: Componente: *Navegador web*

Componente: Navegador web	
Descripción	Aplicación que permite visualizar e interactuar con contenidos web.
Dependencias	<u>Componente</u> : Subsistema web
Implementaciones	Internet Explorer, Mozilla Firefox, Google Chrome, etc

Tabla 13.4: Componente: *Subsistema Controlador*

Componente: Subsistema Controlador	
Descripción	Es el encargado de gestionar la información de todo lo relacionado con los usuarios de <i>Twitter</i> y la aplicación.
Dependencias	<u>Componente</u> : BD_SIGAPTWIT
Implementaciones	<i>Twitter, SIGAPTWIT.</i>

Tabla 13.5: Componente: *Subsistema Web*

Componente: Subsistema Web	
Descripción	Es el encargado de gestionar la información de la parte web de la aplicación, junto con la respectiva base de datos.
Dependencias	<u>Componente</u> : BD_SIGAPTWIT <u>Componente</u> : Navegador web
Implementaciones	<i>SIGAPTWIT.</i>

14. DISEÑO DE LA INTERFAZ

14.1. INTRODUCCIÓN

En este capítulo se van a mostrar, a partir de los criterios establecidos en el capítulo 9, *Especificación de requisitos de la interfaz*, los diseños de las principales ventanas de la interfaz de usuario. También se describen otros elementos, tales como el aspecto que presentarán las diferentes zonas o secciones en las que estarán divididas dichas ventanas y diálogos, así como los componentes contenidos en estas secciones y la funcionalidad de cada uno de ellos. Para ello se hará uso de capturas de pantalla de la propia aplicación. En su diseño se ha intentado que la distribución de los elementos sea lo más intuitiva posible.

14.2. PÁGINA INICIAL

La función principal de esta página es la de proporcionar al usuario un medio para poder utilizar toda la funcionalidad que la aplicación puede ofrecerle.

El diseño que se ha realizado para la página inicial es el que se muestra en la siguiente figura:



Figura 14.1: Diseño página inicial

Los elementos de la interfaz a destacar están señalados en la imagen con números. A continuación se detallan:

1. Logo del sistema SIGAPTWIT. Se trata de un enlace que al clicar sobre él, conduce directamente a la página principal de la aplicación.
2. Formulario de login. Se trata de un formulario para que los usuarios ya registrados previamente, puedan acceder al sistema.
3. Botón “Entrar”. Se trata de un botón que al clicar sobre él, si los datos introducidos del usuario son correctos, muestra la página en la cual el usuario puede crear o borrar una actividad.
4. Nombre completo de la aplicación.
5. Botón “Ver resultados”. Se trata de un botón para que cualquier usuario (ya sea registrado en el sistema o no) pueda ver las gráficas con los resultados en tiempo real de cualquier actividad con perfil público.
6. Botón “Registrar”. Se trata de un botón para que cualquier usuario no registrado en el sistema pueda darse de alta como nuevo usuario.
7. Logo y nombre de la Universidad de Córdoba.

14.3. PÁGINA REGISTRO NUEVO USUARIO

Si se pulsa sobre el botón “Registrar” desde la página inicial, la aplicación muestra la página que se refleja en la figura 14.2 para que un usuario que no está registrado en el sistema, pueda hacerlo.

Figura 14.2: Diseño página nuevo usuario

Los elementos son:

8. Formulario de registro. Se trata de un formulario para que el nuevo usuario lo rellene con sus datos personales.
9. Botón “crear”. Se trata de un botón para guardar los datos introducidos anteriormente en la base de datos.

14.4. PÁGINA PARA VER RESULTADOS

Si desde la página inicial se pulsa sobre el botón “Ver resultados”, se muestra la siguiente página para que cualquier usuario (registrado o no en el sistema), pueda acceder a cierta información de las actividades públicas de la aplicación. A continuación se muestra una imagen para reflejarlo (figura 14.3):

