UNIVERSIDAD CARLOS III DE MADRID
DEPARTAMENTO DE INGENIERÍA TELEMÁTICA

Juego educativo con competición síncrona utilizando J2EE

PROYECTO FIN DE CARRERA:

INGENIERÍA TÉCNICA DE TELECOMUNICACIONES,
ESPECIALIDAD EN TELEMÁTICA

- Autora: **Elvira Padrino Davia**
- Tutor: **Pedro José Muñoz Merino**

Leganés, Octubre de 2015

“Dime y lo olvido,

enséñame y lo recuerdo,

Involúcrame y lo aprendo”.

Benjamin Franklin

“En cuestiones de cultura y de saber,

sólo se pierde lo que se guarda,

sólo se gana lo que se da”.

Antonio Machado

Agradecimientos

Quisiera agradecer el apoyo de todos aquellos que me han motivado a realizar el proyecto, tanto a amigos como a maestros. En concreto, quiero agradecer a mis padres el gran apoyo económico y, sobretodo, emocional que han mostrado durante todos estos años hasta el final de la carrera. Ellos han sido un importante motivo a sumar a los principales que me han llevado a concluir esta ingeniería.

Quiero hacer mención especial a mi tutor Pedro José Muñoz Merino ya que, siempre con una sonrisa, no paró de motivarme y animarme durante todo el proceso, por lo que le estoy muy agradecida.

Resumen

Se puede jugar y aprender a la vez. Es posible divertirse mientras adquirimos nuevos conocimientos. En este proyecto se trata de elaborar una aplicación web en J2EE de un juego educativo con competición síncrona que cumpla estos objetivos.

El juego que aquí se desarrolla es del tipo "Quiz", de preguntas y respuestas respecto a una materia que el profesor decida. Este juego tiene gran libertad. El propio docente pone a disposición del alumnado las preguntas que él considere y del temario que crea oportuno.

El juego realizado en este proyecto funciona a base de torneos pregrabados por el profesor. Cada alumno ya registrado y dado de alta correctamente (todo lo gestiona el propio administrador), verá en sus pantallas un contador de tiempo, marcha atrás, en el que indicará el tiempo que queda para el siguiente torneo. De forma síncrona, la idea es que todos los alumnos jueguen en el aula o de manera remota al ser una aplicación web. A todos y cada uno de ellos le asaltarán las diferentes preguntas a las que deben responder de forma síncrona dando lugar a una competición en el que ganará el que acierte habiendo sido el más rápido en contestar.

Palabras clave: juego, educativo, competición , arquitectura, J2EE, concurso, quiz, *java*, MySQL y MVC.

Índice de contenidos

Índice de figuras:

1. Introducción

1.1. Motivación

La motivación principal de este proyecto es que los alumnos puedan aprender y divertirse a la vez, compitiendo con otros alumnos, de manera síncrona, mediante un juego educativo tipo Quiz (de preguntas y respuestas). De este modo el alumno se sentirá con más motivación para aprender previamente los conocimientos precisos para el juego o a adquirirlos mediante el mismo acertando y/o errando en las diferentes cuestiones. Esto hará que entren en una dinámica de superación personal donde aprendan más de la materia para lograr una puntuación elevada. Cada alumno podrá ver el ranking de las mejores puntuaciones y querrá ponerse en cabeza.

La idea es crear un juego visualmente simple y práctico para que el usuario se familiarice rápidamente con la jugabilidad. Gracias a la ampliación de sus conocimientos en la materia de la asignatura a la que se aplique, irá aumentando su puntuación, en la que se tendrá en cuenta el número de aciertos y la rapidez con que conteste a cada una de las preguntas.

La intención es que todos los alumnos jueguen de forma simultánea en el aula o bien fuera de ella.

Otra motivación es que este juego le sirva como herramienta al profesor para que pueda constatar y evaluar el progreso de cada alumno y comprobar qué nivel medio tiene su clase y cómo se han comprendido y asimilado todos los conocimientos impartidos, llegando a ser así un método de evaluación habitual.

Cada vez la tecnología está más integrada en la educación por lo que cada alumno en los laboratorios de ordenadores o bien con su ordenador portátil

personal, tableta o incluso dispositivo móvil pueden conectarse, mientras que el profesor imparte la asignatura. Esto podría verse como un problema, pero también puede ser una ventaja. Ya que se puede aprovechar dicha tecnología para la docencia.

Pero la idea es ir un paso más allá y no solo lograr que los alumnos aprendan los conocimientos de la asignatura en el aula; sino que el alumno pueda divertirse jugando y compitiendo con sus compañeros de aula con un juego didáctico que contenga los conocimientos adquiridos de la asignatura.

1.2. Objetivos

El objetivo principal de este proyecto es la creación de una aplicación web de un juego educativo utilizando J2EE, diseñarlo y programar la lógica del juego.

El juego debe cumplir los siguientes requisitos:

- Debe ser una aplicación web orientada a cualquier dispositivo que tenga alguno de estos navegadores web: Chrome, Explorer, Firefox, Safari y Opera.
- La idea principal es lograr que los alumnos aprendan los conocimientos de la asignatura mientras que se divierten jugando y compitiendo con sus compañeros de aula con un juego educativo que contenga los conocimientos adquiridos.
- Debe seguir una estética sencilla típica de los juegos Quiz, a la vez que un diseño atractivo con el clásico temporizador de este tipo de juegos.
- Ha de ser un juego motivador que cree adicción.

- Tiene que ser un juego ágil, con una serie de temporizadores que hagan que el jugador responda muy rápido.

- Debe de adaptarse a todas las edades con la única limitación de los contenidos dados por el docente, el cual edita y carga las preguntas que estime conveniente.

- La puntuación se guardará de forma automática para ir tomando las posiciones más altas en el.

- Todas las preguntas son editables desde un fichero XML, sin tener necesidad de acceder al código interno del juego, facilitándole esa labor al docente.

- Sólo el profesor tendrá una serie de privilegios como son: administrar las altas y bajas de los alumnos que lo y también borrar las puntuaciones de todos los alumnos o de alguno en concreto.

1.3. Explicación de capítulos

El capítulo 1, es donde se desarrolla la motivación, objetivos para llevar a cabo este proyecto. En el capítulo 2 se explica detalladamente en qué se ha basado el Juego educativo con competición síncrona utilizando J2EE: la historia de los videojuegos, la importancia de la educación a través de ellos, cómo son los juegos tipo Quiz y el resultado de introducir competición en el juego.

En el capítulo 3, se detallan todas las herramientas y aplicaciones empleadas en el proyecto, donde se explica para qué sirven y como se utilizan.

El capítulo 4 define todo el proceso de planificación del proyecto en sí. El tiempo de realización dividido por tareas, con un diagrama, y el presupuesto de la realización.

A partir del capítulo 5 comienza a explicarse el proyecto con más detalle, en este capítulo se observan los requisitos funcionales y no funcionales con los caso de uso de la aplicación a desarrollar.

En el capítulo 6 se explica el diseño y el patrón que ha seguido el juego y la programación que se ha llevado a cabo, explicando además las clases de Java y las tablas de la base de datos.

El capítulo 7 muestra explicación de la implementación que se ha llevado a cabo, el funcionamiento de la aplicación y el resultado final.

Finalmente, en el capítulo 8 se detallan las conclusiones finales, los problemas y soluciones a la hora de programar el juego y trabajos futuros.

La bibliografía en la que se apoya este proyecto se sitúa al final de la memoria.

2. Estado del arte

2.1. La importancia del juego en el desarrollo humano

La idea de desarrollar un juego educativo tiene una gran motivación: Contribuir al aprendizaje de modo que el alumno aprenda de una forma agradable y amena y conseguir al mismo tiempo que aprenda mejor cada uno de los conceptos.

Hay multitud de investigaciones respecto a la relación entre juegos y educación, pero se resumen en dos líneas claramente definidas: Investigaciones referidas al uso y efecto de los juegos en un entorno educativo e investigaciones sobre las cualidades educativas sobre el aprendizaje de nuevos conocimientos. Esta segunda línea de investigación es la principal a destacar y es donde encontramos numerosos estudios como los de Mandinacht en 1987, White en 1984, Okagaki y Frensch en 1994, que demuestran que los juegos desarrollan el valor cognitivo del individuo. Numerosos expertos defienden que la práctica de determinados juegos favorecen la atención y memoria, como los franceses Diberdier (1998), Gabriel (1994), Perriault (1996) y Lefrance (1995) o los españoles Estallo (1994, 1995), Bartolomé (1998), Calvo (2000), Gros (1997, 2000), Etxebarría (1998) y Marqués (2000). Según Estallo, los videojuegos pueden contribuir enormemente al desarrollo tanto emocional como intelectual de los adolescentes. Estallo defiende que "*los jugadores de videojuegos suelen ser sujetos de mayor nivel intelectual que sus compañeros no jugadores*".

La realidad es que los videojuegos orientados a la educación han tenido menos repercusión en el aula de lo que se pretende, ya que pueden tener menos

poder de motivación que el resto de videojuegos. Por este motivo numerosos estudiosos han dedicado su esfuerzo en hacerlos más estimulantes para el alumnado. Estos autores son Fröhlich, Ramseier y Walter (1998) y fueron de los primeros investigadores que plantearon la necesidad de incorporar los videojuegos al ámbito educativo.

Julián Pindado (Pindado, 2005) en su artículo Las posibilidades educativas de los videojuegos afirma que:

> *"Los videojuegos aparecen señalados en su interés educativo como medio de comunicación de determinadas ideas, valores o actitudes. Los trabajos en los que se analiza esta posibilidad extraen valoraciones muy positivas, dado el interés y la motivación que suscitan estos juegos entre toda la población (incluso entre los adultos). En general las experiencias recogidas permiten afirmar que las nuevas tecnologías interactivas y multimedia -y, en concreto, los videojuegos- poseen un enorme potencial para la transmisión de valores y actitudes entre niños y adolescentes, por lo que cabe considerarlos como un medio educativo innovador y en alza."*

Charo Fuentes Nevado (Fuentes, 2008) en su artículo "El componente lúdico en las clases de ELE (Español Lengua Extranjera)" defiende que las actividades lúdicas llevadas al aula proporcionan un ambiente propicio para que el proceso de enseñanza sea ameno y al mismo tiempo efectivo.

El profesorado ha descubierto que "se aprende jugando en clase" y que el resultado de jugar da lugar a que el aprendizaje de los contenidos para el alumno sea más fácil. A su vez, el uso de juegos en el aula hace que se potencie la interacción y la cohesión grupal. Los docentes adoptan esta estrategia de aprendizaje por estos motivos.

"*Podemos afirmar que las actividades lúdicas satisfacen el deseo de diversión que todos sentimos*" (Fuentes, 2008).

2.2. Historia de los videojuegos.

2.2.1. Introducción

Se puede estudiar el mundo del videojuego desde un punto de vista sociológico, tecnológico, industrial o económico.

Hay varias vertientes, una de ellas fomenta jugar de forma aislada, que resulta estar enfrentada a la idea de relacionarse entre más personas; En definitiva jugar es totalmente libre dada la gran variedad de modalidades ya que los videojuegos tienen la cualidad de poder jugarse desde cualquier entorno, dependiendo del dispositivo.

Al ser un tema realmente moderno, aún no está concluido el estudio sobre la repercusión de jugar individual y colectivamente.

Desde el punto de vista tecnológico para la creación de un juego participan varias disciplinas, principalmente la informática, diseño gráfico, efectos de sonido, música, etc. El punto de vista económico y el industrial, van de la mano y ofrecen multitud de trabajo en numerosas disciplinas por todo el mundo. La industria del juego ha generado más dinero y ha superado en la década de los 2.000 a la industria cinematográfica que ha crecido enormemente gracias al desarrollo de la informática, a la mayor capacidad de procesamiento y a un mejor diseño gráfico. Incluso se han generado multitud de videojuegos relacionados con las películas.

2.2.2. Recorrido histórico del videojuego

La historia de los videojuegos está totalmente relacionada a la vida de las computadoras puesto que ambas dos las desarrollaron los mismos ingenieros. Todo comenzó cuando estos ingenieros diseñaron juegos para la diversión y entretenimiento personal, sirviendo de ocio y relax, sin pensar en fines comerciales.

Más adelante se vio que la comercialización de estos juegos podría ser una buena fuente de ingresos.

En 1947 los americanos Thomas T Goldsmith y Estle Ray Mann simularon una pantalla de radar. Probablemente fueron los precursores del videojuego.

En 1952 se creó el primer juego por ingenio del americano Alexander Sandy Douglas. Este juego, creado con una computadora EDSAC, era una versión del tres en raya "Nought and Crosses" o "OXO",

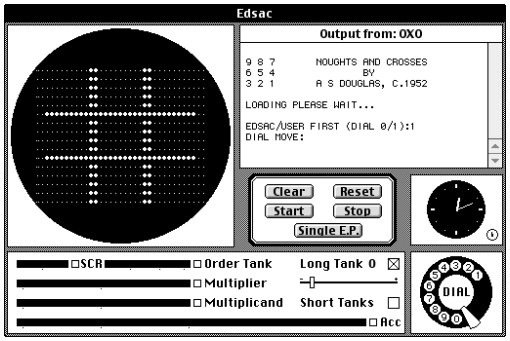
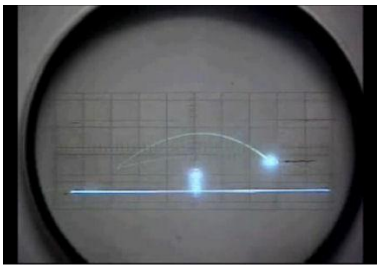


FIGURA 1: Nought and Crosses (OXO)



En 1958 el americano William Higinbotham proyecta un juego llamado "Tennis for Two" que más tarde será el exitoso "Pong" ya que Atari lo comercializó con ese nombre. Para este juego utiliza un osciloscopio de laboratorio.

FIGURA 2: Tennis for two

En 1962 el americano Steve Russel creó el videojuego Spacewar, usando gráficos vectoriales. Fue famoso en el ambiente universitario.

FIGURA 3: Pong

En 1972 fue creado el primer sistema de videojuegos doméstico con juegos pregrabados en su interior que se conectaba a la televisión.

El auge de los videojuegos llegó con la máquina recreativa Pong que se situaba en lugares públicos, bares, aeropuertos,... Fue diseñado por Al Alcorn para Atari. En 1975 Bill Gates y Paul Allen crearon Microsotf y participaron en el negocio de máquinas recreativas.

En los años 80 tienen un gran éxito estas máquinas llevando consigo la popularización de los juegos: "Destruction Derby", "Death Race", "Space Invaders", "Asteroids", "Pac Man" (comecocos), "Donkeykong","Pole Position" y "Zaxxon". Comenzaron a aparecer los primeros ordenadores para el hogar: "Commodore 64" y el "ZXSpectrum" en los cuales más tarde se adaptaron algunos de estos juegos.

2.2.3. Evolución de las videoconsolas

Hay un rápido crecimiento de esta industria y en pocos años, dado el gran éxito, hay grandes avances. En esta misma década de los 80 emerge la videoconsola:

- N.E.S (Nintendo Entertainment System). Cabe destacar los juegos más populares: En 1983, tuvo gran éxito ~~de~~ "Popeye". Nace el famoso personaje Mario Bros con el juego "Super Mario Bros", que era el primer videojuego que tenía un objetivo final. En 1987 aparece el juego "Tetris" en Europa, fue de los más vendidos.

- Mega Drive (SEGA) aparece a principios de los 90 con los juegos "Dragón Ball" o "Fifa 98"

- S.N.E.S aparece con los juegos "Super Mario Kart" o "La leyenda de Zelda".

- Game Boy, primera videoconsola portátil que incluía el famoso "Tetris".

FIGURA 4: Primera Game Boy

Aparecen ordenadores personales con monitor monocromático y sistema operativo MS-DOS. Más tarde se incorpora el soporte CD-ROM que sustituye a los anteriores soportes de cartuchos, dejando así que la evolución de esta industria sea más rápida y más efectiva.

En 1991 Aparece otro famoso personaje que revoluciona el panorama tecnológico: "Sonic". Su juego es mucho más avanzado que los anteriores respecto a

velocidad, gráficos y complejidad de la lógica del juego. En este mismo año tenemos por primera vez un videojuego en 3 dimensiones “Wolfenstein 3D”, aunque dos años más tarde tuvo mayor éxito el juego en 3D “Doom”.

Sigue evolucionando la industria con mejoras en la calidad del juego y velocidad del juego. Esto hizo que las máquinas recreativas perdieran el interés que tuvieron años atrás, tornándose hacia las nuevas videoconsolas para el hogar:

- Play Station 1 de Sony, con su primera versión tuvieron gran acogida los juegos “Final Fantasy VII”, “Resident Evil” o “Metal Gear Solid”.
- Sega Saturn
- Nintendo 64
- Xbox, la crea Microsoft en el siglo XXI
- Play Station 2 ₱pasó a ser número uno en ventas.
- Game Boy Advance, creada por Nintendo con juegos de gran éxito como “Final Fantasy” y “Halo” juego reconocido por revistas reconocidas como el mejor juego creado en la historia.

Dada la gran repercusión de los videojuegos en la sociedad y el boom de creación de miles de juegos nuevos, hubo una necesidad de clasificación y creación de normativas dentro de esta industria tan presente, por lo que en 2003 la Federación de Software Interactivo de Europa (ISFE) crea un método con unos símbolos sencillos para clasificar los juegos según sus características como la violencia, sexo, racismo, lenguaje grosero, etc.…

FIGURA 5: Símbolos para la clasificación de juegos por PEGI

La creación de más videoconsolas para el hogar/portátiles continuó su auge:

- Game Boy Advance SP (portátil) en 2004. Primera con pantalla táctil.
- Nintendo DS (portátil), con micrófono incorporado y conexión wifi.
- PSP (portátil), lanzada por Sony. Juegos que tuvieron gran éxito fueron "FIFA 2006" y "Pro Evolution Soccer 5".
- Xbox360 en 2005, consola de 7ª generación que incluye además un lector de películas HD-DVD (Disco Versátil Digital de Alta Densidad). Juegos de éxito y de muy alta calidad fueron : "Perfect Dark Zero", "Call of Duty 3"
- PlayStation3 en 2006, incorpora, además de las mejoras, un lector de discos ópticos de H.D. llamados Blue-Ray Disk y un servicio de videojuegos en línea
- Wii en 2006, de Nintendo. Se diferenciaba del resto por su uso más interactivo, y sin cables.

- PSP-GO (Sony) Versión de mejora de la previa PSP.

- DSi XL (Nintendo), versión mejorada de la DS.

Es tal la acogida de los videojuegos en 2010, que para expresar las ganancias de esta industria, simplemente con el juego "Call of Duty: Black Ops" se obtuvo un beneficio de más de 1.000 millones de dólares, suponiendo este dato un resultado mucho mayor que los ingresos obtenidos por películas cinematográficas de gran coste. Esto en parte se debe a que el nivel de calidad y de realismo del videojuego es similar al cinematográfico.

En 2012 llegan las videoconsolas de 8ª generación, que usan internet como base fundamental:

- PlayStation 4 en 2013, que además de videoconsola tiene funcionalidades para acceder a internet, reproducción de películas, series,…

- WII U en 2013, lanzada por Nintendo

- Xbox One en 2013, incluye reconocimiento por voz desde los mandos.

2.3. El juego a través de la Historia como proceso educativo.

2.3.1. Introducción

Desde la antigüedad el ser humano ha considerado que el juego es una herramienta válida para la educación. En principio los juegos solo tenían como fin único, el entretenimiento y la diversión, pero hubo un cambio al respecto:

Se demostró que el uso de juegos ofrecía grandes ventajas para la educación, gracias a aportaciones de grandes pedagogos, psicólogos, sociólogos y, antropólogos.

Tanto es así que numerosos estudios han ido demostrando a lo largo de la Historia que el uso de juegos es esencial para el desarrollo de los niños; les aporta habilidades para liberar ansiedades y tensiones, pensamiento creativo, solución de problemas, habilidad para usar herramientas, desarrollo del lenguaje y capacidad para adquirir nuevos conocimientos.

