ÁGORAS: Hacia una Infraestructura para el Aprendizaje Creativo en Superficies Interactivas

Alejandro Catalá Bolós

Tesis de Máster

Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información
Departamento de Sistemas Informáticos y Computación

Director

Dr. Javier Jaén Martínez

Diciembre de 2008, Valencia

Este trabajo ha sido parcialmente financiado por el Ministerio de Educación y Ciencia del Gobierno de España (MEC) en la Convocatoria de ayudas a proyecto I+D 2007. Proyecto TSI-2007-64230.
También ha sido financiada por la beca del programa para Formación de Profesorado Universitario (FPU) del MEC concedida a Alejandro Catalá. Referencia AP2006-00181.

Agradecimientos

A mis padres y hermanos, por su apoyo y comprensión,
porque saben lo importante que esto es para mí.

A mis sobrinos Ricardo Iván, Nerea y Lorena,
que me han dado muchos momentos de alegría.

A Javi, por su paciencia, dirección e instrucción, consejos y su impagable esfuerzo.

A Jose Antonio, José Miguel, David, Paula, Raquel y Víctor,
por su desempeño en el trabajo conjunto y por haber dejado en mí su imborrable
huella a su paso por Futurelab.

A Abel, Carlos, Cristóbal, Elena, Gonzalo, Isabel, Jenifer, Manolo, Nelly y Nour,
gracias por estar ahí y por hacerme pasar tan buenos ratos.

A todos los miembros del grupo de investigación ISSI,
por su atención y colaboración.

A todos aquellos amigos, conocidos y colegas,
que de alguna forma contribuyen a la continua formación que supone en sí misma
la vida.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

CAPÍTULO I

INTRODUCCIÓN

Este capítulo de introducción se centra exclusivamente en la presentación de las principales motivaciones detrás de esta tesis de máster y de sus objetivos directores. De esta forma, la estructura de este capítulo es la siguiente. En primer lugar se describe cuál es la motivación de este trabajo de investigación. En segundo lugar se presentan los objetivos del trabajo. En tercer y último lugar, se presenta brevemente cada uno de los capítulos restantes.

1.1 Motivación del trabajo

Según recoge el informe de 2008 del Ministerio de Educación "Las cifras de la Educación en España. Estadísticas e indicadores." [Min07], cerca de un 30% de los alumnos de Educación Secundaria Obligatoria (ESO) no logra culminar dichos estudios en nuestro país. En palabras de Fernando Reimers (Catedrático de Educación en la Universidad de Harvard) [Per08]

"…a casi 20 años de que la ley propusiera esa reforma educativa es preocupante. Lo es especialmente porque uno de los resultados de la globalización es precisamente dar mayores oportunidades a las personas con mayor nivel de preparación"

Si bien las causas de dicho fracaso pueden ser múltiples y analizadas desde diferentes perspectivas que abarcan el ámbito social, familiar, cultural e incluso el económico, estamos convencidos que uno de los elementos que contribuyen a dicha situación es la escasa aplicación de estrategias educativas activas que permitan a los alumnos españoles desarrollar sus capacidades creativas contribuyendo a incrementar sus niveles de motivación durante los procesos de aprendizaje en los que se ven inmersos. Tal y como reconoce el propio Fernando Reimers "educar no es informar sino desarrollar el talento, y también el carácter de las personas". Uno de los

indicadores que apoya la tesis de que no estamos aplicando de manera efectiva estrategias pedagógicas que conduzcan a los alumnos al desarrollo de habilidades y competencias creativas que les permitan resolver problemas es el reciente informe PISA de la OCDE (2006) [Pis07], el cual no mide el conocimiento escolar como tal, sino la capacidad de los estudiantes de poder entender y resolver problemas auténticos a partir de la aplicación de conocimientos de cada una de las áreas principales de PISA. En este sentido, los últimos resultados de dicho informe para España son demoledores puesto que en todos los indicadores evaluados nuestro país se encuentra por debajo de la media de los países de la OCDE y en tasas de éxito escolar nos encontramos en el último lugar.