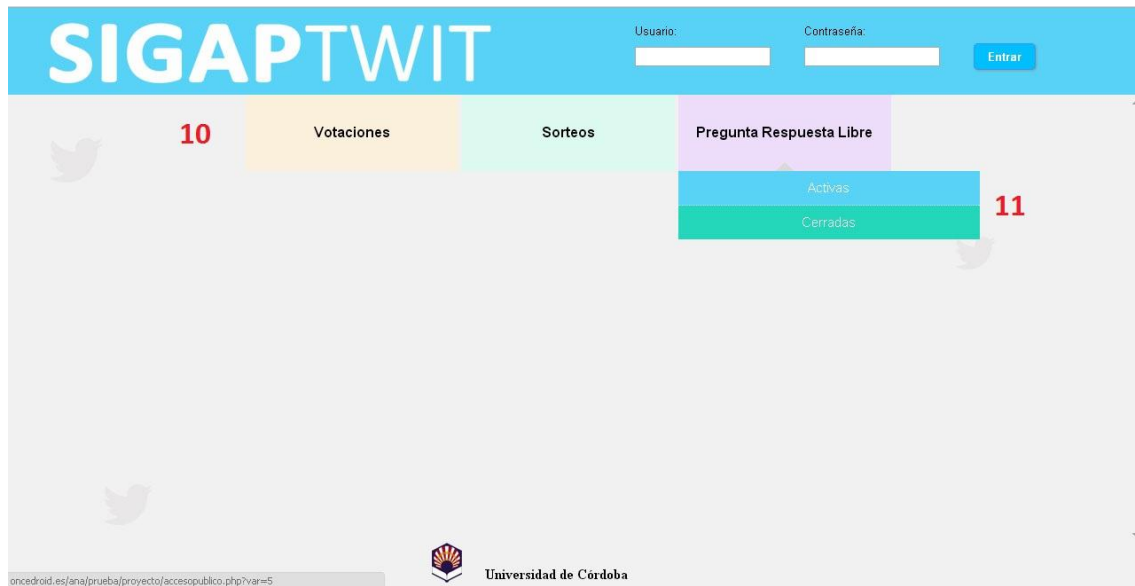


Figura 14.3: Diseño página ver resultados I

10. Menú actividades. Se trata de un menú en el que las opciones son los distintos tipos de actividades que existen. Éstas son *Votaciones*, *Sorteos* y *Pregunta / Respuesta Libre*.
11. Submenú actividades. Así mismo existe un submenú para cada una de las actividades en las que el usuario, para ver los resultados, puede elegir entre las actividades que están activas en ese momento o las que están cerradas. Una vez que el usuario elija, se mostrará la siguiente página (figura 14.4) con los resultados de la actividad elegida:



Figura 14.4: Diseño página ver resultados II

Y, si se pulsara en el botón “Ver”, se obtendría la siguiente vista en la que se muestran los resultados mediante un gráfico de barras.

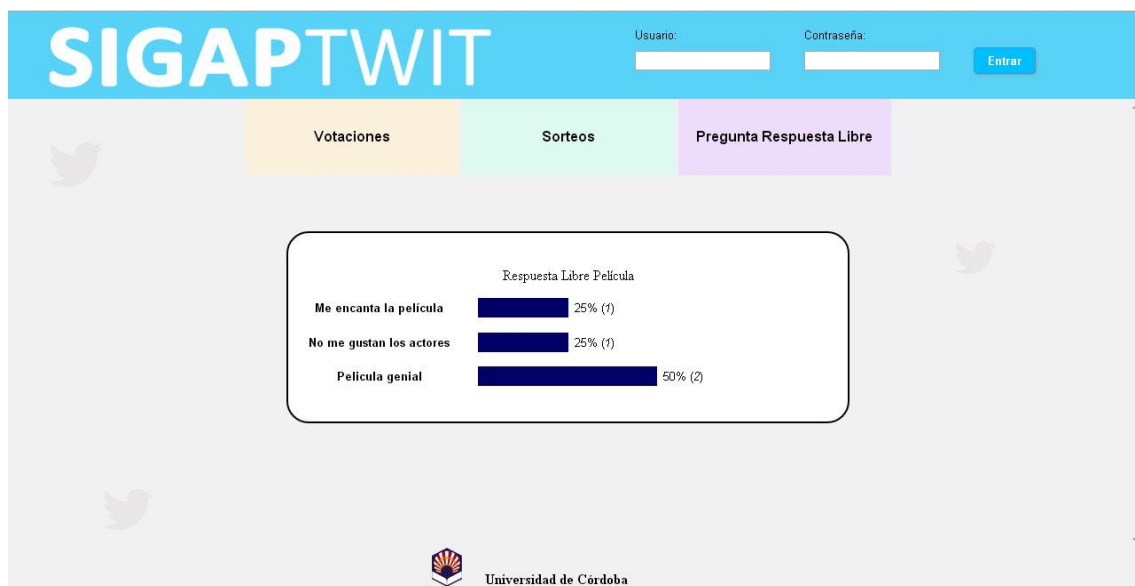


Figura 14.5: Diseño página ver resultados III

14.5. PÁGINA PARA GESTIONAR ACTIVIDADES

Si desde la página inicial el usuario introduce correctamente sus datos en el formulario de *login* y pulsa sobre el botón “Entrar”, se muestra la siguiente página:

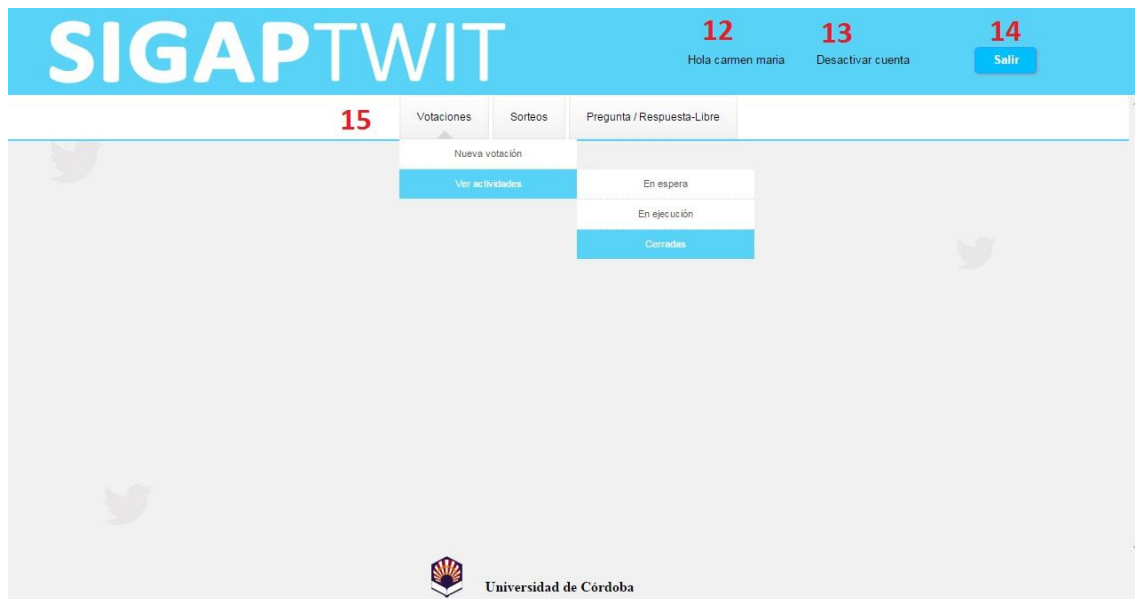


Figura 14.6: Diseño página gestionar actividades

1. Mensaje de Bienvenida. Se trata de un breve mensaje de bienvenida para el usuario que acaba de entrar al sistema. Dicho mensaje está compuesto por la palabra “Hola” más el nombre del usuario.
2. Botón “Desactivar cuenta”. Se trata de un botón para que, en el caso de que el usuario desee darse de baja en el sistema, pueda hacerlo mediante el uso de este botón.
3. Botón “Salir”. Se trata de un botón para que el usuario pueda salir de la aplicación y volver a la página de inicio.
4. Menú actividades. Se trata de un menú en el que las opciones son los distintos tipos de actividades que existen. A su vez, si se posa el ratón por alguna de ellas, se muestra un submenú con las opciones:
 - a. *Nueva votación* (en el caso de las votaciones). Para poder crear una nueva actividad. En la siguiente imagen (figura 14.7) se muestra el formulario para crear una nueva actividad. En este caso, una nueva votación:

The screenshot displays the SIGAPTWIT web application. At the top, a blue header contains the logo 'SIGAPTWIT' on the left, and user information 'Hola carmen maria' and 'Desactivar cuenta' on the right, along with a 'Salir' button. Below the header is a navigation bar with three tabs: 'Votaciones' (selected), 'Sorteos', and 'Pregunta / Respuesta-Libre'. The main content area features a light blue background with faint Twitter bird icons. In the center is a white box titled 'NUEVA VOTACION' containing a form with the following fields: 'Titulo *' (text input), 'Descripción' (text area), 'Inicio *' (text input), 'Fin *' (text input), 'Pregunta *' (text input), 'Respuestas *' (two text inputs, each with a '#Respuesta' placeholder and a small 'x' icon), and 'Acceso público: ☒'. A blue 'Crear' button is at the bottom of the form. Below the form, the logo of the 'Universidad de Córdoba' is displayed.