2.3.1. Autores que respaldan el juego como herramienta educativa

Uno de los primeros filósofos de la Historia que defiende el valor práctico del juego fue Platón, que expone la idea de la utilidad de educar jugando, dado lo placentero del juego para niños y jóvenes (en su obra *Leyes, Introducción, traducción y notas de Francisco Lisi,* Madrid, Gredos, (Tomo VII)). Junto a él, Aristóteles, en el libro *Cosas que debe comprender la educación* reconoce que el juego es esencial como parte de la formación y apunta que el juego es un remedio saludable a tener en cuenta como un fin por el placer y el descanso que produce. En su libro *La Ética* denomina la eutrapelia o "virtud del juego" como un buen aprovechamiento del descanso.

En el siglo XVII humanista y pedagogo Juan Amós Comenio, considerado el padre de la pedagogía moderna, propone considerar el juego como motor de la enseñanza y que el colegio sea un lugar donde aprender jugando. Escribió Didáctica Magna con la que impulsó la creación de la Ciencia de la Educación, novedoso método de enseñanza centrado en el alumno. Fue pionero en un proyecto científico del juego en la vida educativa.

Rousseau, filósofo del siglo XVIII, es considerado filósofo de la educación, revisa la pedagogía tradicional con la perspectiva del pensamiento ilustrado, decía que a través de la libertad y espontaneidad del juego los niños podían aprender. En *Emilio* trata el tema del sistema educativo, definiendo las bases de una pedagogía que debe ser renovada y adecuada a los nuevos tiempos, proponiendo estimular en el niño el deseo de aprender

También, en el siglo XVIII, Pestalozzi realizó unas aportaciones pedagógicas que obtuvieron gran repercusión, porque muchos de los fundamentos de la educación actual se asientan en su método de enseñanza, obteniendo grandes avances al implantarse en toda Europa. Su sistema surge del principio de la intuición y fomentó la idea utilizar la actividad y el juego para el desarrollo de los niños. Mediante la observación y la práctica se va guiando al niño que, mediante el juego, se ejercita en experiencias y actividades. Pestalozzi hizo grandes aportaciones a la educación entre otros planteamientos educativos del juego (fue el precursor de la pedagogía moderna).

Ya, en el siglo XIX, padre de la psicología progresista y alma mater de la Escuela Nueva, Dewey, es un filósofo, psicólogo y pedagogo que se opone a la escuela tradicional. La escuela para él debe ser un proceso activo y los profesores deben guiar a los auténticos protagonistas del aprendizaje: los alumnos. La pedagogía actual es más funcional y dinámica, que considera que el juego es un instrumento de motivación, Dewey considera que el juego es un recurso de apoyo en el aprendizaje.

Johan Huizinga, ya en el siglo XX, hizo la obra magistral *Homo Ludens* donde explica que desde un punto de vista antropológico el juego es una función humana. El historiador holandés la publicó en 1938 y con esta obra obtuvo mucha relevancia al destacar la importancia que tiene en la sociedad y la cultura el juego. Concibe el juego como un fenómeno cultural, Huizinga afirma "la cultura no comienza como juego ni se origina del juego, sino que es, más bien juego. El fundamento antitético y agonal de la cultura se nos ofrece ya en el juego, que es más viejo que toda cultura".

Vygotsky es un importante teórico de la psicología del desarrollo del siglo XX, está considerado como el precursor del constructivismo social. Estudió al niño en el entorno escolar, destacando el juego como una actividad fundamental para él y su influencia en la educación de los niños, su desarrollo de la percepción, la atención y el lenguaje por él los pedagogos soviéticos incorporaron juegos al currículo preescolar y escolar y demostraron lo beneficios educativos del juego. Gracias a él los pedagogos soviéticos incorporaron juegos al currículo preescolar y escolar y demostraron lo beneficios educativos del juego.

Para Piaget, siglo XX, lo más importante era el proceso cognitivo que podía ofrecer el educar a través del juego, analiza detalladamente la concepción del juego. Maite Garaigordobil analiza las aportaciones de Piaget posicionando el juego infantil en un momento en el que el niño asimila y relaciona lo que percibe con sus vivencias anteriores y lo adapta a sus necesidades.

Más recientemente el sociólogo francés Roger Caillois con su obra publicada en 1958 *Los juegos y los hombres*, clasificó los juegos en 4 categorías: competencia, azar, simulacro y vértigo.

La profesora Catherine Garvey ha influido en autores posteriores. Escribió El juego infantil, donde señala *"el juego es placentero y divertido. Aun cuando no vaya acompañado de regocijo, es evaluado positivamente por parte del jugador*".

Otros importantes estudiosos fueron Fröebel con La educación del hombre, Decroly con El juego educativo, Leif y Brunelle con La verdadera naturaleza del juego, Elkonin con Psicología del juego, Bruner con Acción, pensamiento y lenguaje y Ortega con Jugar y aprender.

2.3.2. Resultado del uso de juegos educativos

En 2009 el Parlamento Europeo reconoció que el uso adecuado de videojuegos puede contribuir a "*estimular el aprendizaje de hechos y aptitudes como la reflexión estratégica, la creatividad, la cooperación y el sentido de innovación*" por lo que se lleva a cabo la educación a través de videojuegos.

Los datos recogidos de un estudio[1] realizado por aDeSe (La Asociación Española de Distribuidores y Editores de Software de Entretenimiento) en 2012 sobre la introducción de los videojuegos en el sistema español de enseñanza primaria muestran que tanto padres como profesores señalan que el videojuego es una herramienta de alto valor pedagógico.

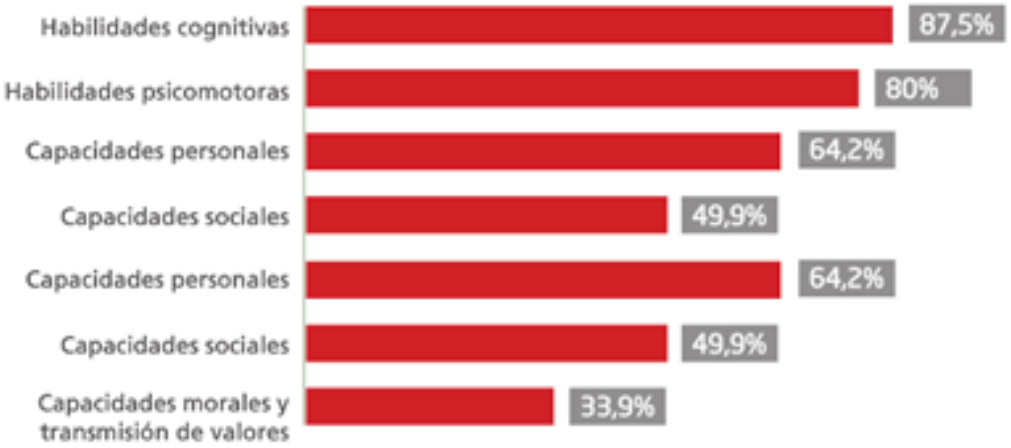
"El 92% de los padres consultados aprueba su uso como herramienta educativa y el 79% considera que pueden ser una herramienta eficaz para la transmisión de conocimiento". Los padres y profesores encuestados están de acuerdo con que la educación a través del uso de videojuegos desarrolla las habilidades cognitivas al 88%, psicomotoras al 87%, personales al 73%, sociales al 65% y morales al 40%.

[1] Datos obtenidos por aDeSe (La Asociación Española de Distribuidores y Editores de Software

El 80% de los profesores considera los videojuegos una herramienta eficaz y el 87% afirma que favorecen las habilidades cognitivas

En general, el 79% de los profesores, tanto de los que los han utilizado como los que aún no, aprueban su uso en el aula y el 79% los considera una herramienta eficaz, principalmente en asignaturas como Conocimiento de Medio (71%), Matemáticas (68%), Lengua extranjera (64%) o Lengua española (60%).

Asimismo, en su opinión, los videojuegos favorecen el desarrollo de habilidades cognitivas, psicomotoras (destreza visual y discriminación perceptiva, coordinación espacial y lateralidad…) o capacidades personales (autonomía, autocontrol, autoestima, creatividad…).



Habilidades Cognitivas señaladas - Bastante o totalmente de acuerdo

- ✓ Mejorar la concentración y focalización de la atención **(73%)**
- ✓ Favorecer el desarrollo de habilidades estratégicas y resolución de problemas **(66%)**
- ✓ Favorecer el desarrollo de la memoria **(66%)**
- ✓ Favorecer la capacidad analítica, estratégica y de planificación de la acción **(66%)**
- ✓ Mejorar la capacidad de comprensión, el desarrollo del pensamiento lógico y sistemático **(63%)**

En definitiva, sólo 2 de cada 10 profesores encuentran todavía objeciones o se muestran reacios a su introducción, principalmente por la falta de información o de planificación y apoyo en el centro.

FIGURA 6: Resultados 1, Estudio aDeSe 2012

En cuanto a las motivaciones para introducir videojuegos en las aulas, los profesores consultados con experiencia en su uso destacan las siguientes:

FIGURA 7: Motivaciones, Estudio aDeSe 2012

2.3.3. Competición: Motivación y emociones

Un elemento que puede mejorar la motivación en los estudiantes es introducir juegos de competición utilizando sistemas tecnológicos para que no se provoquen emociones negativas.

Si los estudiantes están bien motivados, estarán concentrados en sus actividades y tienden a estudiar más tiempo.

A los estudiantes les gusta interactuar con los compañeros comparando sus progresos mediante la competición, pero según sea su diseño, si no obtienen buenos resultados pueden sentir emociones negativas como son la frustración y la decepción.

Para aumentar la motivación es muy útil que los estudiantes compitan en varias rondas, ganando puntos al resolver correctamente los problemas y porque además, lo hagan mejor que sus compañeros.

Para reducir las emociones negativas, se puede evitar que el juego sea eliminatorio, para que todos puedan competir el mismo número de rondas.

No todos los autores están de acuerdo en si la competición aumenta la motivación. Varios indican que la competición puede generar decepción y falta de confianza e interés.

Para mitigar estos problemas, se han proyectado varios sistemas, como los que incluyen mascotas virtuales que representan a los estudiantes; los que utilizan crucigramas que combinan cooperación y competición. Estos son: El sistema Joyce, que utiliza una interfaz gráfica con un tablero en el que se avanza

resolviendo preguntas; el sistema Quest, basado en el desafío, determinando las puntuaciones de los estudiantes por su rapidez y el orden en el que respondieron; el sistema EOT, con la táctica de igualdad de oportunidades que toma en cuenta el tiempo que se emplea para resolver el problema o el juego AnswerMatching, implementación del EOT, para enseñar aritmética.

Cada vez se utilizan más los sistemas de tutoría inteligentes (STI) en la educación. Una STI es un sistema que ayuda a los estudiantes y docentes en el proceso de aprendizaje al que se accede a través de la Web o como una aplicación independiente. Una STI se suele dividir en cuatro módulos (Wenger, 1987): el módulo de interfaz (profesores y estudiantes interactúan en el sistema), el módulo de expertos (contenidos específicos), el módulo pedagógico (las diferentes estrategias de la enseñanza), y el modelo de usuario (algunos datos que representan los estudiantes y los maestros en el sistema, por lo que los diferentes contenidos y estrategias se puede adaptar a los diferentes perfiles).

Los STI son aplicaciones que deben permitir y gestionar el aprendizaje y tomar decisiones acerca de temas como los materiales que se muestran en cada momento para diferentes perfiles, o los aspectos motivacionales y emocionales.

El éxito de la introducción de herramientas competitivas en el aprendizaje depende de las estrategias competitivas específicas aplicadas.

Aunque la competición y los juegos son conceptos diferentes, tienen una relación y la mayoría de los juegos presentan un cierto grado de competencia. Una gran cantidad de juegos en la educación también tienen una parte competitiva porque proporcionan un componente de motivación y disfrute que pueden mejorar el proceso de aprendizaje.

2.4. Juegos tipo Quiz.

El término Quiz es utilizado para definir un tipo de juego de mente, concurso o competición de preguntas donde un jugador o varios tratan de responder correctamente una serie de preguntas.

Cabe destacar que los juegos de preguntas y respuestas suscitan una gran adicción; tanto en juegos de mesa, como en juegos para el ordenador, aplicaciones para móvil o tablet y en concursos televisivos.

Tras el repaso histórico sobre los juegos, la clave de la enseñanza a través de los juegos es que éstos generen una gran adicción, para lo que han de ser sencillos y rápidos a la hora de jugar. Este es el motivo principal por el que el juego a desarrollar en este proyecto educativo sea un juego que tenga el formato "Quiz" (preguntas y respuestas).

Uno de los juegos educativos tipo Quiz más famosos y adictivos, en el que se basa este proyecto es:

- **Trivial Pursuit:** Sus innumerables preguntas y ediciones temáticas y renovadas después de años de reinado, han entretenido a muchas generaciones y han permitido que este juego haya llegado a configurar un género en sí mismo. Ha tenido mucha repercusión este formato, por lo que hay cientos de imitaciones para todo tipo de soportes.

FIGURA 8: Trivial Pursuit

El Trivial Pursuit consiste en tirar los dados en un tablero y según el color de la casilla de destino se formula una pregunta de un tema en concreto. Cada color es un tema siendo así azul: geografía; rosa: espectáculos; amarillo: historia; marrón: arte y literatura; verde: ciencias y naturaleza; naranja: deportes y pasatiempos.

2.4.1. Concursos televisivos de gran éxito

Numerosos concursos de gran audiencia en la televisión han utilizado el recurso de juego tipo Quiz. A continuación se analizan varios concursos de gran acogida en España de los últimos años para extraer las claves de su éxito y adicción:

- **Quién quiere ser millonario:**

FIGURA 9: Película Slumdog Millonaire

FIGURA 10: Versión española: 50x15 ¿Quién quiere ser millonario?

Un Juego originario del Reino Unido (en inglés “Who Wants to Be a Millionaire?”), es un Quiz Show de preguntas y respuestas en el que se ofrecía un millón de libras. Este nuevo tipo de concurso revolucionó por completo el formato de estos juegos televisivos. Tuvo una gran repercusión internacional, se emitió en más de 100 países.

El juego consiste en responder correctamente a 50 preguntas, cada una de ellas tiene 4 respuestas posibles y sólo una es correcta.

El éxito de este juego tipo Quiz fue tal, que se estrenó una película indio-británica basada en este concurso: *Slumdog Millionaire* (¿Quién quiere ser millonario?) dirigida por Danny Boyle premiada con 8 Premios Óscar.

- **Saber y ganar:**

Saber y Ganar es un concurso que ha logrado mantenerse en antena 10 años de la mano de su presentador Jordi Hurtado. Se ha convertido en el programa diario con más años de emisión en la historia de la televisión en España.

Su objetivo principal es la divulgación de la cultura de una forma amena. En el programa tres concursantes compiten tratando de eliminar a uno de ellos a base de responder a preguntas y demostrar sus conocimientos y agilidad mental. Finalmente quedará solo un concursante que si supera una prueba final llamada "la parte por el todo" gana el concurso.

FIGURA 11: Programa SABER y GANAR

- **Un, dos, tres... ¡responda otra vez!:**

Fue un programa de TVE creado en 1972 por Narciso "Chicho" Ibáñez Serrador, y que consta de 10 temporadas.

FIGURA 12: un, dos, tres... ¡responda otra vez! logotipo

El programa siempre ha sido emitido por TVE-1, y en la actualidad es considerado como uno de los clásicos de la Televisión Española. Supuso una revolución en la forma de hacer concursos en España.

Constaba de tres fases diferenciadas: Fusionaba la cultura en la primera fase de preguntas y respuestas, las actividades físicas en la segunda fase y como tercera fase la suerte, intuición y psicología. Anteriormente solo había concursos de uno de los tres tipos mencionados, de ahí la idea del realizador de fusionar los tres y darle el nombre al programa de "un, dos, tres…"

- **El rival más débil**

Es un concurso que se ha televisado en multitud de países en la primera década del 2000. El programa consiste en que los concursantes responden a las preguntas de la presentadora, que por lo general tiene un trato frío y poco amable, criticando y riéndose de los fallos de los concursantes, dejándolos en evidencia; característica principal del programa que hace que los concursantes se pongan nerviosos.

El procedimiento del programa consiste en tener que eliminar a uno de los concursantes entre los ocho que hay con el criterio de quien ha sido el peor. Así sucesivamente hasta que quedan dos y luchan entre ellos con 5 preguntas.

FIGURA 13: El rival más débil, Nuria González

- **Atrapa un millón:**

A una pareja de concursantes se les da un millón de euros en billetes al comienzo del juego. Los concursantes deben responder a ocho preguntas repartiendo el dinero entre las opciones con el fin de mantener la mayor cantidad hasta el final. Las opciones erróneas eliminan el dinero.

FIGURA 14: Atrapa un millón, Carlos Sobera 2011

- **¡Boom!**

Fue el concurso televisivo que sustituyó a Atrapa un millón. Es un juego con más tensión respecto a otros similares ya que el concursante debe ir cortando los cables de una bomba. Estos cables son las opciones posibles a la pregunta dada, donde solo una de ellas es correcta. Debe cortar las respuestas incorrectas y dejar solo la correcta. En caso de errar, la bomba explota fuertemente con polvo de colores.

FIGURA 15: ¡Boom!, Juanra Bonet 2014

- **¡Ahora caigo!**

En este juego un concursante principal reta a sus 10 oponentes uno a uno y se bate en duelo de preguntas. Al concursante que no adivina en un tiempo concreto se le abre una trampilla y se lo traga la tierra literalmente. El papel del Showman es importante.

FIGURA 16:¡Ahora caigo!, Arturo Valls 2015

2.4.2 Videojuegos tipo Show Quiz

- **Buzz!**

Es un videojuego que en 2006 ganó el premio el BAFTA premio concedido por Best Casual and Social game(mejor juego casual y social). Desarrollado para las consolas PlayStation 2, PlayStation 3 y PlayStation Portable. Consiste en un concurso ficticio con una serie de preguntas con cuatro opciones posibles cada una de un color y se responde con los periféricos especiales llamados buzzers. El buzzer es el pulsador que tiene cuatro botones de colores (azul, naranja, verde y amarillo) para responder y un botón rojo: el botón BUZZ!

FIGURA 17: Juego Buzz Quiz World

El show usa un formato multironda como la mayoría de juegos de la saga con ocho rondas individuales. Y como en todo concurso no falta el presentador Buzz (doblado por Jason Donovan en la versión inglesa).

- **Scene it!**

Es un juego tipo "Buzz" creado únicamente la consola Xbox 360 traído del juego de mesa, pero concebido para la temática del cine. Todas las preguntas son cinematográficas y tiene una estructura tipo trivial.

FIGURA 18: Juego Scene it?

3. Herramientas y tecnologías aplicadas

3.1. NetBeans IDE .

NetBeans es un entorno de desarrollo integrado libre y gratuito, hecho principalmente para el lenguaje de programación Java.

Para programar esta aplicación en NetBeans hemos utilizado el lenguaje para el que mejor está desarrollado, Java.

Según la web oficial de NetBeans www.netbeans.org:

> *"NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios (¡y creciendo!) en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos."*
>
> *"NetBeans IDE es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación"*

3.2. El lenguaje de programación JAVA.

El origen del leguaje de programación Java data de los años 90 y se llamaba Oak. Fue creado por James Gosling y desarrollado por Sun Microsystems con la motivación de programar electrodomésticos y aparatos electrónicos de consumo, y así poder cambiar de dispositivo sin tener que reescribir el código. Éste era el mayor problema encontrado con otros lenguajes de programación como C++, que al compilarlo genera un fichero ejecutable que solo es útil en la plataforma en la que se compiló. En 1994, Sun quiso expandir su nuevo lenguaje de programación a Internet, el cual comenzaba a tener éxito, y se propusieron tener un lenguaje universal que permitiera crear aplicaciones que se ejecutaran en cualquier ordenador de Internet.