Una de las estrategias más efectivas para el fomento de la creatividad es el uso del juego como forma de aprendizaje tal y como plantea, ya en los 60s, Clark Abt. Estos juegos, todavía de índole no tecnológica, acuñados bajo el término general de "juegos serios" por el hecho de que no tratan sólo el divertimiento sino que trabajan principalmente el aprendizaje, permiten la participación activa, la implicación y el compromiso en las tareas, que tan importante es considerado por las teorías constructivistas y experienciales, así como la interacción social que permite la inclusión de todos los individuos persiguiendo objetivos comunes mejorando la experiencia de aprendizaje. Además, Abt va más allá, y considera no sólo el jugar a juegos serios como medio para aprender, sino que considera también la etapa de creación del juego como parte del propio juego, haciendo énfasis en la relevante importancia que esta etapa tiene en el aprendizaje en relación al resto de fases del juego.

En cuanto a los juegos serios tecnológicos que posteriormente se han desarrollado, acostumbran a centrarse en un rango de conocimientos muy específico, para propósitos muy concretos, y no orientado a masas, lo que supone que su desarrollo y explotación es bastante costoso en términos económicos. Una alternativa es el uso de determinados juegos comerciales, pero en general supone la realización de tareas convencionales basadas en los contenidos y/o actividades del juego. No obstante, los estudios que utilizan dicha aproximación introducen un proceso cíclico de suma importancia en el aprendizaje, en el cual se contempla la experimentación, la reflexión, la actividad y la discusión, que los desarrollos de juegos serios no suelen contemplar en su totalidad.

Por otra parte, está claro que cuando “creamos” estamos reforzando el aprendizaje, ya que estamos formando, reproduciendo y mejorando ideas y conceptos, y no sólo estamos adquiriendo conocimientos que nos hayan sido transmitidos. En la Figura 1 se muestran algunas aproximaciones tradicionales al juego que fomentan el aprendizaje. En *(a)* se ilustra la creatividad basada en "papel y lápiz", en la que efectivamente existe creación, y con ellas se puede realizar a posteriori algún tipo de juego basado principalmente en la narrativa. En *(b)* se ilustra otra aproximación

altamente atractiva capaz de mantener la motivación en niveles deseables, como es el caso de plastilinas, arcillas, maquetas, y otros objetos tangibles de modelado. Esta aproximación se centra principalmente en la creación y el posterior juego tangible con las creaciones realizadas. Sin embargo, cuando nos movemos a aproximaciones tecnológicas, como en *(c)*, las posibilidades que las aproximaciones tradicionales ofrecían, y para las que nadie discute su invalorable impacto en nuestro aprendizaje, dejan de soportarse de forma natural. No obstante, sí que aporta otras ventajas, y principalmente introduce a los jugadores en un proceso de alfabetización, eso sí, conocido como *alfabetización digital*, que está siendo cada vez de mayor valor para la sociedad de la información. En la figura se ilustra un típico juego electrónico con una amplia variedad de actividades orientadas al aprendizaje, y una videoconsola moderna. Estos dos ejemplos no soportan en general la creación, aunque recientemente se están haciendo esfuerzos en la industria del videojuego para sacar títulos que empiecen a soportarla[1], y además están muy restringidas a las posibilidades de manejo que ofrecen los mandos de juego.

(a) (b) (c)

Figura 1. Aproximaciones tradicionales concernientes al juego y aprendizaje

Además es de vital importancia tomar en consideración los postulados de las teorías de aprendizaje de corte constructivista, tales como el *aprendizaje experiencial*, y el *aprendizaje situado* [Dew63][Bro89][Kol94], en las cuales "aprender es hacer" y que consideran también la reflexión y la discusión como procesos necesarios de suma importancia más allá de la acción para aprender de forma efectiva, así como también del *aprendizaje social* que considera que el conocimiento emerge de la interacción y la comunicación entre individuos. Todo lo anterior motiva que persigamos la provisión de un entorno de aprendizaje en el que se juegue a la creación de juegos, facilitando así el aprendizaje creativo, aunando muchas de las ideas y aproximaciones anteriores y haciendo especial énfasis en el conocimiento, la creatividad, y la motivación.

[1] Ejemplos sonados de ello son “Spore” y “Little Big Planet”.