Figura 14.7: Diseño página gestionar votación

- b. *Ver actividades.* A su vez muestra otro nuevo submenú con las opciones *en espera*, *en ejecución* y *cerradas*. Finalmente, si el usuario pulsa sobre alguna de estas últimas opciones, se muestra en la página una tabla con los datos de cada actividad (título, descripción, fecha inicial, fecha final, pregunta, hashtag) y dos botones (resultados y eliminar).

15. PRUEBAS

15.1. INTRODUCCIÓN

Una de las fases más importantes durante la realización de una aplicación software es la comprobación del correcto funcionamiento de la misma.

Las pruebas del software son un elemento crítico para garantizar la eficiencia de uso del producto y están destinadas a comprobar que la aplicación satisface todos los objetivos establecidos, y que los acomete de forma satisfactoria.

A través de estas pruebas se puede determinar su fiabilidad y consistencia y en base a ellas es cómo se realizan las debidas correcciones, hasta obtener un sistema completamente depurado y de calidad.

15.2. PRUEBAS REALIZADAS

En este capítulo se describen las pruebas que se han realizado sobre la aplicación, describiendo los problemas encontrados, si los hubo, y las soluciones que se decidieron adoptar para solventar dichos problemas. Durante la fase de codificación se realizaron gran cantidad de pruebas que también estarán reseñadas en este capítulo.

En la mayoría de ocasiones, cuando se encontró un fallo, se volvió a la fase de *codificación* del proceso de desarrollo. Sin embargo, existen problemas que requieren volver a fases anteriores.

Las pruebas realizadas han sido las siguientes:

15.2.1. PRUEBAS DE INSTALACIÓN DE LA APLICACIÓN

Tabla 15.1: Prueba 1

Prueba 1	
Descripción	Creación de la base de datos.
Problemas encontrados	Ninguno.
Soluciones adoptadas	Ninguna.

15.2.2. PRUEBAS DE FUNCIONAMIENTO DE LA APLICACIÓN

Tabla 15.2: Prueba 2

Prueba 2	
Descripción	Un usuario sin registrar intenta acceder al sistema.
Problemas encontrados	No se le muestra al usuario que los datos son incorrectos. Además permite a dicho usuario crear cualquier tipo de actividad, así como acceder a datos privados.
Soluciones adoptadas	Una vez que se ha verificado que el usuario no existe en la base de datos, el sistema carga de nuevo la página de inicio mostrando un mensaje por pantalla indicando que los campos son incorrectos.

Tabla 15.3: Prueba 3

Prueba 3	
Descripción	El usuario deja vacío el campo dedicado a escribir su usuario y/o contraseña.
Problemas encontrados	No se le muestra al usuario que debe rellenar los campos. Además permite a dicho usuario crear cualquier tipo de actividad, así como acceder a datos privados.
Soluciones adoptadas	Una vez que se ha verificado que el campo de usuario y/o contraseña están vacíos, se muestra un mensaje por pantalla al usuario indicando que dichos campos deben ser completados.

Tabla 15.4: Prueba 4

Prueba 4	
Descripción	El usuario escribe incorrectamente su usuario o contraseña.
Problemas encontrados	No se le muestra al usuario que los datos son incorrectos. Además permite a dicho usuario crear cualquier tipo de actividad, así como acceder a datos privados.
Soluciones adoptadas	Una vez que se ha verificado que el usuario y/o contraseña no corresponden con los que se introdujeron en el formulario de registro, se muestra un mensaje de error por pantalla al usuario.

Tabla 15.5: Prueba 5

Prueba 5	
Descripción	El usuario deja vacío alguno de los campos obligatorios durante su registro.
Problemas encontrados	No se le muestra al usuario un mensaje de error para informarle que debe introducir los campos obligatorios. Además, el sistema crea el usuario en la base de datos.
Soluciones adoptadas	Una vez que se ha verificado que algunos de los campos obligatorios están vacíos, se muestra un mensaje por pantalla al usuario indicando que dichos campos deben ser completados.