Como curiosidad, se le cambió el nombre de Oak a Java. Lo adquirió a partir de 1995 y se debe a un tipo de café de EEUU. De ahí su famoso logotipo.

Rápidamente tienen gran éxito los applets (programas en Java que se ejecutan en el contexto de una página web en cualquier ordenador independientemente de su Sistema Operativo y de la arquitectura de su procesador)

Las principales características son que es un lenguaje **sencillo**: sin punteros, sin sobrecarga de operadores, no permite la herencia múltiple y la gestión de la memoria dinámica se hace automáticamente; Es un leguaje **orientado a objetos** puro; Es **multiplataforma** eliminando así la dependencia de la máquina: Un programa Java (*.java) al compilar se obtiene un código llamado bytecode que es el *.class. Este bytecode lo interpreta el JVM (Java Virtual Machine) o JRE (Java Runtime Enviroment). El JRE está a disposición de todos los usuarios. Se consigue así que un archivo *.class se pueda ejecutar en cualquier máquina; Es **robusto**: Java trata de

reducir al máximo los errores en compilación y ejecución; Es **seguro**; **Multitarea** y **dinámico.**

En 1999 Sun Microsystems lanza la plataforma J2EE para solucionar los problemas encontrados a la hora de desarrollar aplicaciones web, que a continuación se detalla.

3.2.1. Arquitectura J2EE

Es preciso hablar de la arquitectura J2EE puesto que es el pilar fundamental del desarrollo de este proyecto.

J2EE es un conjunto de especificaciones de API's Java para el desarrollo de aplicaciones empresariales. Unas de las principales tecnologías proporcionadas por J2EE son las tecnologías Web (J2EE), donde las API's son básicamente Servlets y JSPs y permiten implementar la interfaz gráfica de una aplicación web.

Esta especificación nace de los problemas generados de las arquitecturas de dos capas (cliente – servidor), ya que si surgían cambios en la implementación del modelo era necesaria la recompilación de toda la aplicación y la reinstalación en todos los clientes; Por lo que la solución a estos problemas es tener un servidor intermedio (tres capas), de este modo, un cambio en la implementación del modelo solo afecta al servidor mientras que los clientes solo tienen la interfaz gráfica.

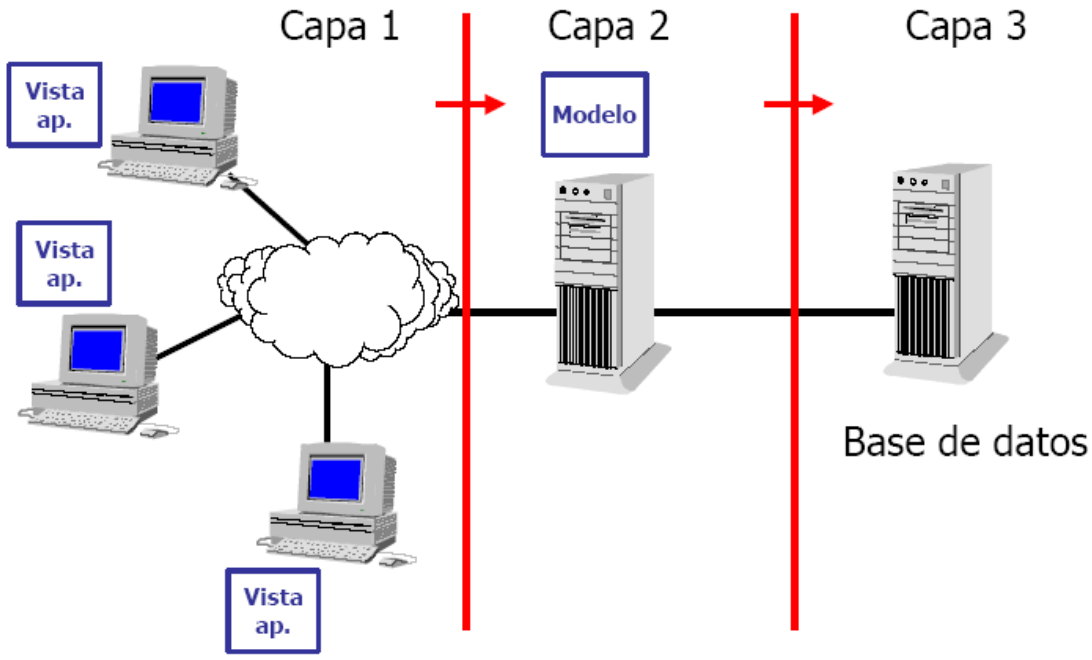


FIGURA 19: Arquitectura de 3 capas J2EE - 1

Por lo tanto, para una aplicación web es más eficiente usar una arquitectura en tres capas. De este modo, la comunicación entre el modelo y la interfaz gráfica es local. Además se puede dividir el desarrollo de la aplicación asignando a diferentes expertos en cada nivel y la utilización de recursos es más eficiente. Cierto es que aumenta la dificultad de desarrollo de la aplicación al tener que integrarse correctamente cada nivel, pero finalmente sigue siendo más eficiente.

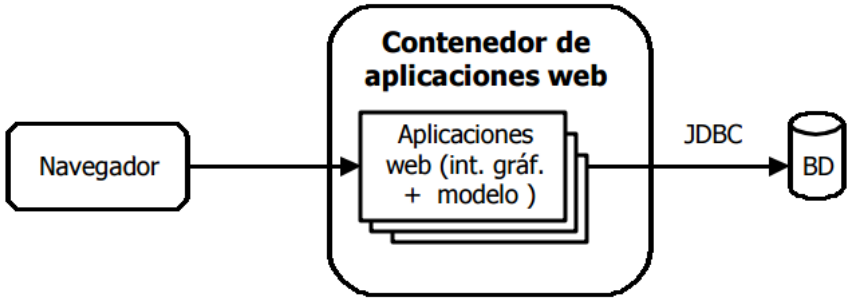


FIGURA 20: Arquitectura de 3 capas J2EE- 2

Esta arquitectura tiene bien separadas las tres capas entre el Navegador, contenedor de aplicaciones web y base de datos como indica la figura 20.

Los contenedores proveen un entorno de ejecución de Java y a la vez ofrecen los servicios de seguridad, transacciones, administración del ciclo de vida, caching, persistencia y comunicación en la red. A estos servicios se accede a través de API's con lo que se mejora el tiempo de desarrollo y se simplifica el mantenimiento.

Existen dos tipos de contenedores: Los contenedores web, que almacenan **Servlets** y **JSP** (componentes de presentación) y los contenedores EJB que administran la ejecución EJB (Enterprise Java Bean).

Es preciso que para esta arquitectura en 3 capas se utilice el patrón MVC (Modelo – Vista – Controlador) puesto que separa la lógica de negocio de la interfaz de usuario. De este modo, ambos pueden ser modificados sin perjudicar al otro en su funcionamiento, lo que hace de esta aplicación una aplicación muy flexible y reutilizable para el desarrollador, tal como requiere la arquitectura J2EE. En el apartado 5.1 se explica más detalladamente respecto a este proyecto en concreto. En definitiva la arquitectura J2EE y el modelo MVC están totalmente correlacionados.

3.2.2. ¿Qué son los Servlets Java?

Son programas en Java destinados a ejecutarse en el servidor. Son clases que derivan de los anteriores applets, y su utilización es necesaria para extender las aplicaciones que están alojadas en servidores web. Es como si fuera un applet que se ejecuta en el servidor, en lugar del navegador. Los Servlets generan páginas webs de forma dinámica a partir de los parámetros de petición que le envíe el navegador web.

A continuación se explicará cómo crear un Servlet con un ejemplo de este proyecto `UsuarioServlet.java`

```
public class UsuarioServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws IOException, ServletException {
            doPost(request, response);
        }


    public void doPost(HttpServletRequest request, HttpServletResponse res)
        throws IOException, ServletException  {
        String AdminUser="Admin"; //Nommbre de usuario del Administrador

        HttpSession session = request.getSession(true);
        res.setContentType("text/html");
        String  user = request.getParameter("user");
        String  pass = request.getParameter("pass");
        session.setAttribute("user", user);

        BaseDeDatos bbdd = new BaseDeDatos();
        bbdd.OpenConexionMySQL();
        if(bbdd.coincide(user, pass)){
            if(user.equals(AdminUser)){
                request.getRequestDispatcher("PaginaAdministrador.jsp").forward(request,res);
            }
            else request.getRequestDispatcher("PaginaUsuario.jsp").forward(request,res);
            }
        else  request.getRequestDispatcher("Inicio.jsp").forward(request,res);
    }
}
```

FIGURA 21: Clase UsuarioServlet.java

El servlet hereda de una clase HttpServlet, la cual es abstracta. Esto quiere decir que ha de ser implementada como muestra la figura 21, donde lo hacen únicamente los métodos `doGet()` y `doPost()`.

Los servlets permiten gestionar elementos Http mediante las clases: HttpServletRequest que recibe la petición (ámbito de request), HttpServletResponse que genera la respuesta, HttpSession que permite crear una sesión común a un

conjunto de request (ámbito de sesión) y ServletContext que gestiona la información común a todas las peticiones realizadas sobre la aplicación.

Los métodos a implementar de la clase HttpServlet según en API de Java son:

Method Summary	
protected void	doDelete(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a DELETE request.
protected void	doGet(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a GET request.
protected void	doHead(HttpServletRequest req, HttpServletResponse resp) Receives an HTTP HEAD request from the protected service method and handles the request.
protected void	doOptions(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a OPTIONS request.
protected void	doPost(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a POST request.
protected void	doPut(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a PUT request.
protected void	doTrace(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a TRACE request.
protected long	getLastModified(HttpServletRequest req) Returns the time the HttpServletRequest object was last modified, in milliseconds since midnight January 1, 1970 GMT.
protected void	service(HttpServletRequest req, HttpServletResponse resp) Receives standard HTTP requests from the public service method and dispatches them to the do*XXX* methods defined in this class.
void	service(ServletRequest req, ServletResponse res) Dispatches client requests to the protected service method.

FIGURA 22: Métodos de HttpServlet

El funcionamiento de los dos métodos principales de un Servlet - `doGet()` y `doPost()`- se utilizan para enviar datos de formularios en el navegador hacia servidores . La solicitud GET por ejemplo podría ser:

`http://www.juegos.com?tipo=6&jugador=Juan`

En esta solicitud estamos pasando como parámetros tipo=6 y jugador=Juan. En cambio en una solicitud POST la url va limpia de parámetros:

`http://www.juegos.com`

En ambos casos, la manera habitual de pasar estos parámetros, ya sean transparentes o no al usuario, es hacerlo mediante formularios HTML.

Un ejemplo de cómo hacer una petición GET desde un formulario sería:

```
<form id="formulario" name="formulario" method="get" action="">
      Tipo:
   <input type="text" name="tipo"/>
      <br>Jugador:
    <input type="text" name="jugador"/>
      <br>
   <input type="submit" name="Envia" value="Envia"/>
</form>
```

Para hacer una petición POST, bastaría con cambiar el nombre del método de `method="get"` a `method="post"`.

Para saber qué método de los dos es más conveniente para el Servlet, es a elección del programador, pero la tendencia es usar peticiones GET para enviar pocos parámetros poco significativos y usa peticiones POST para datos de gran tamaño, formularios, contraseñas…

3.2.3. ¿Qué son los JSP's?

Un JSP (JavaServer Pages) está compuesto por código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por lo que es factible escribirlas un editor HTML/XML.

JSP puede ser visto como una abstracción de alto nivel de los servlets Java. El motor de los JSP está basado en los Servlets. Es mucho más fácil aprender a programar JSP's que Servlets, por lo que hay muchos más desarrolladores de JSP's que de Servlets.

El JSP se crea de forma similar a ASP o PHP (dos tecnologías de servidor). Son ficheros con extensión ".jsp" en los que conviven una estructura de etiquetas de html con sentencias Java.

Para incorporar sentencias java en el jsp ha de hacerse de la siguiente manera:

```
<h1 class = "H1">REGISTRO</h1>
<p align="center" class="P">Introduzca el usuario y contraseña nuevos</p>

     <% String mensaje1 = request.getParameter("mensaje1");
        if (mensaje1==null)  mensaje1 ="";  %>
     <%   String mensaje2 = request.getParameter("mensaje2");
        if (mensaje2==null) mensaje2 ="";  %>

<p class = "error" align = "center" ><%=mensaje1%></p>
<p class = "error" align = "center" ><%=mensaje2%></p>
<form  method = "post" action = "registro">
    <table align = "center" border="0" cellpadding="0" cellspacing="0" width="45%">
        <tr>
            <td width="15%">Nombre del usuario: </td>
            <td width="10%"><input type="text" size="30" name="user">
        </tr>
        <tr>
            <td width="15%">Nueva contraseña (6 a 8 dígitos): </td>
            <td width="10%"><input type="password" size="30" name="pass1"></td>
        </tr>
        <tr>
            <td width="15%">Repita contraseña (6 a 8 dígitos): </td>
            <td width="10%"><input type="password" size="30" name="pass2"></td>
        </tr>
    </table>
```

FIGURA 23: Parte del código Registro.jsp

Se observa que el código es como un XML habitual, sólo que entre los símbolos "**<%**" y "**%>**" se puede insertar código java. En el caso de la figura 24 es el código de registro de un nuevo usuario y el código java quiere imprimir los parámetros recogidos si los hubiera, sino no imprimiría nada.

También hay etiquetas pertenecientes a la especificación JSP que proporcionan una funcionalidad básicas: como **<jsp:forward>** que redirige la request a otra URL, **<jsp:include>** que incluye el texto de un fichero dentro de la página, **<jsp:plugin>** que descarga un plugin de Java (una applet o un Bean), **<jsp:useBean>**, permite manipular un Bean (si no existe, se creará el Bean), **<jsp:getProperty>**, obtiene la propiedad especificada de un bean previamente declarado y la escribe en el objeto response y **<jsp:setProperty>**, establece el valor de una propiedad de un bean previamente declarado. Estos tres últimos permiten manipular componentes JavaBean sin tener conocimientos de Java.

Para este proyecto, en los JSP's, ha tenido una gran importancia el leguaje de programación de JavaScript, para la correcta ejecución del juego.

3.2.4. ¿Qué es JavaScript?

JavaScript es un lenguaje bastante eficaz ya que es un leguaje de programación interpretad. Esto quiere decir que no es necesario compilar los programas para ejecutarlos, se pueden probar en el navegador directamente. Nada tiene que ver con el leguaje de programación de Java pese a llamarse JavaScript.

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas (páginas que incorporan efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario).

Un ejemplo del como de incluir JavaScript en JSP's (o cualquier XHTML) :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

    <head>

        <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

        <title>Ejemplo como incluir un JavaScript en el documento</title>

        <script type="text/javascript">

             alert("Un mensaje de prueba");