1.2 Objetivos del trabajo

El objetivo general de este trabajo es avanzar en la concepción y el desarrollo de un sistema de información avanzado que sirva de soporte tecnológico adecuado para un aprendizaje creativo, social y constructivista. Con este objetivo en mente, y tras un análisis de las tecnologías existentes que detallaremos en el próximo capítulo, los requisitos que necesitamos satisfacer son: 1) proveer de un entorno físico común que facilite la comunicación, que fomente la reflexión, la discusión y el desarrollo de habilidades sociales. 2) permitir la interacción por medio de elementos tangibles, de forma que ésta sea más natural. 3) situar el aprendizaje en un ecosistema interactivo, atendiendo a postulados del aprendizaje experiencial y situado. 4) permitir contemplar la creación del propio ecosistema como parte del juego, dando soporte por tanto a la creación del ecosistema para el soporte de escenarios de aprendizaje lo que implicará el diseño de entidades, acciones, eventos, comportamientos, artefactos tangibles, etc. 5) permitir la simulación del juego diseñado sobre el cual observar las consecuencias de las reacciones de las entidades diseñadas, así como de las acciones de los jugadores.

En nuestra propuesta, y tras un estudio del estado del arte que presentaremos a continuación, abordaremos el uso de superficies interactivas como entornos tecnológicos facilitadores de los requisitos anteriormente enunciados. Éstas no son más que superficies multi-táctil que funcionan a la vez como pantallas de visualización bidimensional. El uso de este tipo de plataforma tecnológica de base permite fácilmente, como veremos, proveer una interacción y reflexión colaborativa, mantener la coexistencia del mundo real y del virtual y utilizar objetos tangibles como forma de interacción.

Dado que el objetivo final, desarrollar una plataforma completa para dar soporte al aprendizaje creativo mediante el uso de superficies táctiles, es demasiado ambicioso para una tesis de máster, los objetivos particulares del presente trabajo no son otros que 1) establecer las bases teóricas y prácticas que deban tenerse en cuenta para la futura construcción de un entorno de aprendizaje creativo, social y constructivista por medio del uso de superficies interactivas, 2) desarrollar una infraestructura software que permita el seguimiento de objetos tangibles sobre superficies interactivas, 3) desarrollar una infraestructura de comunicación basada en eventos que finalmente soporte las reacciones en el ecosistema.

1.3 Estructura del documento

Este trabajo se ha organizado en cinco capítulos. A continuación se describe brevemente el contenido de cada uno de los capítulos restantes:

- Capítulo 2. Juegos Serios para el Aprendizaje Creativo

  El capítulo 2 introduce el trasfondo filosófico que soporta el objetivo de este trabajo, centrándose en la presentación de las principales teorías de aprendizaje, la discusión de los juegos serios como medio para el aprendizaje, y la propuesta para el entorno de aprendizaje creativo, social y constructivista que se quiere alcanzar, identificando los requisitos del mismo y proponiendo los elementos tecnológicos de los que deberíamos disponer para construirlo.

- Capítulo 3. Un Sistema de Realidad Aumentada basado en Marcadores

  El capítulo 3 presenta uno de los primeros elementos identificados como necesarios para alcanzar la realización del entorno presentado en el capítulo 2. En particular, se trata de un subsistema de seguimiento de marcadores que permitirá el seguimiento de objetos tangibles sobre las superficies interactivas. Además, este subsistema se ha desarrollado en términos de marcadores de Realidad Aumentada, de forma que su evaluación preliminar se ha llevado a cabo sobre este tipo de aplicaciones.

- Capítulo 4. Plataforma Semántica para la Gestión de Eventos

  El capítulo 4 trata otro elemento necesario, que consiste de una subsistema de publicación/suscripción de eventos semánticos basado en ontologías OWL-DL, que servirá para la comunicación a bajo nivel de los eventos entre las entidades de los ecosistemas reactivos que compondrán los escenarios de aprendizaje que se diseñen en el entorno de aprendizaje que estamos proponiendo.

- Capítulo 5. Conclusiones y Trabajo Futuro

  Este último capítulo presenta las conclusiones de este trabajo y los trabajos futuros que pretenden abordarse tanto a corto como a largo plazo. También presenta los resultados en forma de publicaciones directamente relacionadas con este trabajo, así como otros resultados en forma de relaciones y colaboraciones externas, impacto social reflejado en la prensa escrita y en televisión.