Tabla 15.6: Prueba 6

Prueba 6	
Descripción	El usuario, al registrarse en la aplicación, no introduce la misma contraseña en los campos destinados para ello.
Problemas encontrados	No se le muestra al usuario un mensaje de error para informarle que las contraseñas tienen que ser las mismas. Además, el sistema crea el usuario en la base de datos.
Soluciones adoptadas	El sistema comprueba que el valor de cada campo introducido sea el mismo. Si no es así, se muestra un mensaje de error por pantalla.

Tabla 15.7: Prueba 7

Prueba 7	
Descripción	El sistema accede a la base de datos.
Problemas encontrados	Ninguno.
Soluciones adoptadas	Ninguna.

Tabla 15.8: Prueba 8

Prueba 8	
Descripción	El usuario deja vacío alguno de los campos obligatorios durante el registro de una actividad.
Problemas encontrados	No se le muestra al usuario un mensaje de error para informarle que debe introducir los campos obligatorios. Además, el sistema crea el usuario en la base de datos.
Soluciones adoptadas	Una vez que se ha verificado que alguno de los campos obligatorios está vacío, se muestra un mensaje por pantalla al usuario indicando que dichos campos deben ser completados.

Tabla 15.9: Prueba 9

Prueba 9	
Descripción	El usuario no introduce el símbolo “#” en la definición del campo <i>hashtag</i> .
Problemas encontrados	No se le muestra al usuario un mensaje de error.
Soluciones adoptadas	Una vez que se ha verificado que el campo del <i>hashtag</i> no empieza por #, se muestra un mensaje por pantalla al usuario indicando el error.

Tabla 15.10: Prueba 10

Prueba 10	
Descripción	El usuario introduce en el campo <i>hashtag</i> , espacios en blanco.
Problemas encontrados	El sistema admite un <i>hashtag</i> que contiene espacios en blanco sin mostrar ningún mensaje de alerta.
Soluciones adoptadas	Una vez que se ha verificado que el campo del <i>hashtag</i> contiene espacios en blanco, se muestra un mensaje por pantalla al usuario indicando el error.

Tabla 15.11: Prueba 11

Prueba 11	
Descripción	El sistema no empieza a ejecutar una actividad a la hora de inicio indicada en su creación.
Problemas encontrados	El sistema controlador no inicia la ejecución de una actividad aun cumpliendo las condiciones impuestas de hora y fecha de inicio y estado.
Soluciones adoptadas	Revisar que la conexión a la base de datos de <i>Mysql</i> es correcta. Revisar el estado de dicha actividad.

Tabla 15.12: Prueba 12

Prueba 12	
Descripción	El sistema no es capaz de conectar con <i>Twitter</i> .
Problemas encontrados	El sistema controlador no obtiene la información de <i>Twitter</i> y muestra por pantalla en el <i>log</i> un mensaje de conexión fallida.
Soluciones adoptadas	Revisar que existe conexión con el servicio de <i>Twitter</i> . Revisar las credenciales proporcionadas por <i>Twitter</i> para conectar a sus servicios.

Tabla 15.13: Prueba 13

Prueba 13	
Descripción	El sistema contabiliza varias respuestas distintas para una misma actividad por parte de un mismo usuario.
Problemas encontrados	En la base de datos aparecen introducidos varios <i>tweets</i> diferentes para una misma actividad por parte de un mismo usuario de <i>Twitter</i> .
Soluciones adoptadas	Implementar un filtro mediante el cual un usuario solo puede participar una sola vez en una misma actividad, pero sí pudiendo participar en otras actividades diferentes activas.

Tabla 15.14: Prueba 14

Prueba 14	
Descripción	El sistema contabiliza respuestas sin distinguir el <i>hashtag</i> entre letras mayúsculas y minúsculas.
Problemas encontrados	En la base de datos se introducen <i>tweets</i> sin tener en cuenta la diferencia entre letras mayúsculas y minúsculas de un <i>hashtag</i> , es decir, se aceptan todos los <i>tweets</i> cuyo <i>hashtag</i> contiene las mismas letras estén escritas indistintamente con mayúsculas y minúsculas.
Soluciones adoptadas	Implementar en código una condición mediante la cual sólo se aceptan aquellas respuestas cuyo <i>hashtag</i> coincide exactamente con el propuesto para una actividad, haciendo distinción entre letras mayúsculas y minúsculas.