        </script>

    </head>

     <body>

        <p>Un párrafo de texto.</p>

    </body>

</html>
```

Lo más conveniente es introducir, mediante la etiqueta **<script>**, el código dentro de la etiqueta <head>.

Como se puede observar, la sintaxis de JavaScript es muy similar a Java y a C. Mantiene unas normas básicas: No se tienen en cuenta los espacios en blanco y las nuevas líneas igual que sucede con XHTML; se distinguen las mayúsculas y minúsculas, no se define el tipo de las variables, no es necesario terminar cada sentencia con el carácter de punto y coma (;) y se pueden incluir comentarios del mismo modo que se hace en java.

3.3. Apache Tomcat.

Apache Tomcat es un contenedor web de Servlets y JSP's. Es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. Tiene un compilador que convierte los JSP's en servlets.

Hay diferentes contenedores de Servlets. Los podemos dividir en tres grupos:

1. Contenedores de Servlets Stand-alone (Independientes). Estos son una parte integral del servidor web. Es un servidor web basado en Java

2. Contenedores de Servlets dentro-de-Proceso**.** El contenedor Servlet es una combinación de un plugin para el servidor web y una implementación de contenedor Java.

3. Contenedores de Servlets fuera-de-proceso. El contenedor Servlet es una combinación de un plugin para el servidor web y una implementación de contenedor Java que se ejecuta en una JVM fuera del servidor web.

(2001, Eloy A. Esteban)

3.4. Base de datos.

3.4.1. My SQL

MySQL es un sistema de Gestión de Bases de datos relacional creado en 1995 por la empresa MySQL AB. Está desarrollado principalmente en C y C++ y para las consultas utiliza SQL.

SQL es el lenguaje más común para construir las consultas en los sistemas de gestión bases de datos relacionales y al ser un lenguaje de "alto nivel" permite una alta productividad en la codificación y la orientación a objetos y aprovecha la flexibilidad y potencia de este tipo de sistemas.

Se obtiene como software libre bajo licencia GNU GLP aplicable en este proyecto. Aplicaciones grandes y populares como Facebook, Google, Twitter, Youtube, etc. utilizan MySQL.

Caracterísiticas principales que hacen de MySQL uno de los mejores gestores de mayor rendimienro:

- Bajo coste en requerimientos para la creación de bases de datos ya que debido a su bajo consumo puede ser ejecutado en máquinas con pocos recursos.
- Facilidad de instalación y configuración.
- Alto nivel de portabilidad entre sistemas.
- Soporta una gran variedad de sistemas operativos.
- Garantiza seguridad en la conexión y en los datos mediante un sistema flexible de contraseñas y gestión de usuarios.
- Aprovecha la potencia de los sistemas multiproceso gracias a su implementación multihilo.

- Ofrece selección de mecanismos de almacenamiento con distinta velocidad de operación, soporte físico, capacidad, distribución geográfica, transacciones, etc.

- Amplio subconjunto del lenguaje *SQL*.

- Cada base de datos posee tres archivos: uno de estructura, uno de datos y uno de índices, con hasta 32 índices por tabla.

- Replicación, transacciones y claves foráneas.

- Búsqueda e indexación de campos de texto.

- Baja probabilidad de corromper datos.

3.4.2. Heidi SQL

HeidiSQL, ha sido la herramienta utilizada para la creación de las tablas de este proyecto, ya que de una forma muy intuitiva y sencilla te facilita la creación de las mismas.

HeidiSQL, anteriormente MySQL-Front, es un interfaz libre y de código abiertopara MySQL y otras aplicaciones. Desarrollado en Delphi principalmente por Ansgar Becker MySQL. Aunque las características son suficientes para operar en la mayoría de Bases, tablas y registro de datos, se mantiene activo su desarrollo hacia la completa funcionalidad que se espera de un interfaz MySQL.

3.4.3. Wamp Server

WAMP es un sistema que combinando Windows (Sistema operativo), Apache (servidor web) MySQL (gestor de bases de datos) y PHP, Perl o Python (lenguaje de programación), permite subir páginas html a internet así como gestionar sus datos. De ahí viene su acrónimo **W**indows **A**pache **M**ySQL **P**HP.

También proporciona un entorno para el desarrollo de aplicaciones web, para lo que fue necesario en este proyecto

WampServer

3.4.4. WorkBrench MySQL

MySQL Workbench es una herramienta visual de diseño de bases de datos que integra desarrollo de software, Administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL.

Esta herramienta es muy últil para visualizar las tablas de forma gráfica.

4. Planificación

Para la realización de este proyecto y de los objetivos mencionados se ha utilizado la arquitectura J2EE y JavaScript y mySQL como herramienta de programación y XML para la gestión de las preguntas. Con todo esto, se ha llevado la siguiente planificación:

Lo primero de todo ha sido proceder a la instalación de las herramientas *NetBeans IDE 8.0.2*, Apache Tomcat 8.0, MySql, *WampServer 2.5*, *HeidiSQL 9.3*. Se han realizado pruebas para obtener una buena conexión entre la base de datos, jsp y servlets; Estudio del comportamiento de diferentes juegos tipo quiz en la web; Se ha diseñado la lógica interna del juego así como de las diferentes pantallas y los servlerts asociados a cada jsp.;

También se han diseñado y creado las tablas precisas de la base de datos con *MySQL*. Lo siguiente ha sido la implementación de un inicio de sesión con las herramientas mencionadas, pruebas de inserción de usuarios a la base de datos y acceso posterior como administrador para gestionar altas y bajas, diseño de las imágenes de los botones y fondo de pantalla con la herramienta Photoshop Cs6.

Fue necesario el aprendizaje de JDOM para la extracción del enunciado y respuestas de los ficheros XML de las preguntas y del lenguaje de programación en JavaScript. Objetivo importante ha sido lograr la sincronización del juego para varios usuarios jugando a la vez.

Por último es necesaria la recopilación de la documentación del proyecto y redacción de la propia memoria.

4.1. Hitos.

Se ha sido dividido el trabajo en una serie de hitos a cumplir para abordar paso a paso la programación del juego educativo, consiguiendo así simplificar su creación:

Hito 1: Descarga e instalación correcta de todos los programas. (30 horas)

Hito 2: Crear un servlet que se conecte a la base de datos. (75 horas)

Hito 3: Crear la base de datos y generar la tabla usuarios. (25 horas)

Hito 4: Insertar a la base de datos un usuario y seleccionarlo desde el servlet. (65 horas)

Hito 5: Crear un JSP que muestre el servlet creado en el hito 4. (25 horas)

Hito 6: Planificar y crear un boceto de las pantallas del juego y generar la interrelación entre servlets y JSP's. (70 horas)

Hito 7: Crear el inicio de sesión y registro de nuevo usuario. Generar la tabla peticiones. (65 horas)

Hito 8: Leer un fichero XML con JDOM y mostrar su contenido en un JSP. (100 horas)

Hito 9: Crear la página de usuario y la página de administrador. (50 horas)

Hito 10: Generar las tablas puntuaciones y preguntas. (15 horas)

Hito 11: Insertar desde el administrador un torneo de una sola pregunta. (35 horas)

Hito 12: Introducir un temporizador de cuenta atrás del tiempo restante entre la fecha del torneo y la actual con JavaScript. (30 horas)

Hito 13: Realizar la extracción de datos de múltiples ficheros XML y mostrarlos uno tras otro tras un tiempo breve determinado. (30 horas)

Hito 14: Insertar puntuación en la base de datos cuando la respuesta sea correcta. (50 horas)

Hito 15: Comprobar que el primer usuario que responda coja el turno. (50 horas)

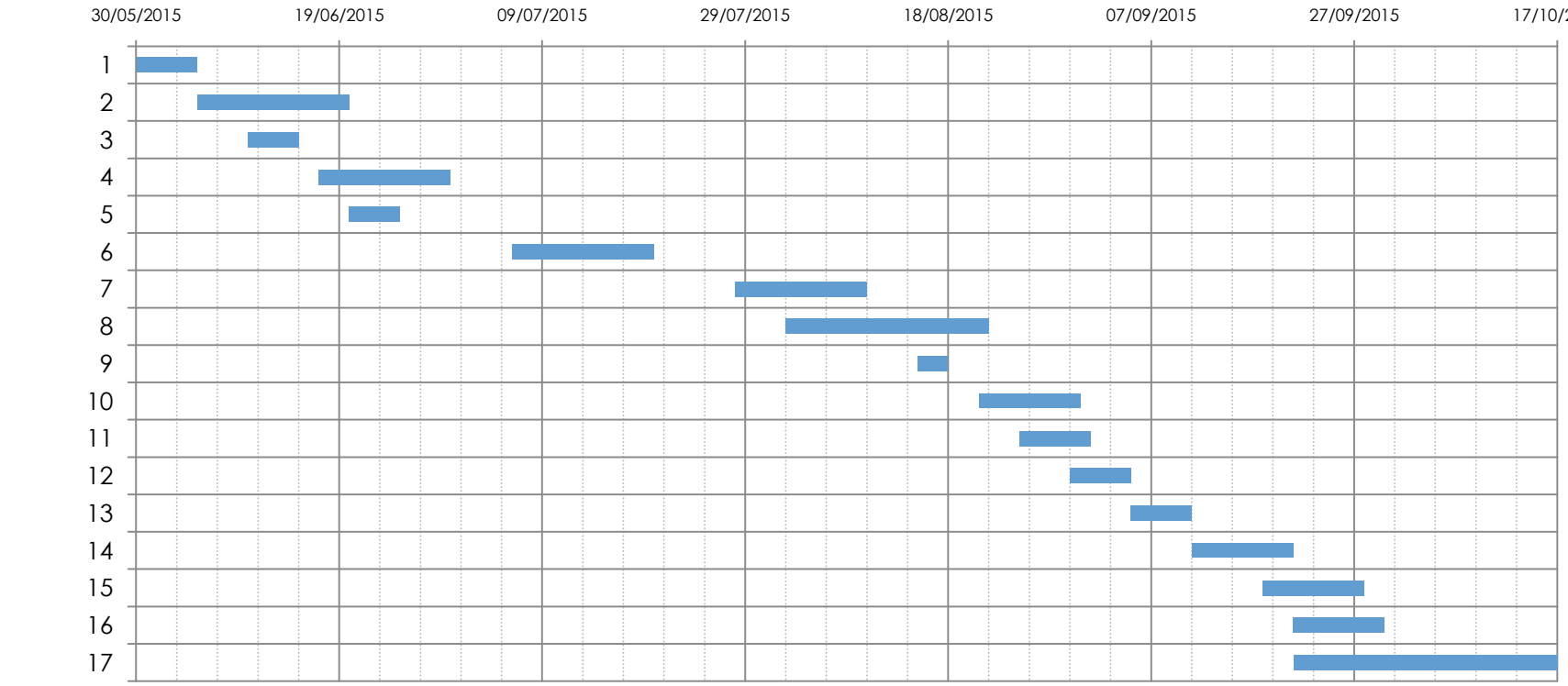
Hito 16: Recopilación de la documentación del proyecto. (40 horas)

Hito 17: Redacción de la memoria. (130 horas)

A continuación se presenta el diagrama de Gant, un gráfico con la planificación por fechas de cada uno de los hitos,

Con fecha de inicio 30/05/2015 y fecha de fin 17/10/2015, lo que en duración en días es 140 días. Horas trabajadas: 890 horas.

4.2. Diagrama de Gant.



	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
fecha de inicio	21/09/2015	21/09/2015	18/09/2015	11/09/2015	05/09/2015	30/08/2015	25/08/2015	21/08/2015	15/08/2015	02/08/2015	28/07/2015	06/07/2015	20/06/2015	17/06/2015	10/06/2015	05/06/2015	30/05/2015
duración	26	9	10	10	6	6	7	10	3	20	13	14	5	13	5	15	6

FIGURA 24: Diagrama de Gant

4.3. Presupuesto.

Teniendo en cuenta los recursos necesarios para la ejecución de esta aplicación y el tiempo empleado en el diseño y en el desarrollo de la misma, a continuación se detalla el presupuesto y los costes del proyecto:

Desarrollador: Elvira Padrino Davia

Categoría: Ingeniera Técnica de Telecomunicaciones

Tiempo Trabajado. 890 horas

Coste de ingeniero por hora: 18€/hora

Total personal: 16.020 €

Software de desarrollo: 215 €

Hardware de desarrollo: 476€

Total material: 1.167€

Costes indirectos: 2.578 €

Desplazamientos: 150 €

El presupuesto total de este proyecto asciende a la cantidad de 19.915 € sin IVA

Leganés, a 12 de Octubre de 2015

El ingeniero proyectista: Elvira Padrino Davia

---

5. Requisitos

En este apartado se describen los requisitos funcionales y no funcionales de este proyecto. Los requisitos funcionales describen tanto los servicios que deben ofrecer el sistema, como sus restricciones. Definen **qué** debe hacer un sistema. En cambio, los requisitos no funcionales definen **cómo** debe ser el sistema (las cualidades). También se incluye en este apartado el boceto inicial que se ha diseñado de la aplicación, para cumplir todos los requisitos, que incluye los casos de uso.

5.1. Requisitos funcionales.

1. Mostrar al usuario una interfaz sencilla: Para que el alumno no pierda tiempo en averiguar cómo funciona la aplicación y sea visualmente atractiva.
2. Permitir el inicio de sesión al usuario: Mediante un campo para introducir el nombre y otro para introducir la contraseña.
3. Comprobar que el usuario existe: Mediante consultas a la base de datos comprobará que el usuario está dado de alta.
4. Distinguir entre usuario y administrador: El administrador tiene una ejecución completamente diferente al usuario común.
5. Almacenar en la base de datos los nuevos usuarios: Los usuarios registrados se almacenarán en la base de datos peticiones.
6. Dar de alta a un nuevo usuario: El sistema ha de cambiar al usuario que está pendiente de ser dado de alta a la base de datos usuarios. Esto se hará por petición del administrador.
7. Dar de baja a un usuario: El sistema puede borrar a un usuario por petición del administrador.