CAPÍTULO II

JUEGOS SERIOS PARA EL APRENDIZAJE CREATIVO

En este capítulo se pretende introducir el marco filosófico que sustente a largo plazo el trabajo desarrollado en este trabajo así como el concepto de *juego serio para el aprendizaje creativo* que en última instancia se quiere desarrollar utilizando las infraestructuras software que esta tesis de máster presenta.

Para ello, se introduce informalmente al concepto clásico de juego, y el concepto de *Serious Game*, cuya comunidad se ha visto especialmente potenciada en los últimos años.

Tras ello se realizará un breve repaso de las principales teorías de aprendizaje que de una u otra forma subyacen a la mayoría de los diseños de juegos educativos en la actualidad. También se abordarán ciertos aspectos psicológicos a tener en cuenta que se consideran importantes para incidir positivamente en el aprendizaje, sintetizados en la *Teoría del Flow*, así como la perspectiva de lo que se conoce como *aprendizaje creativo*.

Finalmente se procederá a exponer los diferentes soportes tecnológicos para entornos de aprendizaje y en qué contexto se enmarcaría nuestra propuesta basada en tecnología de superficies horizontales, realidad aumentada e interacción mediante objetos tangibles.

2.1 Juegos Serios en la Educación

Con la palabra *juego* se define todo aquel ejercicio recreativo sometido a reglas, en el cual se gana o pierde. El juego se considera tan antiguo como la propia cultura/civilización, e incluso generalmente se percibe como algo natural en la mayoría de las especies animales. Independientemente de sus formas y de la especie, el juego se considera como un vehículo de

transmisión de conocimientos y práctica entre sus participantes pese a que la gran mayoría de los juegos buscan tan sólo el entretenimiento y hacer pasar un buen rato divertido.

Se dice que un juego puede ser jugado, o bien informalmente, o seriamente. Cuando se juega informalmente sencillamente los participantes buscan la diversión, y ésta ha sido la principal motivación para jugar a juegos. Sin embargo, un juego jugado seriamente, típicamente conocidos como *juego serio* (*Serious Games*) [Mic05], significa que éste no está siendo principalmente jugado por entretenimiento. Por tanto, otros motivos están detrás de este tipo de juegos tales como entrenamiento, aprendizaje, educación, etc. Éstos intentan unir el poder de la diversión suministrada por los juegos con el objetivo de obtener un alto grado de implicación en las actividades a llevar a cabo, alcanzando de esta forma el objetivo de transmitir conocimiento y el desarrollar ciertas habilidades en los participantes de forma más efectiva. En los 60s, Abt [Abt70] ya escribió sobre la importancia y la capacidad de los juegos serios en el ámbito de la educación hablando entonces desde un punto de vista no tecnológico y por tanto desarrollando juegos en el sentido tradicional (sin tecnología) que explotasen sus posibilidades.

En los últimos años, la industria del entretenimiento digital (videojuegos) en compañía con ciertos grupos de investigación afines ha crecido para cubrir un sector emergente del mercado cuyos requisitos podrían ser ampliamente cubiertos con el desarrollo e implantación de juegos serios para el entrenamiento, el aprendizaje y la formación. A continuación se citan algunos pocos trabajos y desarrollos, para dar muestra así de la amplia variedad que existe en el área de juegos serios: *The Monkey Wrench Conspiracy* [Monw3] es un tutorial basado en videojuego para el entrenamiento de ingenieros en un software de diseño 3D; el popular videojuego sobre el conflicto palestino-israelí *Peace Maker* [Peaw3] establece el reto de conseguir poniendo al jugador en el rol de líder, poniendo a prueba sus capacidades, y su conocimiento sobre el conflicto; el proyecto *The Space Mission: Ice Moon*, con la ayuda de un videojuego y material multimedia pone a los alumnos en clase en el rol de científicos expertos pertenecientes a un equipo de respuestas de emergencias después de un desastre en el espacio [Gra06]; en eCoology [Aco06] se pretende inculcar a los jóvenes valores de comportamiento, de ecología y sobre alimentación adecuada mediante la asignación de responsabilidades sobre animales que habitan en un entorno de realidad aumentada. Y así podríamos seguir citando a un sinfín de juegos serios específicos que tienen entre alguno de sus objetivos el que los consumidores de los mismos aprendan conocimientos sobre algún tema concreto.