Tabla 15.15: Prueba 15

Prueba 15	
Descripción	El sistema no muestra ningún dato como resultado de una actividad finalizada.
Problemas encontrados	No aparecen resultados para una actividad que ya ha finalizado.
Soluciones adoptadas	Revisar si existen usuarios que han participado en dicha actividad, ya que puede darse el hipotético caso de que no haya participado nadie esa actividad.

16. CONCLUSIONES

16.1. INTRODUCCIÓN

Una vez concluidas todas y cada una de las fases de desarrollo de la aplicación software, se exponen las conclusiones a las que se ha llegado tras la finalización de dichas fases. Estas conclusiones relacionan los objetivos planteados al comienzo del desarrollo, descrito en el presente documento (capítulo 2), con los resultados obtenidos durante la fase de prueba del mismo (capítulo 15).

El resultado es una aplicación informática que permite conocer la opinión ciudadana sobre temas concretos a través de la red social *Twitter*, desarrollada y planteada como respuesta a una necesidad ante la falta de aplicaciones de similares características.

Cabe destacar los siguientes aspectos:

- Se trata de una aplicación web multiplataforma y multiusuario.
- En cuanto a los aspectos técnicos que presenta la aplicación, cabe destacar:
 - La elección del lenguaje de programación *Python*, ha supuesto una gran ventaja a la hora de codificar la aplicación, debido a su simpleza unida a una gran robustez. El uso de *Python* conlleva también, una mejor legibilidad del código generado, debido a la propia naturaleza del lenguaje.

- Al tener una interfaz web, el uso de la aplicación se simplifica bastante, ya que en la actualidad, el conocimiento y la utilización de aplicaciones web y las redes sociales, en concreto *Twitter*, están muy extendidos entre la población.

16.2. CONCLUSIONES SOBRE LOS OBJETIVOS PLANTEADOS

Todos los objetivos que fueron definidos en el capítulo 2 del presente manual han sido cumplidos:

- Se puede decir que el objetivo principal del proyecto ha sido alcanzado, pues éste consistía en el desarrollo de una herramienta que posibilita saber criterios sobre temas planteados, usando como medio de comunicación la red social *Twitter*. La conclusión final para este objetivo es su consecución total en base a los resultados expresados anteriormente.
- Se ha definido e implementado una arquitectura software extensible por lo que la aplicación es susceptible de poder ser mejorada en un futuro, debido a la modularidad de la que dispone.

16.3. CONCLUSIONES DE LA FASE DE PRUEBAS

Durante la fase de pruebas se han llevado a cabo diversas ejecuciones de la aplicación con las que se ha pretendido que ocurran todos los casos que se expusieron con anterioridad y que eran susceptibles de que la aplicación fallara.

Tras la realización de todas estas ejecuciones, se pudo comprobar que la aplicación no sólo se mantenía estable, sino que todos los datos que mantenía conservaban su integridad. Aun con estos resultados no se puede afirmar que la aplicación esté plenamente libre de errores.

Estos posibles errores se deben a que pueden ocurrir situaciones no contempladas que se hayan pasado por alto en la ejecución de la aplicación, debido principalmente a que el período de prueba de la aplicación ha sido corto con respecto al período de tiempo que se va a estar utilizando el sistema software, por lo que existe

cierta probabilidad de que con el uso continuo del mismo pueda aparecer algún error que haya que subsanar.

16.4. DOCUMENTACIÓN

Se ha elaborado una documentación técnica que describe los procesos de análisis y diseño del proyecto *SIGAPTWIT*. Esta documentación facilita posibles modificaciones y actualizaciones del sistema realizadas por una persona distinta a las autoras del proyecto.

Se ha elaborado un manual de usuario que describe los elementos de la interfaz, el funcionamiento del sistema al completo y varios ejemplos de uso de la aplicación en la práctica.

El código de los diferentes archivos de la aplicación se encuentra debidamente comentado para que una persona con conocimientos de programación (*Python*, *HTML*, *Javascript*, *PHP* y *CSS*) pueda interpretarlo y realizar modificaciones oportunas.

17. FUTURAS MEJORAS

Como punto y final a la realización de este Proyecto Fin de Carrera, incluso habiendo cumplido con los objetivos propuestos al principio del proyecto, cabe citar que a lo largo del proceso de desarrollo se han observado algunas mejoras que enriquecerían el marco de trabajo del sistema.