8. Debe avisar al usuario de cada operación: Mediante mensajes, el sistema deberá confirmar cada paso final.

9. Permitir salir de la aplicación al usuario y al administrador: En todo momento debe haber una opción de salir y cerrar sesión.
10. Permitir volver a la página anterior al usuario y al administrador: En todo momento debe haber una opción de volver/atrás.
11. Mostrar las puntuaciones totales: Debe acceder a la base de datos puntuaciones y mostrar las puntuaciones de forma ordenada.
12. Borrar las puntuaciones: El sistema solo dará esta funcionalidad al administrador.
13. Mostrar las instrucciones de uso: El sistema ha de facilitar siempre las instrucciones del juego.
14. Almacenar la fecha del torneo: El sistema almacenará la fecha del torneo en la base de datos dada por el administrador.
15. Ofrecer dos temporizadores:
    - Primer temporizador: el sistema debe hacer un cálculo con la fecha del torneo y la actual para mostrar el tiempo restante de modo que cada segundo se descuente una unidad de este resultado. Se dejará de mostrar cuando éste llegue a cero.
    - Segundo temporizador: Es un temporizador de tiempo fijo, que cada segundo de resta una unidad. El sistema debe inicializarlo en el comienzo de cada pregunta tras haber llegado a cero.
16. Leer ficheros XML dada una ruta: El sistema debe tener la capacidad de leer todo tipo de ficheros XML bajo es estándar QTI[2] de donde se obtendrán las preguntas.
17. Almacenar las preguntas en la base de datos: Se obtendrán los parámetros necesarios de cada pregunta y se almacenarán de forma ordenada.

[2] https://es.wikipedia.org/wiki/QTI: La Especificación de Interoperabilidad de Preguntas y Pruebas de IMS (IMS/QTI por sus siglas en inglés) define un formato estándar para la representación de contenidos y resultados de evaluaciones educativas. Permite la creación y entrega de materiales de pruebas en múltiples sistemas de forma intercambiable. La especificación consiste en un modelo de datos que define la estructura de preguntas, evaluaciones y resultados a partir de preguntas y evaluaciones juntas con una representación vinculada en XML.

18. Reconocer qué jugador responde primero: El sistema debe controlar qué jugador es el más veloz.
19. Comprobar si la respuesta es correcta: El sistema hará comprobaciones en la base de datos para saber si es correcta o no la respuesta seleccionada por el usuario.
20. Pasar el turno a otro jugador: Si el usuario responde erróneamente el sistema le pasará el turno al siguiente jugador más rápido.
21. Almacenar las puntuaciones: El sistema debe almacenar las puntuaciones de sólo aquel jugador que responda el primero correctamente a cada pregunta.
22. Mostrar al usuario el final del juego: El sistema avisará al jugador de que el torneo ha finalizado.

5.2. Requisitos no funcionales.

Interfaces:

- Hardware: El sistema se debe implementar en un ordenador con Sistema Operativo Windows 7/10.
- Software: La aplicación deberá funcionar sobre los navegadores Chrome, Firefox, Internet Explorer, Opera, Safari.

Rendimiento:

- El sistema no debe tardar más de 3 segundos en mostrar los resultados de una búsqueda.

Disponibilidad:

- La aplicación está diseñada para que pueda adaptarse a todo tipo de preguntas, por lo que este sistema puede ser actualizado en su contenido como se desee.

Extensibilidad y mantenibilidad:

- La implementación de la aplicación se rige bajo el patrón MVC, por lo que se facilita enormemente su crecimiento en el futuro al poder mejorar la vista independientemente del control.

Usabilidad:

- La facilidad de uso es una de las premisas de este proyecto. Es una aplicación de fácil aprendizaje en su uso.

Portabilidad:

- Es una aplicación web, por lo que desde cualquier dispositivo con acceso a internet, se podrá acceder a la aplicación.

5.3. Boceto inicial del Juego.

Antes de comenzar a crear este proyecto, lo primero de todo fue diseñar un boceto previo del juego que se muestra a continuación.

Es una primera versión con la idea principal del funcionamiento del mismo que más tarde ha ido modificándose según las mejoras que ha adquirido.

El boceto parte de la idea de tener una pantalla inicial con un diseño muy limpio con los elementos mínimos para el inicio de sesión. Esta pantalla tiene dos funciones: Por un lado tiene dos cajas de texto para introducir el usuario y contraseña y un botón para acceder; y por otro lado un vínculo para registrarse como nuevo usuario.

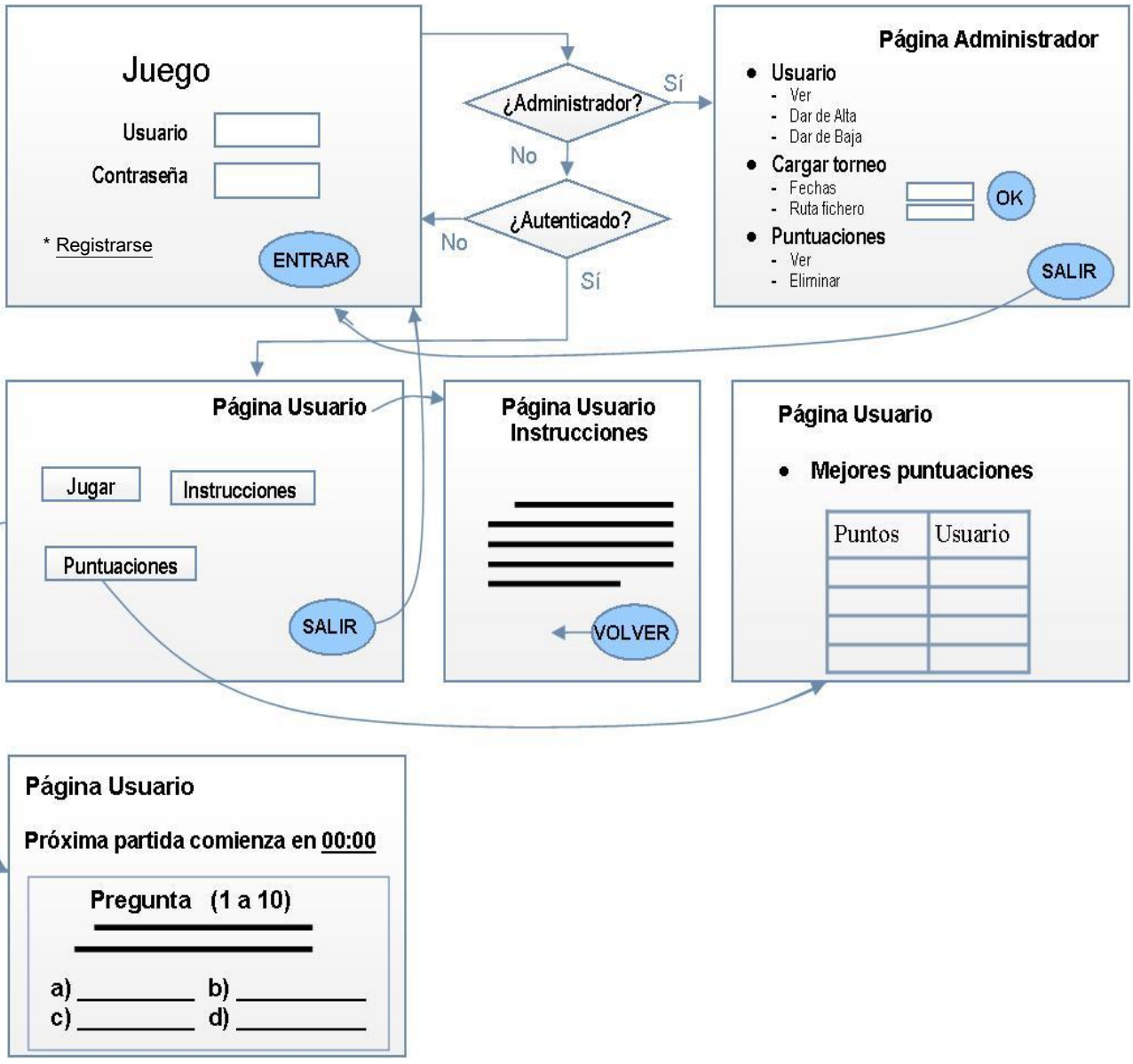


FIGURA 25: Boceto del Juego

El resto de pantallas seguirá la misma estética con los mínimos elementos para facilitar al usuario y que de un solo vistazo, sepa en todo momento como acceder a las diferentes funcionalidades.

Tras la primera pantalla pueden ocurrir cuatro casos según el elemento seleccionado y/o los parámetros introducidos:

CASO 1: Si se introducen los parámetros y es el administrador

En este caso la segunda página que se presenta es una página de herramientas del propio administrador donde siempre hay opción de salir y volver a la página de inicio. Estas herramientas son:

- Dar de alta y dar de baja a los usuarios. Solo el administrador puede gestionarlos, verlos y modificarlos. En el boceto estas dos funcionalidades sólo están indicadas, pero finalmente se implementarán con un desplegable con los usuarios y junto a éste un botón para dar de alta o baja en cada caso. Toda la gestión se hará por detrás y en usuario solo verá los usuarios dados de alta y los que está pendiente de estarlo en dicho desplegables.
- Cargar torneo. Con una caja de texto con formato de fecha para la misma, otra segunda caja de texto para poner la ruta donde se sitúan los ficheros XML y un botón de confirmación, basta para cargar los torneos.

- Puntuaciones. Consiste en un único botón que si se pulsa se accede a una tercera pantalla que contiene una tabla con los datos de los mejores resultados. Esta pantalla en concreto será la misma pantalla que tiene el usuario común, solo que con dos privilegios extra: Mediante una caja de texto y un botón, puede introducir el nombre del usuario que quiera borrar la puntuación y mediante un único botón puede borrar toda la tabla y resetearla. Esta pantalla no aparece en el boceto inicial ya que se dejó para más adelante la decisión de su diseño.

CASO 2: Si se introducen los parámetros y es un usuario

En este caso la segunda página que se presenta es una página personal del usuario donde hay cuatro posibilidades que se visualizan con cuatro botones definidos:

- Jugar te lleva a otra pantalla donde, si hay un torneo cargado para una fecha que aún no ha expirado, aparecerá una cuenta atrás de tiempo. Cuando ésta llegue a 0, aparecerá la primera pregunta con cuatro opciones a seleccionar. La idea es que, tras un temporizador fijo que llegue a 0, se cargue la siguiente pregunta, reseteando ese temporizador fijo.
- Instrucciones es una pantalla con un texto informativo con las instrucciones del juego.
- Puntuaciones, es la misma pantalla que utiliza el administrador, pero en el caso de un usuario solo aparece la tabla a modo de tabla informativa con las mejores puntuaciones.
- Salir, es un botón que te llena de nuevo a la primer pantalla, haciendo así que el usuario cierre sesión.

CASO 3: Si se introducen los parámetros y no está registrado

Este es el caso de error. Si se introduce a un usuario que no exista en la base de datos aparecerá en la pantalla inicial un mensaje de texto de error. También ocurre igual si no se introducen parámetros, el mensaje de error especificará cual ha sido el problema para que el usuario sea consciente del fallo.

CASO 4: Si se accede a “Registrarse”

En el caso de acceder al vínculo “Registrarse”, te lleva a una pantalla para ingresar los datos de nuevo usuario. En esta pantalla aparecerán tres campos de texto: uno de ellos para introducir el nuevo usuario, y dos más para la contraseña y para la repetición de la misma. A su vez hay un botón de confirmación para validar los datos.

6. Diseño del Juego

En este capítulo se va a explicar la lógica de programación interna del juego. Se mostrarán primero las diferentes clases de java, su funcionamiento y su interrelación con otras clases y JSPs.

La dinámica de programación ha sido sencilla con el uso de JSPs para la vista y de los Servlets para el control. En todo momento se ha seguido el patrón MVC (Modelo – Vista – Controlador)

6.1. Patrón MVC.

La razón de usar el patrón MVC (Modelo – Vista – Controlador) es porque separa la lógica de negocio de la interfaz de usuario. De este modo, ambos pueden ser modificados sin perjudicar al otro en su funcionamiento, lo que hace de esta aplicación una aplicación muy flexible y reutilizable para el desarrollador.

En el Modelo-Vista-Controlador hay un modelo, varios controladores y varias vistas. Estos dos últimos suelen estar muy relacionados.

En las aplicaciones Web la vista es la página HTML, el controlador sería el código que obtiene los datos de forma dinámica y genera el contenido del HTML y, por último, el modelo es la información guardada en una base de datos y/o XML.

A continuación se muestran todas las clases que comprenden nuestra aplicación web basada en J2EE organizadas según el patrón MVC.

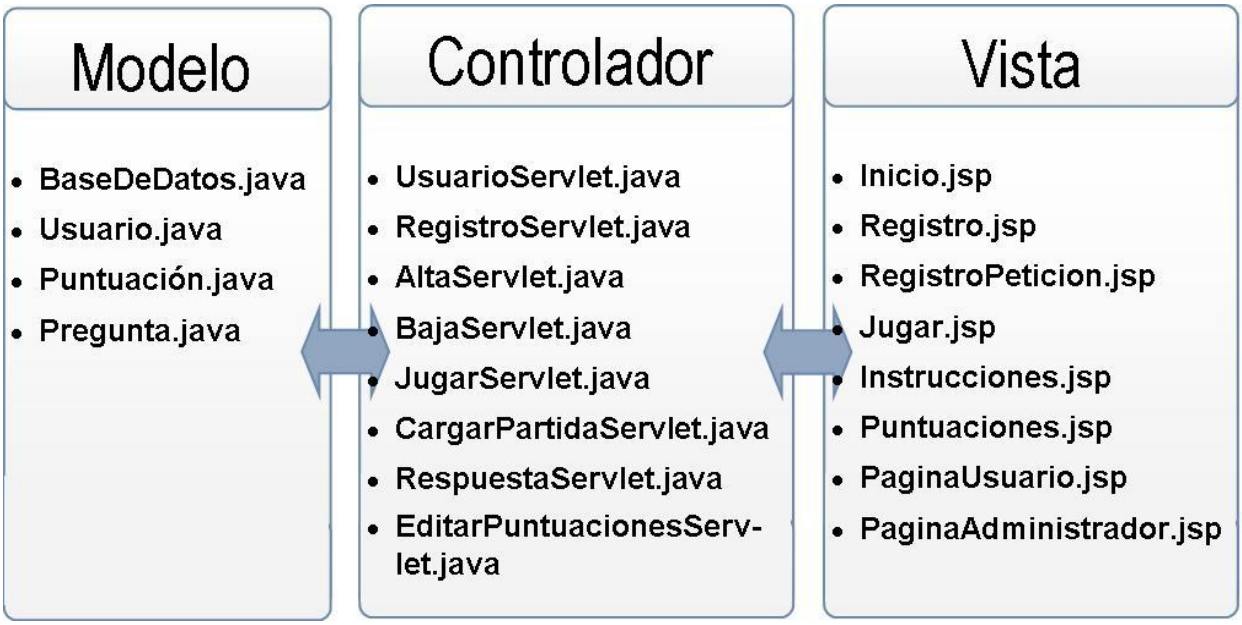


FIGURA 26: Clases, MVC del juego educativo

En la parte del **MODELO** se hallan las clases: **BaseDeDatos.java**, que es la que tiene la interacción con la base de datos y contiene todos los métodos que realizan consultan e inserciones a la base de datos; **Usuario.java** es la clase que define a cada usuario, incluido el administrador y le dota de una identificación, nombre y contraseña; **Puntuación.java** es la clase que define las puntuaciones obtenidas; **Pregunta.java** es la clase que define a las preguntas y a su vez la que tiene todo el código preciso para acceder a los ficheros XML y guardarlos como clase Pregunta.

En el **CONTROLADOR** tenemos todos los servlets de la aplicación desarrollada, que son **AltaServlet.java**, **BajaServlet.java**, **CargarPartidaServlet.java**, **EditarPuntuacionesServlet.java**, **JugarServlet.java**, **RegistroServlet.java**, **RespuestaServlet.java**, **UsuarioServlet.java**. Estos utilizan métodos y clases del modelo para darle un resultado a la **VISTA**, que son todos los JSP de la aplicación y ofrecen el resultado de una forma visual. Éstos son: **Inicio.jsp, Registro.jsp, RegistroPeticion.jsp, Jugar.jsp, Instrucciones.jsp, Puntuaciones.jsp, PaginaUsuario.jsp** y **PaginaAdministrador.jsp.**