En este capítulo se van a proponer algunas posibles mejoras que pueden incluirse en versiones posteriores del sistema *SIGAPTWIT*:

- Sería de utilidad que la aplicación permitiese al usuario modificar sus datos de registro, como por ejemplo su email o contraseña, así como los datos de cualquier actividad realizada sin necesidad de eliminarla y volver a crearla.
- Sería interesante que la aplicación permitiese al usuario recuperar su contraseña en el supuesto caso que éste la haya extraviado o no la recuerde.
- Resultaría útil adaptar la aplicación para que funcionase correctamente en dispositivos móviles como smartphones o tablets.
- Sería de gran ayuda que la aplicación permitiera al administrador de la plataforma, mediante un panel de control, gestionar todas las actividades.

18. BIBLIOGRAFÍA

- [1] Ericsson - http://www.ericsson.com/res/docs/2012/consumerlab/tv_video_consumerlab_report.pdf [Última consulta: Septiembre de 2014].
- [2] Blog twitter - <https://blog.twitter.com/2013/a-look-back-at-the-oscar> [Última consulta: Septiembre de 2014].
- [3] Luparoom - <http://www.luparoom.com/index.php/men-prin-blog/60-la-voz-hito-television-social> [Última consulta: Septiembre de 2014].
- [4] Twitter - <https://about.twitter.com> [Septiembre de 2014].
- [5] Python - <http://www.pythoncentral.io/introduction-to-tweepy-twitter-for-python> [Última consulta: Septiembre de 2014].
- [6] Booch G., Rumbaugh J. y Jacobson I. *The Unified Modeling Language User Guide*. s.l. : Addison-Wesley Professional. ISBN-10: 0201571684, ISBN-13: 9780201571684.
- [7] JSON - <http://json.org> [Última consulta: Septiembre de 2014].
- [8] Tweepy - <http://www.tweepy.org> [Última consulta: Septiembre de 2014].
- [9] Twitter - <https://dev.twitter.com/overview/documentation> [Última consulta: Septiembre de 2014].
- [10] Pollowers - <http://www.pollowers.com> [Última consulta: Octubre de 2014].
- [11] Twtpoll | Social Survey Platform - <http://twtpoll.com> [Última consulta: Octubre de 2014].
- [12] Sortweet | Sorteos profesionales en Twitter - <http://sortweet.com> [Última consulta: Octubre de 2014].

- [13] Sorteos en Twitter - <<http://sortwit.com>> [Última consulta: Octubre de 2014].
- [14] Luque Ruiz, I., Gómez - Nieto, M. A., López Espinosa E. y Cerruela García G. 2001. *Bases de Datos, Desde Chen hasta Codd con ORACLE*. RA-MA. ISBN-10: 8478974784, ISBN-13: 978-8478974788.
- [15] Normalización de base de datos - Wikipedia
<http://es.wikipedia.org/wiki/Normalizaci%C3%B3n_de_bases_de_datos> [Última consulta: Octubre de 2014].
- [16] Modelo de datos relacional - <http://www.cs.us.es/cursos/bd-2001/temas/modelo_relacional.html> [Última consulta: Noviembre de 2014].
- [17] Modelo de datos relacional - <http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02148.pdf> [Última consulta: Noviembre de 2014].
- [18] Silberschatz, Korth y Sudarshan. *Fundamentos de bases de datos*. 2007. ISBN: 9788448156718.
- [19] Nevado Cabello, M.V. *Introducción a Las Bases de Datos Relacionales*. 2010. ISBN-10: 8498868092, ISBN-13: 978-8498868098.
- [20] MySQL - <<http://dev.mysql.com/doc/refman/5.7/en/connecting-disconnecting.html>> [Última consulta: Enero de 2015].
- [21] Rumbaugh, I. Jacobson, G. Booch. *El lenguaje unificado de Modelado. Manual de Referencia*. 2007. Pearson Educación, s.a., Segunda Edición. ISBN: 978-84-7829-087-1.
- [22] Perdita, S. y Pooley, R. *Utilización de UML en Ingeniería del Software con objetos y componentes*. 2007. Pearson Educación, s.a. ISBN: 978-84-7829-086-4.
- [23] Dependencias funcionales -
<http://decsai.ugr.es/~gnavarro/files/docencia/DDSI/DDSI_Tema_4B_x4.pdf> [Última consulta: Enero de 2015].