6.2. Diagrama de clases.

A continuación se presentan los diagramas de las clases de este proyecto separadas según el patrón MVC

6.2.1. Diagrama de clases-Modelo

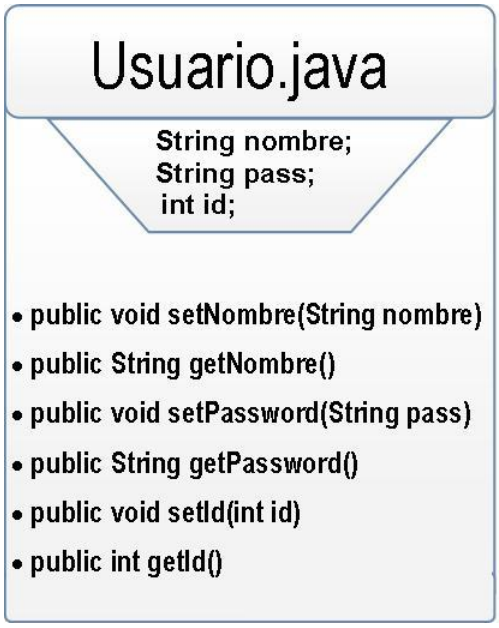
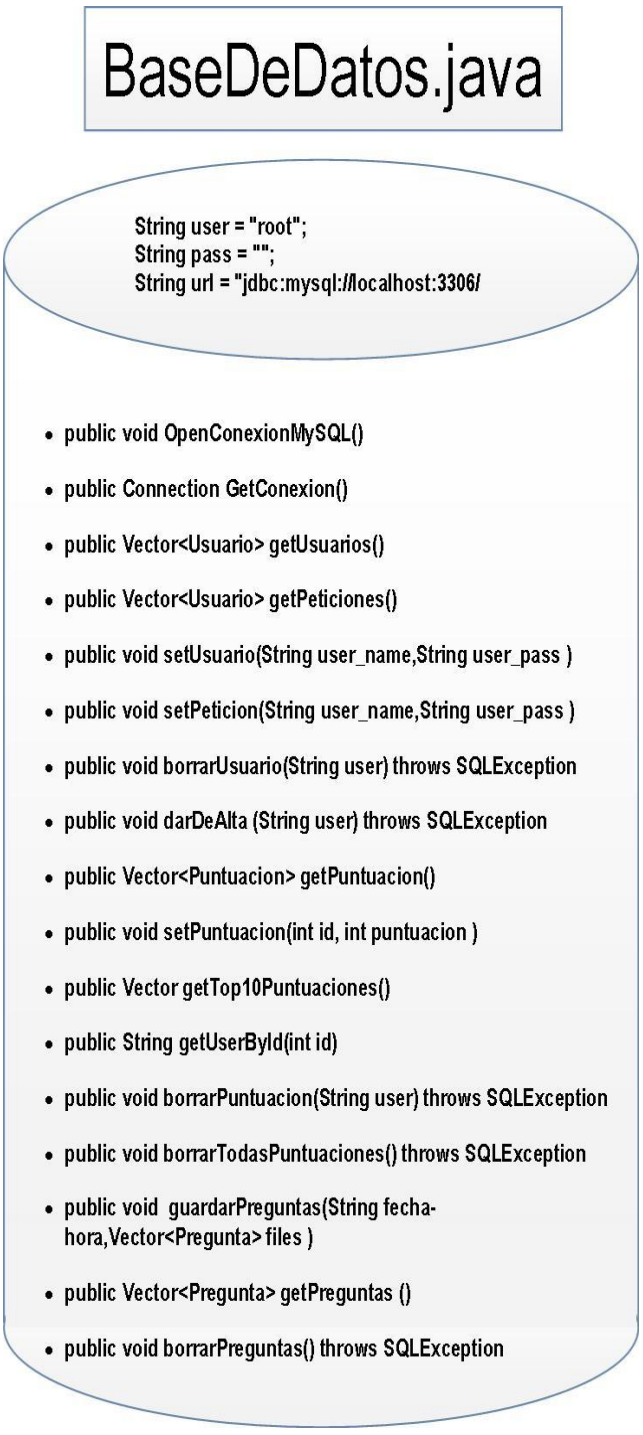


FIGURA 27: Usuario

FIGURA 28: Puntuación

Las clases Usuario.java y Puntuación.java son necesarias para definir los formatos de usuario y puntuación y que así el controlador tenga fácil acceso a dichas clases y pueda gestionar toda la aplicación. Los métodos que tienen, en ambos casos, son los que hay en el interior de cada figura.

La clase BaseDeDatos.java es la clase con más contenido, ya que en ella está toda la gestión de la aplicación. Tenemos los métodos de consulta tipo **SELECT** a la base de datos como getUsuarios, getPeticiones, getPuntuacion, getTop10Puntuaciones, getUserById y getPreguntas.



Los métodos del tipo **INSERT** son setUsuario, setPeticion, setPuntuacion y guardarPreguntas.

Los método tipo **DELETE** son borrarUsuario, borrarPuntuacion, borrarTodasPuntuaciones y borrarPreguntas.

El método darDeAlta combina las tres, ya que lo que hace es selecciona y borrar de una tabla, para insertar esos datos en una nueva tabla.

Un método muy importante es OpenConexionMySQL, ya que es necesario llamarlo para cada gestión con la base de datos.

FIGURA 29: Base de datos

La clase Pregunta.java maneja el acceso a los ficheros XML a través de las clases File, Document,, SAXBuilder y Element. Con ellas se han implementado los métodos getData y getFilesByDirectorio de forma que acceden al fichero XML en cuestión, lo leen etiqueta a etiqueta guardando solo aquellas que tienes el contenido enunciado, respuesta correcta, opción a, opción b, opción c y opción d. Estos parámetros se almacenan como una pregunta de tipo Pregunta.java. De modo que así la información de cada pregunta sea accesible por la aplicación en todo momento.

El resto de métodos son para crear y devolver estos seis parámetros mencionados en esta explicación.

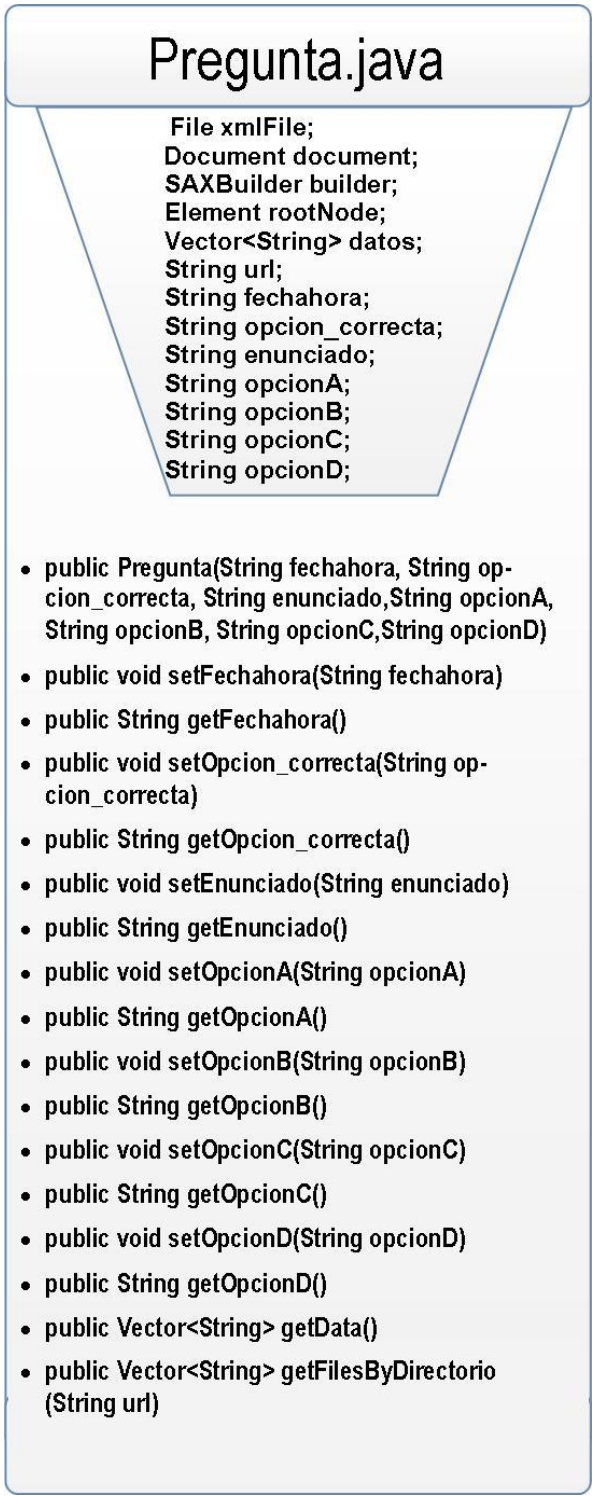


FIGURA 30: Pregunta

6.2.2. Diagramas Servlets-Controlador

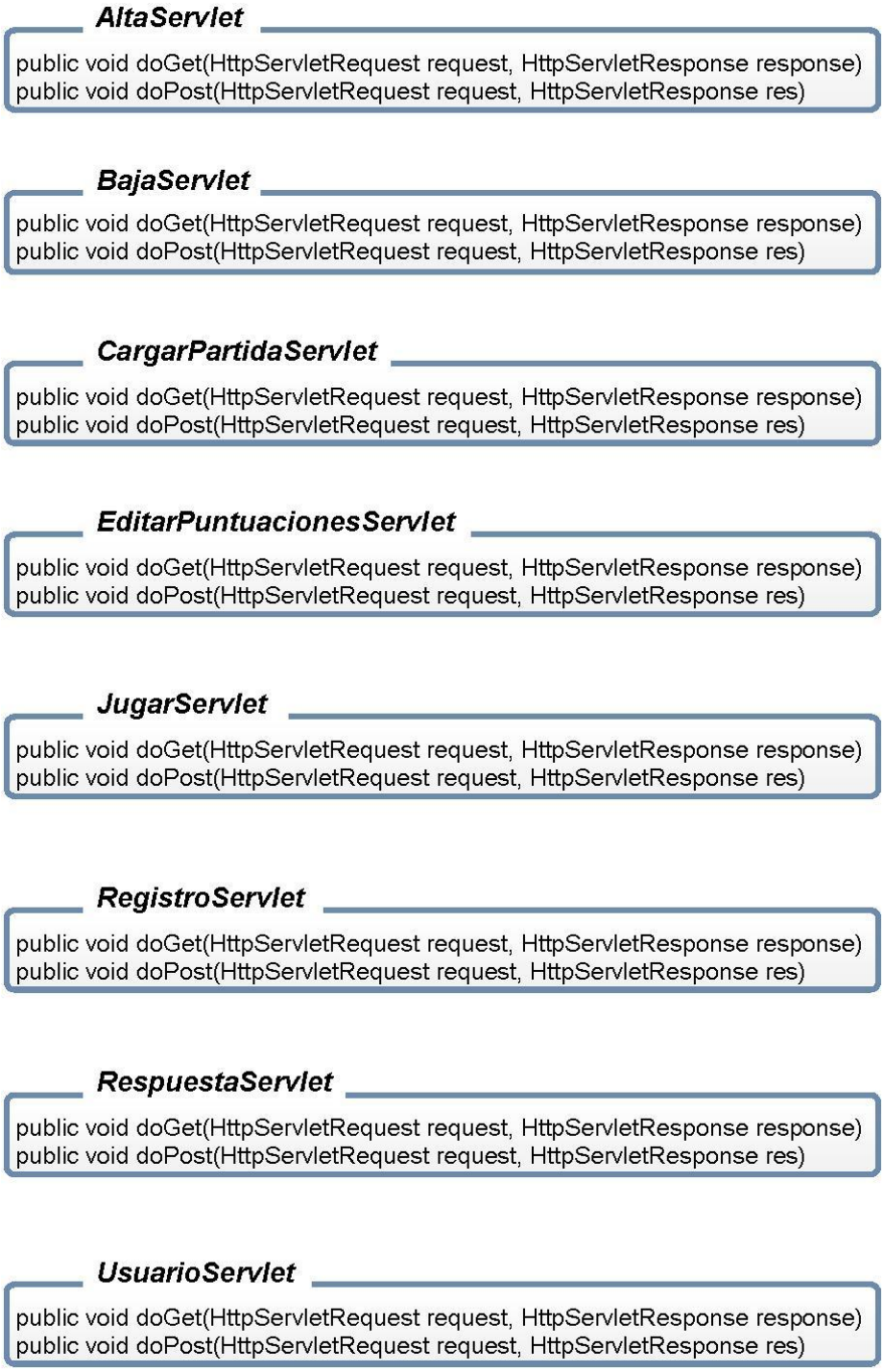


FIGURA 31: Diagramas de clases (Controlador)

Los servlets gestionan y controlan toda la aplicación mediante peticiones GET y POST. Cogen y devuelven los parámetros a los JSP tras aplicar métodos del Modelo.

6.2.3. Diagrama de actividad

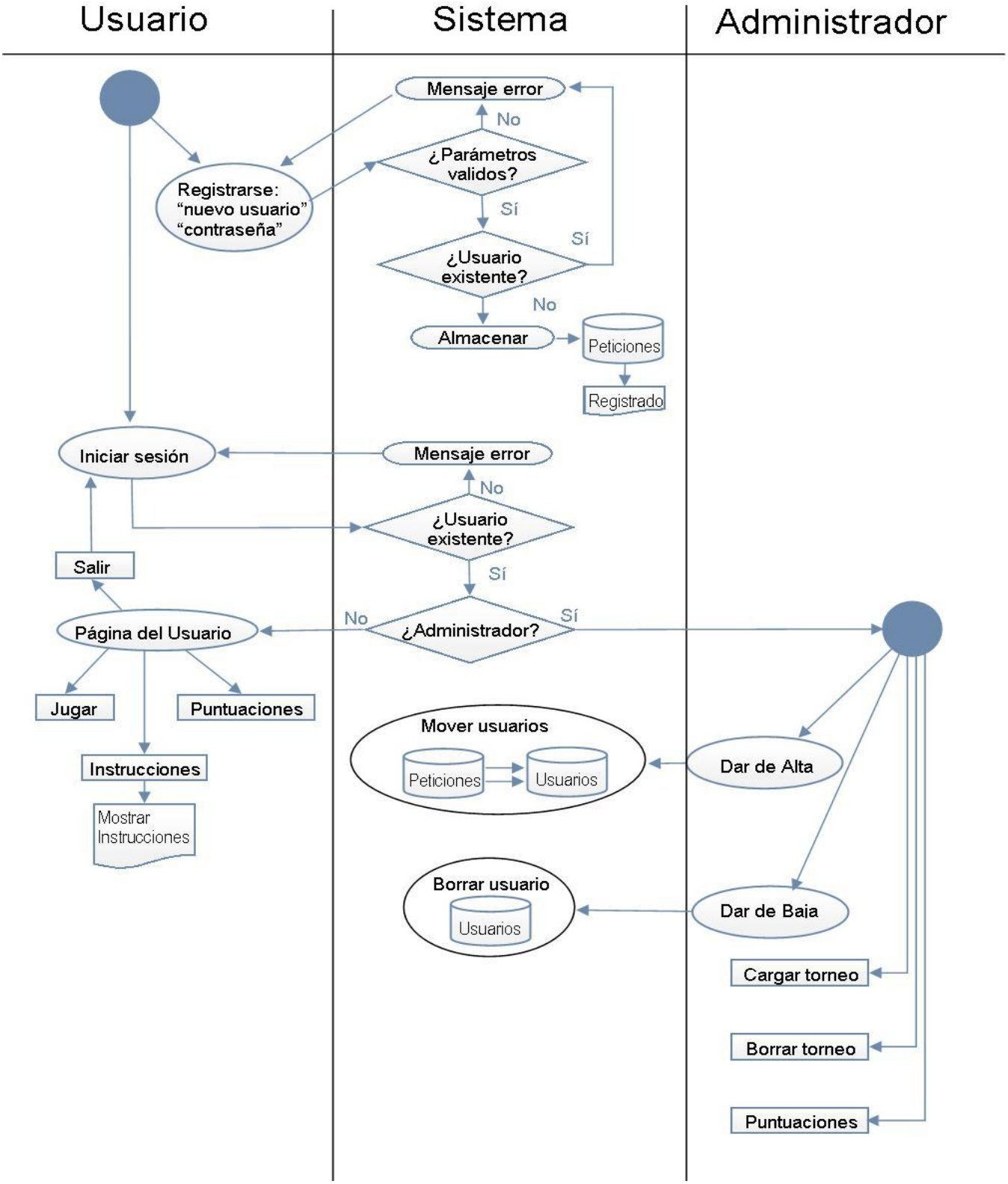


FIGURA 32: Diagrama de actividad

En la figura anterior se muestra un diagrama de actividad de la aplicación del juego, desde el punto de vista del usuario común, del sistema y del administrador, el funcionamiento de la aplicación.

Un usuario (el punto azul de la izquierda) tiene dos opciones, registrarse o iniciar sesión. Si elije registrarse, inserta el nuevo nombre de usuario y la nueva contraseña y pasa a tener el control el sistema, donde se realizan comprobaciones de si son correctos los campos introducidos de si el nombre de usuario no está ya registrado. Si hay cualquier error, se le hace saber al usuario llevándole al inicio con un mensaje de error. Pero si todo el proceso de registro es válido, este usuario se almacena en la base de datos "peticiones".

En cambio, si el usuario decide iniciar sesión directamente, al introducir su nombre y contraseña, le otorga de nuevo el poder al sistema, el cual hará las comprobaciones precisas para saber si el usuario está dado de alta. Si no fuese así, de nuevo se le devuelve a la página inicial indicándole un mensaje de error para que el usuario sepa que algo hubo mal. Pero si el usuario fuese válido, estando dado de alta, el sistema comprueba si este usuario es el administrador. En caso de no serlo, a este usuario se le devuelve el control con la pantalla personal del mismo usuario. En este punto tiene las cuatro opciones que puede seleccionar (Jugar, Puntuaciones, Instrucciones y Salir). Si decide salir, el sistema le devuelve al inicio. Si selecciona Instrucciones, el sistema le muestra una pantalla con las instrucciones del juego. Si selecciona "Jugar" o "Puntuaciones" se explicará que sucede detalladamente en las siguientes figuras. Antes de esto, todo cambia si el usuario que inicia sesión es el administrador.

Si es el administrador el que inició sesión, le aparece la siguiente pantalla con todas las funcionalidades del administrador, que son Dar de alta (que el sistema mueve al usuario de la base de datos peticiones a usuarios), Dar de baja (donde el sistema borra por completo al usuario que el administrador seleccionó), borrar torneos (borra la base de datos de preguntas) y Cargar torneo y Puntuaciones que se explicará a continuación:

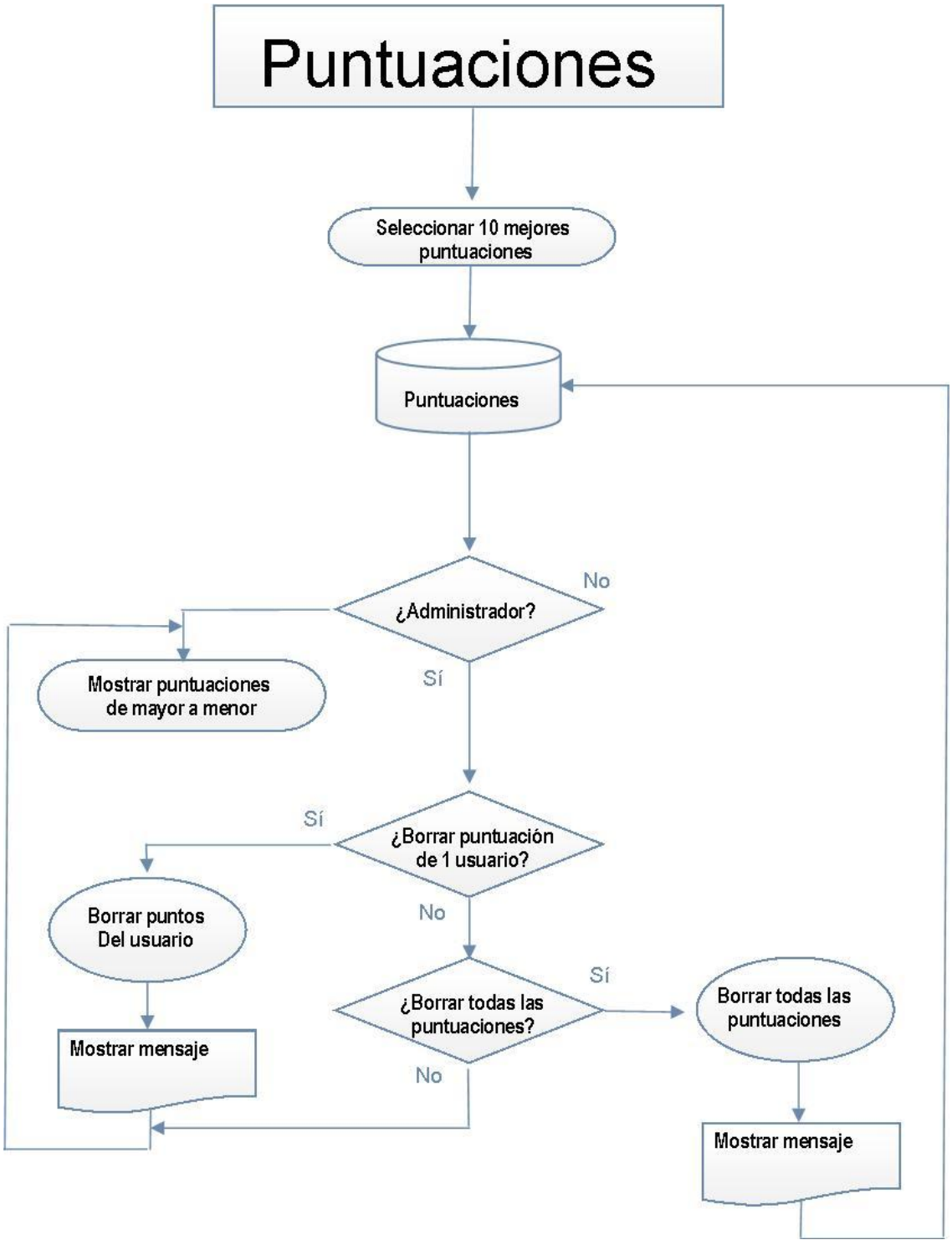


FIGURA 33: Diagrama de Puntuaciones

Se ha sacado fuera del diagrama de actividad, la casilla Puntuaciones para poder explicarla con claridad.

Como se puede observar tanto el usuario como el administrador tienen la funcionalidad "Puntuaciones". Si es seleccionada, el sistema accede a la base de datos y selecciona las diez mejores puntuaciones. Antes de devolverle el control al usuario o al administrador, comprueba esta cuestión. Si el usuario es un usuario común, se le devuelve en control mostrándole las mejores puntaciones. Si resulta ser que el usuario es el administrador, le muestra en su página las puntuaciones seleccionadas y tiene dos opciones extra borrar una puntuación concreta o todo el ranking. El borrado lo hará de nuevo el sistema. Al administrador le aparecerá solo un mensaje en la pantalla confirmando que ha sido borrado.

Para el siguiente diagrama Jugar, continuando el hilo del usuario común, cuando éste accede a jugar, el sistema pasa a comprobar si hay algún torneo vigente cargado. Si no fuera así, el sistema le muestra al usuario un mensaje informándole de que debe esperar. Pero en el caso de que si haya un torneo, al usuario le aparecerá una cuenta atrás del tiempo que le queda al torneo para comenzar. Cuando ésta llegue a cero, se muestra la primera pregunta junto a un temporizador de tiempo fijo. Si el jugador decidiese no contestar, cuando este tiempo llegase a cero, se mostraría la siguiente pregunta. Pero si decide responder una de las cuatro opciones, entonces el sistema pasa a hacer unas rápidas comprobaciones mientras el tiempo va descontando. Lo primero que comprueba es si el usuario ha sido el primero en responder. Si no fuese así, debe esperar a que le llegue el turno. Pero en caso afirmativo, que haya sido el más veloz en responder, se comprueba si ha sido correcta su respuesta. Si así lo es, se almacena en la base de datos puntuaciones. De no ser así, pasa el turno al siguiente jugador más rápido. Éste seguiría el mismo proceso. Para todos los casos, si el tiempo llega a cero, se muestra la siguiente pregunta hasta llegar a 10. Y para todos los casos, se responda bien o mal, ha de esperarse a que el temporizador acabe. De esta forma jugarán todos los usuarios de forma síncrona.

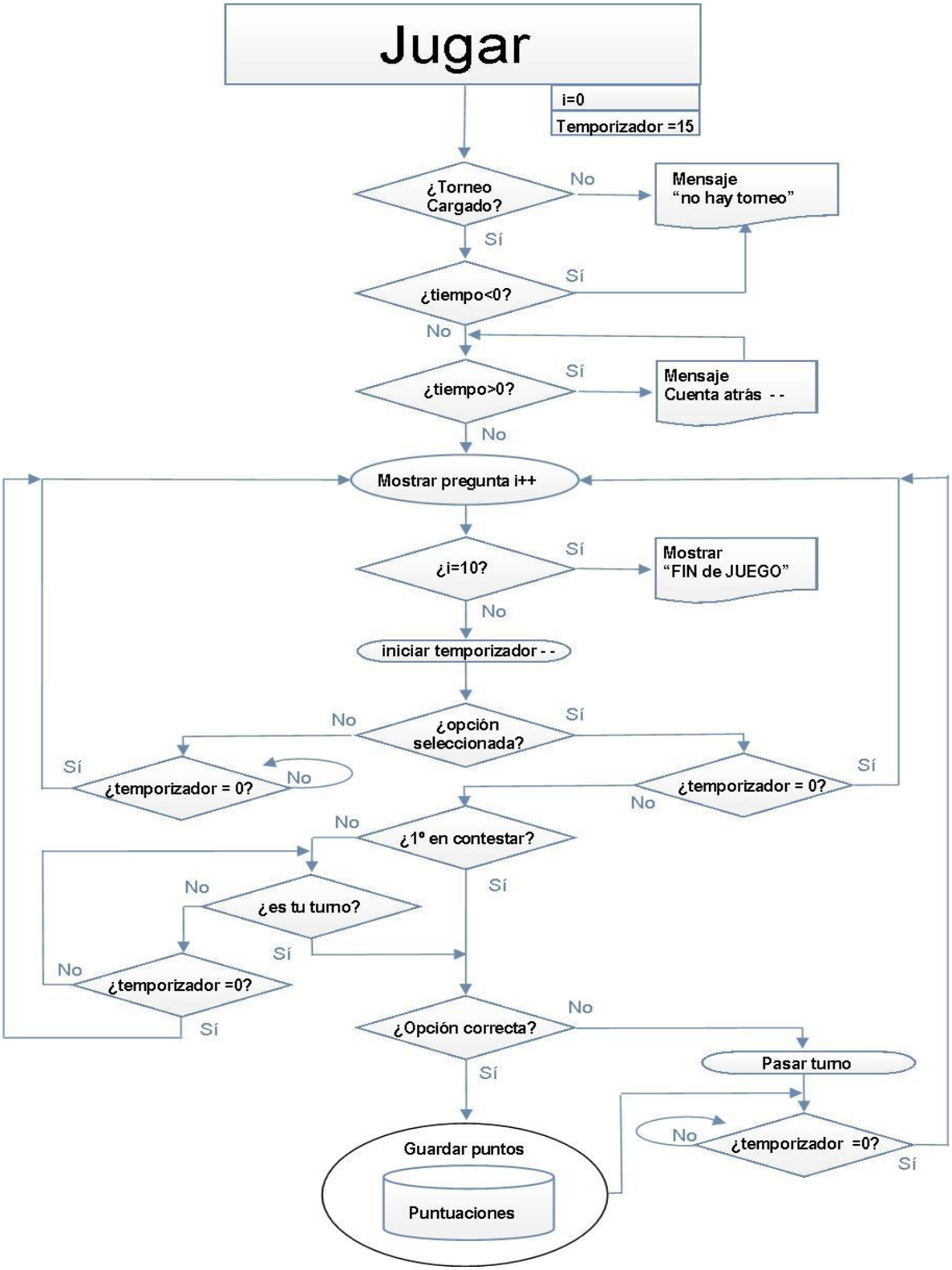


FIGURA 34: Diagrama Jugar

Solo queda explicar el caso de que el administrador seleccione la opción Cargar torneo, para la cual previamente introduce la fecha de comienzo del torneo y la ruta donde se encuentran los ficheros XML de las preguntas bajo el estándar QTI[3] y el sistema se encarga de ir a la carpeta que se ha indicado, coger cada fichero XLM uno a uno, guardar la información precisa de cada pregunta y almacenarla en la base de datos preguntas. Si la ruta estuviera mal introducida o no hubiera ficheros XML tipo QTI, saltaría un error al administrador informándole. En caso de que la carga del torneo con todas sus preguntas sea exitosa, el administrador solo verá el mensaje con la información de que fue correcta la carga.

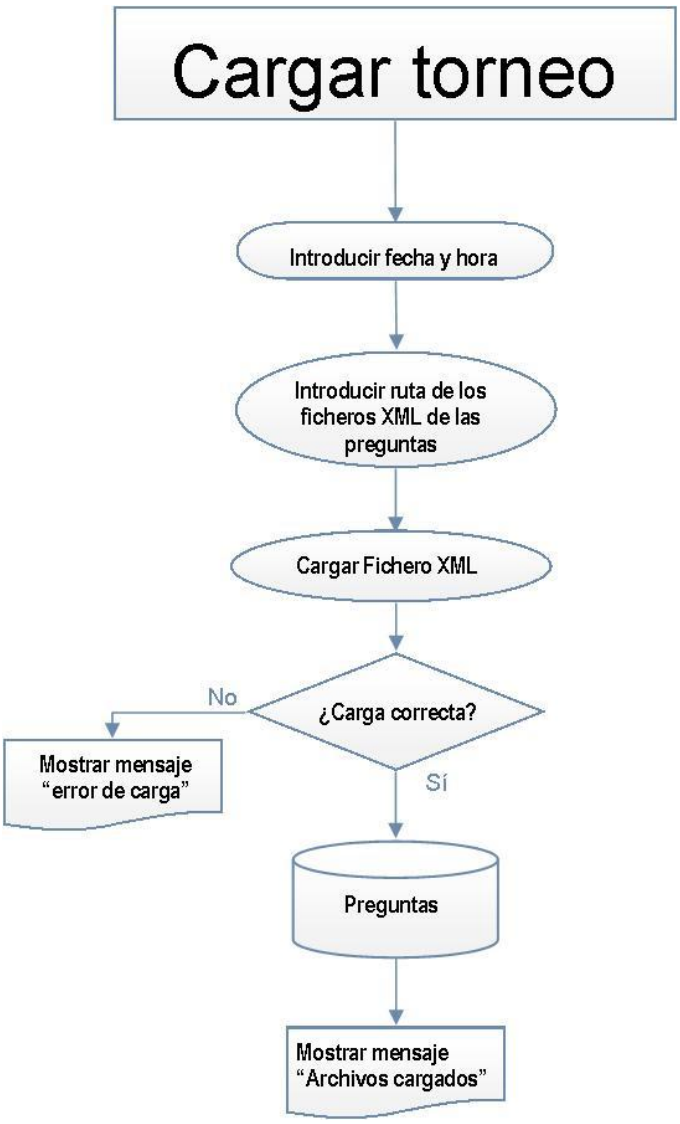


FIGURA 35: Diagrama Cargar torneo

[3]https://es.wikipedia.org/wiki/QTI: La Especificación de Interoperabilidad de Preguntas y Pruebas de IMS (IMS/QTI por sus siglas en inglés) define un formato estándar para la representación de contenidos y resultados de evaluaciones educativas. Permite la creación y entrega de materiales de pruebas en múltiples sistemas de forma intercambiable. La especificación consiste en un modelo de datos que define la estructura de preguntas, evaluaciones y resultados a partir de preguntas y evaluaciones juntas con una representación vinculada en XML.

6.2.4. Tablas de la base de datos

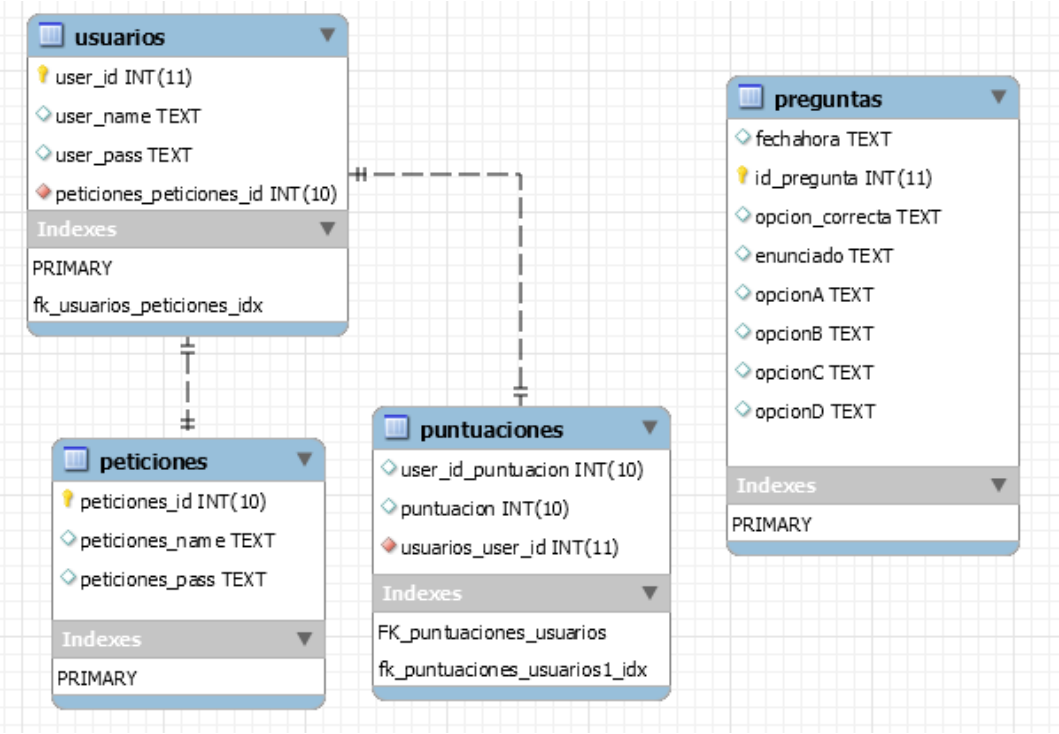


FIGURA 36: Tablas de la base de datos bd_juego

La tabla "usuarios" contiene la identificación única, el nombre de usuario y la contraseña proporcionada por él. Es la tabla en la que están almacenados estos datos del administrador y todos los usuarios que éste ha dado de alta en la aplicación. La tabla "peticiones" están almacenados todos aquellos usuarios que hay hecho una petición de registro a la aplicación del juego y están a la espera de ser dados de alta y pasar así automáticamente a la tabla "usuarios". En la tabla "puntuaciones" se almacena la puntuación de cada alumno.

Por último, la tabla "preguntas" almacena los datos del torneo: la fecha de comienzo del mismo más el enunciado, respuestas y respuesta correcta de cada pregunta, la cual tiene su identificador único. Estas preguntas las ofrecen los ficheros XML cargados previamente por el administrador y se almacenan junto con la hora de comienzo del torneo.

7. Funcionamiento e Implementación

7.1. Registro de nuevo usuario.

FIGURA 37: Página de inicio del Juego

En la pantalla principal el usuario debe introducir el usuario y contraseña y pinchar en "ENTRAR". Si es un usuario que nunca ha sido dado de alta o nunca se registró, la aplicación le llevará a la página de inicio con un mensaje de error como indica la siguiente figura.

FIGURA 38: Mensaje de error de usuario

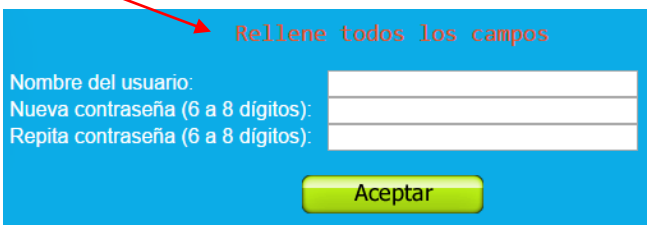
Esta página es el JSP Inicio.jsp. Si el usuario accede a "Nuevo usuario", llamará a Registro.jsp:

FIGURA 39: Registro.jsp nuevo usuario

En este JSP El usuario ha de introducir su nombre y su contraseña, la cual debe ser de entre 6 y 8 dígitos; Para mayor seguridad ha de rellenar la misma dos veces para eliminar los posibles errores de escritura. Una vez introducidos estos datos, se hace una comprobación de si los datos son correctos y de que no haya campos vacíos. Finalmente se comprueba en la base de datos que el usuario no esté ya dado de alta, o esperando a estar dado de alta. Toda esta gestión la atiende la clase `RegistroServlet.java` con el siguiente código:

```
BaseDeDatos bbdd = new BaseDeDatos();
bbdd.OpenConexionMySQL();
Vector<Usuario> u;
Vector<Usuario> p;
boolean valido =true;

if((pass1.equals(""))||(user.equals(""))&&(pass2.equals(""))||(user.equals(""))&&(pass1.equals(""))||
(pass2.equals(""))){
 request.getRequestDispatcher("Registro.jsp?mensaje1= Rellene todos los campos ").forward(request,res);
        }
else{
        if(pass1.equals(pass2)){
               if(bbdd.GetConexion()!=null){
                        int k=0;
                        int j=0;
                        u=bbdd.getUsuarios();
                        p=bbdd.getPeticiones();
                        request.setAttribute("user",user);
                                while(j<u.size()){      //buscamos el usuario en la bbdd (usuarios)
                                        if (u.get(j).getNombre().equals(user)) valido=false;
                                        j++;
                                        }
                        while(k<p.size()){     //buscamos el usuario en la bbdd (peticiones)
                                if (p.get(k).getNombre().equals(user)) valido=false;
                                k++;
                                }
                        if(valido == true) {
                               bbdd.setPeticion(user, pass1);//Usuario registrado, pendiente de alta
                               request.getRequestDispatcher("RegistroPeticion.jsp").forward(request,res);
                                }
                        }
                 }
        else {
        request.getRequestDispatcher("Registro.jsp?mensaje2= El usuario ya existe ").forward(request,res);
               }
        }
```



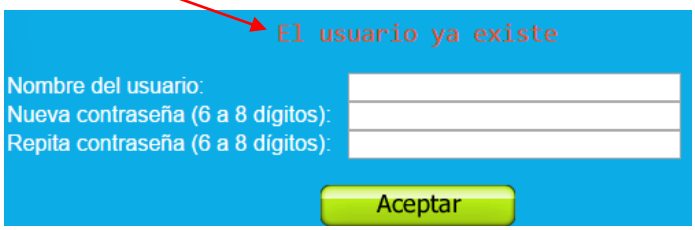


FIGURA 40: RegistroServlet.java

En caso de que todos los parámetros sean correctos y el usuario no exista en la base de datos se procede al almacenamiento en la base de datos del usuario en cuestión. Se le mostrará el mensaje de que el registro ha sido correcto y que debe esperar. El usuario deberá esperar a que el administrador le de de alta para poder iniciar su sesión de usuario.

7.2. Sesión del usuario.

Una vez que el administrador ha dado de alta al usuario, éste podrá ingresar libremente a su página personal donde aparecen diferentes elementos que se explican a continuación:

FIGURA 41: Página personal

7.2.1. Jugar

En la opción "Jugar", cuando el usuario acceda podrán ocurrir dos casos. Esta parte de la aplicación está controlada por JavaScript:

CASO 1:

Si el administrador no ha cargado el torneo o bien haya expirado porque el usuario acceda tarde al mismo. En este caso aparecerá el mensaje:

FIGURA 42: Torneo, caso 1

CASO 2:

Si el administrador ha cargado el torneo y aún no ha expirado en el tiempo aparecerá una cuenta atrás del tiempo que queda para que comience el torneo.

FIGURA 43: Torneo, caso 2

El contador de tiempo actualiza la información cada segundo. Como se mencionó anteriormente la gestión del juego se basa en scripts de java. A continuación se detalla el código y el algoritmo para obtener el tiempo restante:

```
function countdown(id){
  var fecha=new Date(a,me,d,h,mi,s); //fecha y hora del torneo
  var hoy=new Date();//fecha y hora actual
  var dias=0;
  var horas=0;
  var minutos=0;
  var segundos=0;
  var n= <%=n%>; //contador de preguntas
      if(n===11){
          document.getElementById('fin').innerHTML='FIN DEL JUEGO';
      }
      if (fecha>hoy){
          var diferencia=(fecha.getTime()-hoy.getTime())/1000;
          dias=Math.floor(diferencia/86400);
          diferencia=diferencia-(86400*dias);
          horas=Math.floor(diferencia/3600);
          diferencia=diferencia-(3600*horas);
          minutos=Math.floor(diferencia/60);
          diferencia=diferencia-(60*minutos);
          segundos=Math.floor(diferencia);
          document.getElementById(id).innerHTML='Quedan ' + dias + ' D&iacute;as, '
                  + horas + ' Horas, ' + minutos + ' Minutos, ' + segundos + ' Segundos';

          if (dias>0 || horas>0 || minutos>0 || segundos>0){
              setTimeout("countdown(\"" + id + "\")",1000);
          }
          else  {
             // document.getElementById('jugar').innerHTML='A jugar1!!!';
          document.getElementById('jugar').innerHTML= ''+
          '<form name = "formjugar" action="jugar?jugar=1" method="post" \n\
                                  onsubmit="document.formujugar.submit()">'+
          '<input  class="botonX" type = "hidden" value="Aceptar"> </form>';
          document.formjugar.submit();
          }
      }
```

FIGURA 44: Código de countdown , JavaScript

La función `countdown` rige el torneo. Hace el algoritmo para restar la fecha del torneo con la actual dividiéndola así entre días, horas, minutos y segundos. Esta información se actualiza cada segundo hasta que la fecha actual y la del

torneo coinciden. Cuando esto ocurre, mediante un flag que se pasa por parámetro, se activa el torneo en sí mismo, con todas sus preguntas y respuestas. Como se puede observar en el código, al llegar a 10 preguntas el juego finaliza. Si el administrador quisiera cargar menos de 10 preguntas, esta opción se contempla y terminaría el juego con la última cuestión.

EL TORNEO:

Cuando el torneo está cerca de llegar a su fecha, el usuario deberá de estar pendiente de la pantalla aumentando así la tensión por empezar y por ser el primero en leer todas las cuestiones y responder acertadamente y lo más veloz posible. Debe tener los conocimientos y la habilidad necesarios para ganar el torneo.

FIGURA 45: Torneo - Pregunta

Una vez comenzado el torneo, aparecerá una pregunta tras otra con un temporizador ("TIEMPO") con una cuenta atrás que no espera a que el usuario elija una opción. El administrador podrá variar para aumentar la dificultad a su gusto modificando el valor del tiempo como la dificultad teórica de las cuestiones.

Si el usuario elije una opción, usuario solo verá su pantalla con las cuatro opciones deshabilitadas y verá únicamente pasar el tiempo restante hasta la siguiente pregunta. El sistema, por detrás, hará las comprobaciones necesarias; Primero comprobará si este usuario ha sido el primero en contestar. Pueden ocurrir dos casos:

1. **Que el usuario sea el primero en contestar:**

   En este caso este usuario coge el turno y se procede a comprobar que la opción escogida por el usuario sea correcta.

   - En caso afirmativo, se almacena en la base de datos la puntuación pertinente. El usuario en cuestión no sabrá si acertó o siquiera si fue el primero en contestar. Solo lo sabrá cuando aparezca la siguiente pregunta al acabar el tiempo de responder. Lo sabrá mediante un mensaje: "¡Has acertado!". De este modo el estar pendiente de este mensaje hará que pueda despistarse y perder unos segundos muy valiosos.

   - En caso negativo, al ser errónea la respuesta no se almacena ninguna puntuación y pasa el turno al siguiente usuario más rápido. Del mismo modo que en el caso anterior, el mensaje de haber respondido erróneamente "¡Respuesta incorrecta!" aparecerá al inicio de la siguiente cuestión. El paso de turno se consigue mediante el método `synchronized` de java:

```
synchronized(this){
    if (request.getParameter("opcion").equals(session.getAttribute("respuesta"))){
        if(!acierto){
            bbdd.setPuntuacion(idUser, 20);
            request.getRequestDispatcher("Jugar.jsp?jugar=1&&mensaje=¡Has acertado!").forward(request,res);
        }
       else request.getRequestDispatcher("Jugar.jsp?jugar=1&&mensaje=Demasiado tarde...").forward(request,res);
    }
    else request.getRequestDispatcher("Jugar.jsp?jugar=1&&mensaje=¡Respuesta Incorrecta!").forward(request,res);
}
```

FIGURA 46: Código RespuestaServlet.java

2. **Que el usuario no sea el primero en contestar:**

   En este caso el usuario ha contestado una de las cuatro opciones y debe esperar a que los usuarios que fueron más rápidos que él, hayan respondido mal para tener opción a comprobar si es correcta o no la respuesta.

   - En el caso de que le toque su turno pasaría a seguir los pasos del caso 1 (que el usuario sea el primero en contestar).

   - En el caso de que lo te toque el turno, haya contestado bien o mal le aparecerá el mensaje "Demasiado tarde…" y no se almacenará ninguna puntuación. Este mensaje al usuario probablemente le resulte irritante, por lo que para las siguientes preguntas y torneos tendrá más motivación para ser el primero en acertar.

7.2.2. Puntuaciones

Otra de las opciones de la página personal del usuario (véase la FIGURA 40 – Página personal) es la que se va a explicar en este apartado.

Las puntuaciones tienen dos visualizaciones dependiendo de si eres el administrador o no. Si el usuario no es el administrador, al acceder a Puntuaciones aparecerá una tabla con las mejores puntuaciones registradas en la base de datos, ordenadas de mayor a menos. Además esta tabla, para mejorar su visualización, cuando el usuario mueve el cursor por encima se ilumina cada jugador con su puntuación. Siguiendo nuestro modelo MVC el control de acceso a la base de datos lo tiene la clase `EditarPuntuacionesServlet.java`.

En el caso de ser el administrador el que accede a las Puntuaciones, además de la tabla de puntuaciones y usuarios mencionada anteriormente, tiene la opción de borrar todas las puntuaciones o bien, introduciendo un nombre de usuario, borrar sólo su puntuación. En ambos casos aparecerá un mensaje de la que se ha borrado de la base de datos correctamente. Y automáticamente se actualiza la tabla.

FIGURA 47: Puntuaciones

7.2.3. Instrucciones

La opción Instrucciones, de la página personal del usuario (véase la FIGURA 40 – Página personal) simplemente abre un JSP donde se explica detalladamente el funcionamiento del juego.

7.2.4. Salir

Para la opción Salir, de la página personal del usuario (véase la FIGURA 40 – Página personal), si el usuario la selecciona este botón, le aparecerá una ventana emergente pidiendo confirmación para cerrar la sesión. Al pulsar "Aceptar" cierra la sesión del usuario y le devuelve a la página de Inicio (véase la FIGURA 36 – Página de inicio del Juego)

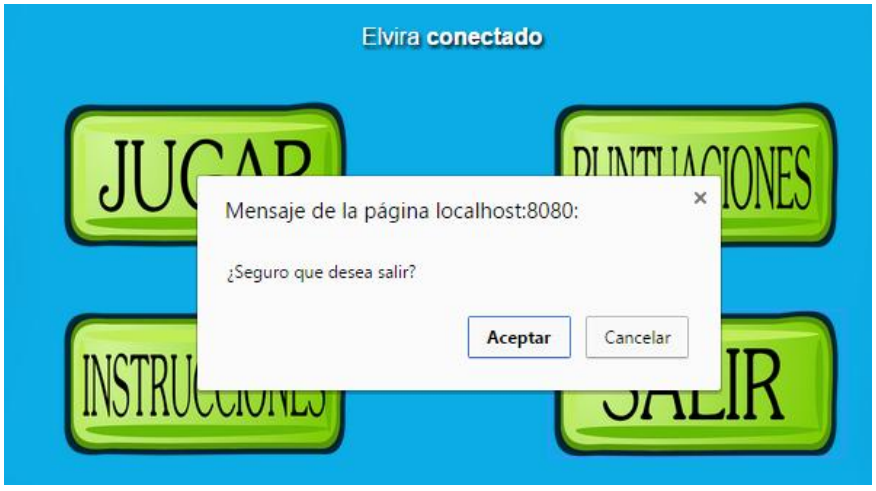


FIGURA 48: Salir

7.2.5. Menú superior

En todo momento, durante la sesión del usuario, aparece un menú con las opciones (véase la FIGURA 40– Página personal):

- Inicio: La cual te lleva siempre a la página inicial
- Cerrar sesión: Cierra la sesión de usuario y te devuelve a la página de inicio de sesión
- Atrás: Sin importar en que página esté el usuario, te da la opción de volver a la página de la que procedía.

Este menú está pensado para facilitar el uso del juego al usuario. Se consigue así que toda la aplicación sea intuitiva sin conocerla de antemano.

7.3. Sesión del administrador.

El administrador, al iniciar sesión como tal, el sistema le lleva a una página exclusiva que solo él puede ver en la que se presentan las diferentes funcionalidades a las que tiene acceso. A continuación se muestra la apariencia de la página de herramientas del administrador y se detallan cada una de ellas:

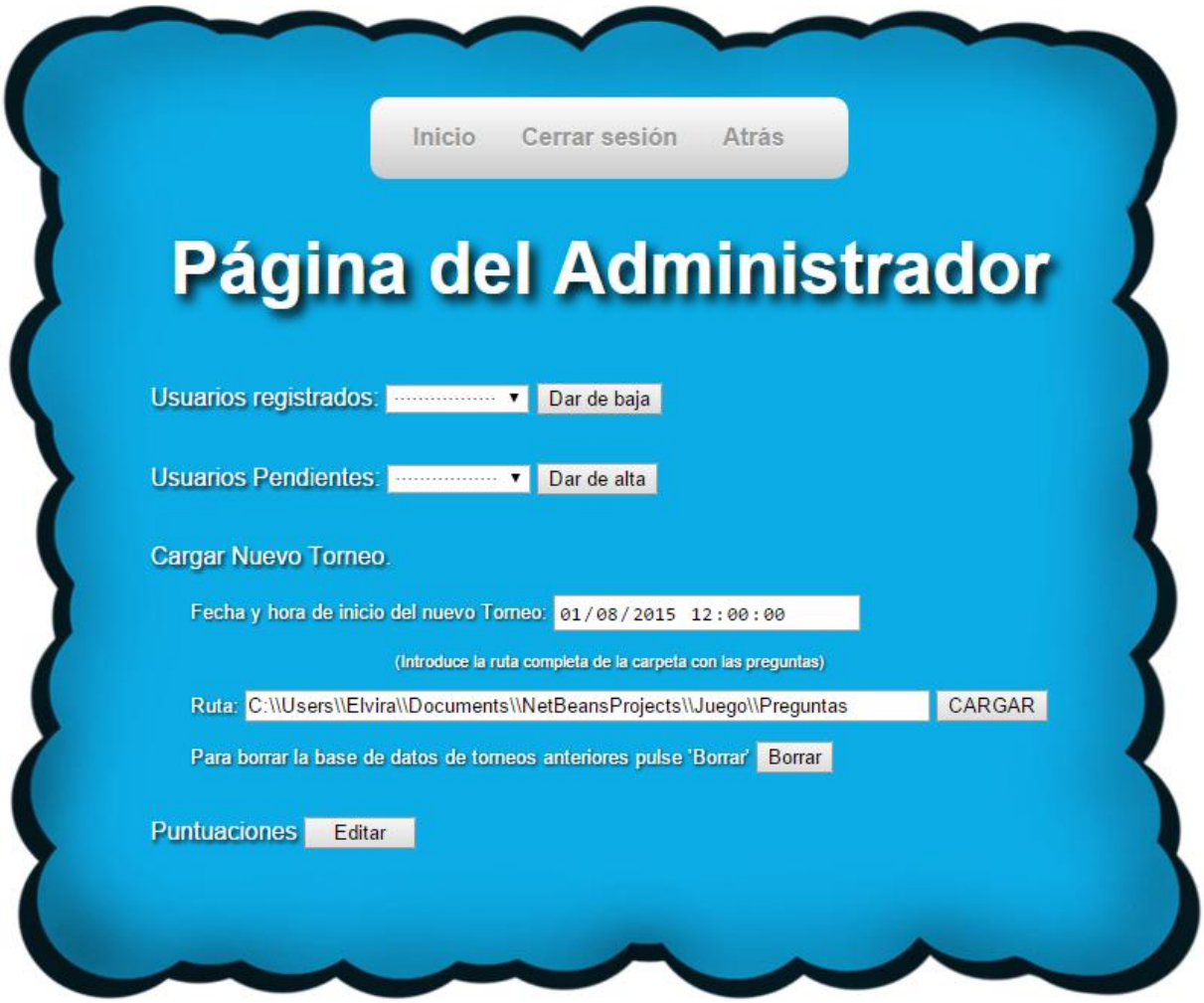


FIGURA 49: Página del Administrador

7.3.1. Menú superior

Es exactamente igual al menú del usuario. Su función es agilizar la navegación del administrador por toda la aplicación.

7.3.2. Dar de baja

En la página del administrador aparece un elemento despegable donde están todos los usuarios registrados y dados de alta ordenados alfabéticamente. A su derecha hay un botón que al acceder el administrador en él, automáticamente se elimina al usuario seleccionado, seguidamente se actualiza la base de datos y el propio despegable.

Para mayor comprobación, el sistema lanza un mensaje de que el usuario dado de baja está eliminado definitivamente de la base de datos.

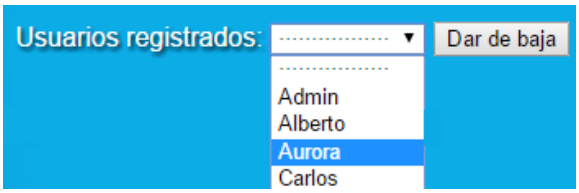


FIGURA 50: Baja

El código de la `clase BaseDeDatos()` para dar de baja al usuario es el siguiente:

```
public void borrarUsuario(String user) throws SQLException {

        Statement stmt = conn.createStatement();

        stmt.executeUpdate("DELETE FROM usuarios WHERE user_name = \"" + user + "\"");

    }
```

7.3.3. Dar de alta

Otra herramienta primordial que tiene el administrador es Dar de alta. Como se comenta anteriormente, es necesario que el administrador de permiso a los usuarios de nuevo ingreso, los cuales permanecen en la base de datos a la espera de ser dados de alta por éste. En este desplegable aparecen los usuarios, ordenados alfabéticamente, que permanecen a la espera y este es el punto en el que el administrador da la orden y puedan pasar a ser usuarios dados de alta. Aparecerá un mensaje del sistema confirmando que el paso del usuario seleccionado de la tabla "peticiones" a la tabla "usuarios" ha sido satisfactorio.

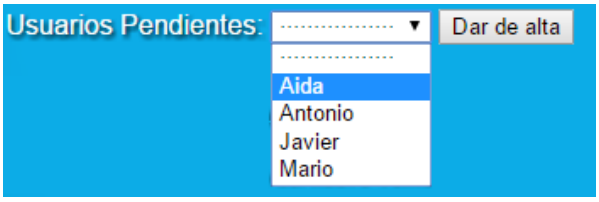


FIGURA 51: Alta

El código de la `clase BaseDeDatos()` para dar de alta al usuario es el siguiente:

```
public void darDeAlta (String user) throws SQLException {

    try{
    Statement stmt = conn.createStatement();
    if(!coincide(user)){
           ResultSet rs =stmt.executeQuery("SELECT peticiones_name,peticiones_pass FROM peticiones WHERE
           peticiones_name = \"" + user + "\"");
            while (rs.next()) {
                this.setUsuario(rs.getString("peticiones_name"), rs.getString("peticiones_pass"));
                 }
           }
    stmt.executeUpdate("DELETE FROM peticiones WHERE peticiones_name = \"" + user + "\"");

    }
```

7.3.4. Cargar nuevo torneo

Este es el cometido principal del administrador: Cargar el torneo. Lo primero que ha de hacer es introducir en el calendario la fecha del torneo y la hora, como muestra la imagen. Después debe escribir la ruta de acceso de la carpeta donde el profesor, en este caso, decida colocar las preguntas de ese torneo en cuestión; Y por último pulsar el botón.

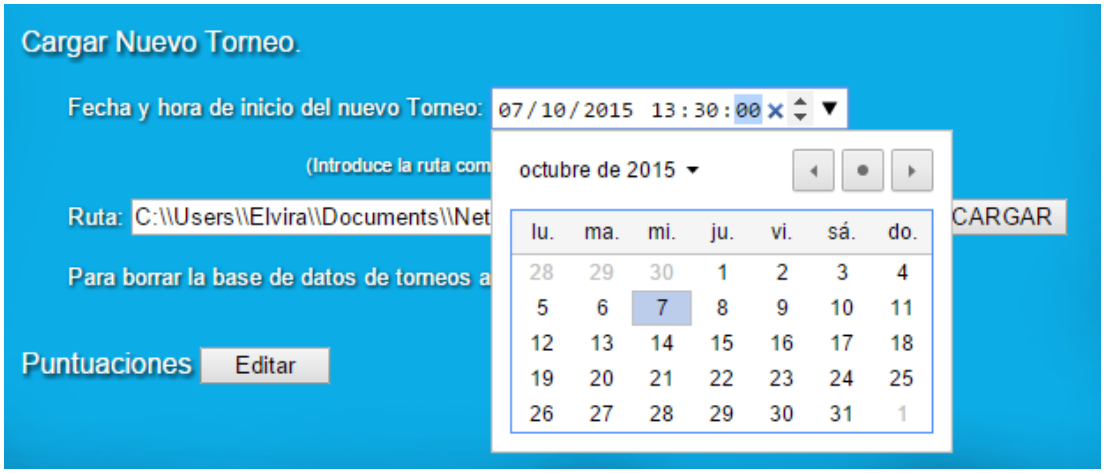


FIGURA 52: Cargar Nuevo Torneo

Cuando se pulsa el botón "Cargar", el sistema recoge la ruta de acceso de los ficheros XML y la fecha y hora.

Con la ruta de los ficheros, mediante las siguientes sentencias se coge la información de cada fichero XML necesaria, se guarda como un objeto `Pregunta` de la clase `Pregunta.java`:

```
fileName = ff[i].getName();  //Nombre del fichero i
pregunta = new Pregunta(new File(ruta+"\\"+fileName)); //Guardar como objeto Pregunta
/* Método getData() recoge la información de
  Enunciado, Opción correcta,
  opción A, opción B, opción C y opción D
  y los guarda en el Vector "datos" */
```

```
datos = pregunta.getData();
```

Código del método getData() (Pregunta.java)

```
public Vector<String> getData() {
    try{
        document = (Document) builder.build(xmlFile);
        rootNode = document.getRootElement();
            List list = rootNode.getChildren();
            Element node1 = (Element) list.get(0);
            List list2 = node1.getChildren();
            Element node2 = (Element) list2.get(0);
            List list3 = node2.getChildren();
            Element node3 = (Element) list3.get(0);
            String correctChoice = node3.getTextTrim();
            datos.addElement(correctChoice);
            Element node12 = (Element) list.get(2);
            List list22 = node12.getChildren();
            Element node22 = (Element) list22.get(0);
            List list23 = node22.getChildren();
            int k=-1;
            for (int i = 0; i < list23.size(); i++) {
                Element node23 = (Element) list23.get(i);
                if(i==0) datos.addElement(node23.getTextTrim());
                else {
                    if (node23.getAttributeValue("identifier").equals
                    (node3.getTextTrim())) k=i;
                    datos.addElement(node23.getTextTrim());
                }
            }
            datos.addElement(""+k);
            return datos;
        }catch (IOException | JDOMException e) {
            System.out.println(e.getMessage());
            return null;
    }
}
```

FIGURA 53: Método getData() de la clase Pregunta.java

En el código anterior se recorren los diferentes nodos hasta guardar el contenido de los que nos concierne con la información básica de cada pregunta: Enunciado, respuesta correcta, opción a, opción b, opción c y opción d.

Una vez guardada en el vector "datos" dicha información, se almacena en la base de datos las preguntas junto con la fecha y hora, para que así cada usuario dado de alta pueda ver en sus pantallas de cuándo será el próximo torneo con el contador de días, horas, minutos y segundos marcha atrás ya explicado anteriormente (véase el punto 6.2.1 caso 2).

Para finalizar este punto, queda sólo mencionar que hay habilitado un botón para borrar toda la información de anteriores torneos guardados en la base de datos, y así poder resetearla y liberarla en cualquier momento.

7.3.5. Puntuaciones

La gestión de las puntuaciones queda detallada para el caso del administrador en el punto 6.2.2

8. Conclusiones y trabajos futuros

8.1. Conclusiones.

Como conclusión, se puede decir que la aplicación web ha sido desarrollada con éxito. Ha cumplido con los objetivos dados.

Este juego cumple con la adaptabilidad necesaria para llevar el juego a diferentes aulas de distintas materias, para cualquier edad y nivel de dificultad. Es un juego rápido y adictivo, de modo que cumple con uno de los objetivos de mayor relevancia del proyecto: que el alumno se divierta y aprenda a la vez. Es un juego que consigue que el alumno no aparte la vista de la pantalla cuando quedan los últimos segundos antes de comenzar, aumentando notoriamente la tensión; y al tratarse de un juego de velocidad, una vez comience la primera pregunta, el alumno queda inmerso en el juego.

La decisión de poner solamente 10 cuestiones es para que el juego sea breve y no agote al alumno en ninguno de los casos. “Lo bueno, si breve, dos veces bueno“. Motivo por el cual la atención será máxima ya que tiene pocas oportunidades. A esto se le suma que el docente podrá visualizar las puntuaciones de cada alumno, por lo que éste tratará de hacerlo lo mejor posible.

La idea principal ha sido hacer un juego educativo competitivo tipo concurso, de preguntas y respuestas, síncrono y en tiempo real. Además se ha seguido la línea de pensamiento que defiende que la educación del ser humano mejora cuando se realiza a través de juegos educativos. En esta premisa se basa todo el proyecto y ha sido cumplida con éxito.

8.2. Problemas y soluciones.

En este apartado se van a mencionar todas las dificultades encontradas a la hora de realizar la programación y las soluciones dadas:

La primera dificultad que aparece al comenzar a desarrollar la aplicación, es la instalación de todos los programas necesarios en el equipo (NetBeans, ApacheTomcat, MySQL, Wampserver, HeidiSQL, WorkBrench Photoshop,…) y conseguir la interconexión entre ellos. Es un proceso un tanto tedioso, pero en cuestión de tiempo todo funcionó correctamente.

Un problema que tuvo este proyecto en concreto fue que no insertaba los datos de los usuarios en la base de datos, pero si se podían leer. Tras varias revisiones del código, se solucionó con la sentencia: *stmt.executeUpdate(sql);* siendo *sql* la sentencia *insert* a la base de datos.

A la hora de registrarse nuevos usuarios, mover usuarios de una tabla a otra y dar de alta, había colisiones de nombres de usuarios, por lo que se procedió a aumentar las comprobaciones en la clase BaseDeDatos para que un nuevo usuario nunca coincida en el nombre con ningún otro.

Una dificultad añadida fue el diseño de las imágenes, ya que precisaban ser sencillas y atrayentes. Esto se solucionó mediante múltiples pruebas y ajustes del fichero CSS[4] y mediante la creación de diferentes botones y fondos de pantalla con Photoshop cs6[5].

Una clara dificultad fue cómo abordar las preguntas del juego. Cómo obtener preguntas con sus cuatro opciones. ¿Habría que añadirlas una a una? ¿Podría utilizarse un formato y habría que crearlo? ¿Se añadirían manualmente a la base de

[4] http://www.w3schools.com/css/: Hoja de estilo
[5] https://es.wikipedia.org/wiki/Adobe_Photoshop : Editor de gráficos

datos? Estas preguntas dan lugar a que si así fuera, sería complicado para el docente la utilización de la aplicación y no cumpliría los requisitos de la misma. Por lo que la solución inminente fue utilizar ficheros XML bajo el estándar QTI ya mencionado en otros apartados. Esta solución es idónea, ya que ya existe un formato XML para cuestiones con varias respuestas, y la aplicación solo deberá poder abordar estos ficheros, que siguen una estructura semejante, y extraer el enunciado, las opciones y la respuesta correcta. Esta solución agiliza mucho la aplicación y da la opción, de que mediante una ruta del lugar donde residen los ficheros XML tipo QTI, acceda a todas las preguntas de una sola carga. Para ello se procedió al aprendizaje de JDOM[6] .

Otro reto en esta aplicación, fue el cómo abordar la sincronización, premisa clave en este proyecto. En concreto, para la cuestión de que todos los usuarios comenzaran exactamente a la vez se solucionó mediante la clase Date7 de Java. Con JavaScript se creó un temporizador de cuenta atrás con un algoritmo matemático que muestra la figura 44 (apartado 7.2.1 Jugar). Para ello se procedió al estudio de JavaScript. Pero la mayor dificultad ha sido cómo conseguir que el alumno más rápido coja el turno y los demás esperen. Esta cuestión se resolvió mediante el código de la figura 46 (apartado 7.2.1 Jugar).

Una dificultad en la ejecución del juego, era que el jugador seleccionaba su opción elegida y el sistema por detrás hacía la gestión del turno y de la base de datos, mientras este esperaba el turno. Hasta aquí todo es correcto, pero el usuario podía volver a seleccionar el botón, pudiendo así mandar otra solicitud de turno al sistema consiguiendo saturarlo. La solución a este fallo fue un JavaScript que lograba desactivar los cuatro botones de las opciones una vez pulsara el jugador

[6] https://es.wikipedia.org/wiki/JDOM: JDOM es una biblioteca de código abierto para manipulaciones de datos XML optimizados para Java. A pesar de su similitud con DOM del consorcio World Wide Web (W3C), es una alternativa como documento para modelado de objetos que no está incluido en DOM. La principal diferencia es que mientras que DOM fue creado para ser un lenguaje neutral e inicialmente usado para manipulación de páginas HTML con JavaScript, JDOM se creó específicamente para usarse con Java

[7] https://docs.oracle.com/javase/6/docs/api/java/util/Date.html

uno de ellos, inhabilitándolas de modo que el usuario solo puede responder una sola vez en cada respuesta y no se satura el sistema.

Una dificultad añadida fue la actualización del software a Windows 10, la cual eliminó los permisos de edición del los ficheros a programar. Se solucionó al encontrar la manera de otorgárselos.

8.3. Trabajos futuros.

Como mejora para avanzar en esta aplicación una buena idea es adaptarla a aplicaciones para los sistemas operativos Android y Apple, de modo que aumentaría más aún el interés de los alumnos.

Respecto a la aplicación que concierne a este proyecto, el paso siguiente sería añadir nuevas funcionalidades:

- Pistas, que el jugador tenga sólo dos pistas y cada una de ellas le oculte una opción errónea. Si necesita más pistas y no le quedan, que tenga la opción de "comprar" restándolo su puntuación.

- Que el alumno pueda generar preguntas para futuros torneos. Si el profesor la considera correcta, la actualiza y el alumno sumaría puntos en su puntuación del juego. De este modo el alumno trataría de hacer preguntas interesantes con varias opciones para superar posiciones en el ranking. Esta opción hace que el alumno esté aprendiendo al plantear nuevas preguntas, a la vez que ayuda al profesor. Si esta pregunta aparece en algún torneo, también

- Añadir otro tipo de juego. Por ejemplo, de forma alternativa al propio torneo, podría haber una sopa de letras y/o crucigrama generado con vocabulario de la asignatura. De modo que mientras quede mucho tiempo para el torneo o aún no haya un torneo cargado, el alumno pueda jugar a estos "mini juegos" o "mini pasatiempos". Todo alumno que lo resuelva se llevará puntos, y podrá acceder a más "mini juegos".

- Generador de preguntas. Darle al administrador una opción para generar preguntas (ficheros XML) desde la misma aplicación.

Otra mejora sería cambiar la interfaz gráfica a una más potente visualmente. Con el único fin de que sea más atractiva para el alumno, pero manteniendo el canon de simplicidad para que la aplicación no deje de ser intuitiva y de fácil manejo.

Esto no sería de gran dificultad ya que el código sigue el patrón MVC. Sería cuestión de mejorar el estilo de los JSP.

Sería recomendable hacer una evaluación con alumnos de una clase, en la que jueguen un torneo en el aula, para medir el grado de motivación, aprendizaje, competitividad y diversión adquirido.

Bibliografía

Las posibilidades educativas de los videojuegos http://www.sav.us.es/pixelbit/pixelbit/articulos/n26/n26art/art2605.htm	Julián Pindado	2005 Málaga
El componente lúdico en las clases de ELE. http://dialnet.unirioja.es/servlet/articulo?codigo=3152470	Charo Nevado Fuentes	2008 La Rioja
Desafíos pedagógicos de la escuela virtual http://dialnet.unirioja.es/servlet/articulo?codigo=3062803	Santiago Castro	2009 La Rioja
Diseño y desarrollo de un juego educativo. Herramienta docente más de apoyo al profesor universitario http://www.redalyc.org/pdf/920/920171890.pdff	Consuelo Giménez Pardo1; Carmen Pagés Arévalo2; José Javier Martínez Herráiz2	2011 Cádiz
UNA APROXIMACIÓN A LA VIRTUALIDAD EDUCATIVA DE LOS VIDEOJUEGOS http://dadun.unav.edu/handle/10171/20643	Charo Sádaba Concepción	2008 Navarra

ProBot: Juego para el aprendizaje de lógica de programación http://www.tise.cl/2009/tise_2009/pdf/1.pdf	Julian Moreno Edgar Alberto Montaño	2009 Colombia
El Juego Didáctico como estrategia de enseñanza y aprendizaje http://www.grupodidactico2001.com/PaulaChacon.pdf	Paula Chacón	Caracas
Breve historia de los videojuegos, atheneadigital.net/article/view/570/437 [Última consulta, 13 mayo 2014]	Simone Belli y Cristian López Raventós	Universitat Autònoma de Barcelona
La historia de los videojuegos, www.elotrolado.net/wiki/Historia_de_los_videojuegos [Última consulta, 13 mayo 2014]	Varios autores	
"Motivation and Emotions in Competition Systems for Education: An empirical study", IEEE Transactions on Education vol. 57, no. 3 (2014), pp. 182-187	Pedro J. Muñoz-Merino, Manuel Fernández-Molina, Mario Muñoz-Organero, Carlos Delgado Kloos	2014
J2EE [artículo en línea] http://es.wikipedia.org/wiki/J2EE	Wikipedia, la enciclopedia libre.	2015
MySQL [artículo en línea] http://es.wikipedia.org/wiki/MySQL	Wikipedia, la enciclopedia libre.	2015
SQL [artículo en línea] http://es.wikipedia.org/wiki/SQL	Wikipedia, la enciclopedia libre	2015

An Architecture for Combining Semantic Web Techniques with Intelligent Tutoring Systems.	Muñoz Merino, P.J.; Delgado Kloos, C.:	Universidad Carlos III. Madrid.
Videojuegos – normativa . Articulo en línea http://www.culture-games.com/a-la-decouverte-de/pegi-ou-la-censure-europeenne-dans-les-jeux-video		2015
Modelo MVC - Articulo en línea https://www.fdi.ucm.es/profesor/jpavon/poo/2.14.MVC.pdf		2014
Beginning J2EE 1.4 : from novice to professional	Weaver, James L.	2004
J2EE Artículo en línea. http://www.it.uc3m.es/~celeste/docencia/ittt/swc0405/	Celeste Campo Vázquez	2005
Beginning Java EE 5 Platform: From Novice to Professional. Apress. Murach, Joel; Steelman, Andrea Java Servlets and JSP (2ª edición).	Mukhar, Kevin; Zelenak, Chris	2008
"Core J2EE Patterns". Sun Developer Network. http://java.sun.com/blueprints/corej2eepatterns/Patterns/index.html.	Sun Microsystems	2002
The Java EE 5 Tutorial: For Sun Java System Application Server 9.1.	Sun Microsystems	2007

Zambon, Giulio; Sekler, Michael Beginning JSP, JSF, and Tomcat Web Development: From Novice to Professional. Apress.	Sun Microsystems.	2007
Java™ Servlet Specification Version 2.3. JSR-53. 8/13/01. Disponible en: jcp.org/en/jsr/detail?id=53.	COWARD, Danny.	2001
SUN MICROSYSTEMS, INC. Java™ 2 Platform Enterprise Edition Specification, v1.4. JSR-151. 11/24/03. Disponible en: java.sun.com/javaee/downloads/index.jsp y jcp.org/en/jsr/detail?id=151.	Sun Microsystems.	2003
ROTH, Mark; PELEGRI-LLOPART, Eduardo. JavaServer Pages™ Specification Version2.0. JSR- 152. November 24, Disponible en: jcp.org/en/jsr/detail?id=152.	ROTH, Mark; PELEGRI-LLOPART, Eduardo.	2003
Java™ Servlet Specification Version 2.3. JSR-154. November 24th, 2003. Disponible en: jcp.org/en/jsr/detail?id=154.	COWARD, Danny; YOSHIDA, Yutaka.	2003
The Java™ Language Specification Third Edition. Serie: The Java™ Series. Disponible en: java.sun.com/docs/books/jls/	GOSLING, James; STEELE, Guy; BRACHA, Gilad.	2004
Java Code Conventions. SSerie: The Java™ Series. Disponible en: http://java.sun.com/docs/codeconv/	Sun Microsystems.	1997

Designing Enterprise Applications with the J2EETM Platform, Second Edition. Disponible en:http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/.	SINGH, Inderjeet et al.	2002
"The Java Web Services Tutorial" http:\\www.java.sun.com\webservices\docs\ea1\tutorial\index.html.	Sun Microsystems.	2004
"WASP Server for Java 5 and UDDI Server" http:\\www.systinet.com.	Systinet.	2004