La construcción de software complejo se considera un proceso costoso, y en el caso de juegos no es una excepción. La industria de videojuegos está enfocada a la publicación de productos que van a ser consumidos por

grandes masas a un precio razonable. Sin embargo, el sector orientado a los videojuegos serios está más especializado y el público objetivo real es bastante más reducido, con lo que el coste de estos productos es bastante elevado. Es por ello que toda una línea de investigación ha sido enfocada al estudio del uso de determinados videojuegos comerciales, i.e. paquetes software ya existentes no pensados como juegos serios desde su concepción, en el contexto de una clase evaluando los problemas, ventajas e inconvenientes que estos podrían aportar a la instrucción de los alumnos y por tanto en su aprendizaje.

En ese sentido, varios estudios han sido conducidos para explorar las posibilidades, la aplicabilidad, y las consecuencias de usar videojuegos comerciales en la escuela [McF02][Gro04][Ell06][DeF07]. Aunque es muy difícil caracterizar los métodos educacionales con una única teoría o estilo de aprendizaje, la gran mayoría han sido influenciadas por ciertas variedades de constructivismo [Keb08]. La idea básica del constructivismo, como veremos en el punto 2.2, es aprender por medio de la acción dado que los individuos requieren realizar acciones para conocer y descubrir nuevo conocimiento, y porque el conocimiento no es sencillamente una copia de la realidad sino una internalización de las ideas que se transforman a medida que se entienden y adquieren. Por tanto, aspectos esenciales para el refuerzo y la guía del aprendizaje son la práctica experimental y la reflexión sobre dicha práctica.

MacFarlane *et. al* reportan sobre el uso educacional de juegos en la escuela [McF02]. En dicho informe se expone la evaluación experimental de varios videojuegos comerciales en ambientes escolares formales de acuerdo a un marco de evaluación que ellos mismo propusieron. Los autores dividieron en tres categorías el aprendizaje que típicamente soportan los juegos. La primera categoría se caracteriza por el aprendizaje como resultado de las tareas estimuladas por el contenido de los juegos. La segunda se trata de aquella en la que el conocimiento se desarrolla a través del contenido del juego, aunque en general éste no se ajusta demasiado al contenido curricular. Finalmente, la tercera se caracteriza por conducir el aprendizaje por medio de la práctica en el juego desarrollando de esa forma determinadas habilidades o capacidades. Algunas de estas habilidades son esenciales para el contexto del aprendiz autónomo, tales como resolución de problemas, secuenciación, razonamiento deductivo, memorización, y otras surgen como consecuencia del trabajo en grupo sobre las tareas a realizar, como la tutorización entre iguales, cooperación, colaboración, co-aprendizaje, habilidades de negociación y toma de decisiones de grupo.

Gros en [Gro04] y [Gro07] también discute el uso de videojuegos como material educativo de tal forma que los profesores crean entornos de aprendizaje que permitan la posibilidad de tratar con un sistema complejo multidimensional, multimedia e interactivo. La inclusión del juego en la clase permite al grupo entero de estudiantes trabajar cooperativamente y en

discusiones de grupo que deberían proveer de un espacio adecuado para el análisis y la reflexión crítica. Los autores utilizaron una metodología de cuatro fases compuesta de: experimentación, reflexión, actividad, y discusión. En la fase de experimentación los estudiantes deben tomar notas de las decisiones y resultados que consiguen durante el juego. En la reflexión, al final de cada una de las sesiones de juego, se comparan los resultados y estrategias de cada grupo de estudiantes. En la fase de actividad, ciertas tareas específicas curriculares son realizadas tomando como tema el juego en cuestión. En la fase de discusión, se lleva a cabo una discusión conjunta sobre todas las actividades realizadas, la cual se considera como el proceso más importante en el aprendizaje en este modelo.

El resultado más extendido en todos estos estudios, es que los juegos son artefactos de enseñanza y de entrenamiento efectivos porque son altamente motivantes y pueden comunicar muy convenientemente los conceptos y hechos de muy diversa índole. La mayoría de las actividades políticas, económicas, y sociales implican a los conceptos de participante, regla, procedimiento, éxito, fallo, entre otros. Hasta cierto punto, todos estos ingredientes también aparecen en juegos, y por ello estos son entendidos como una metáfora adecuada para tales actividades de la vida real. De esta manera, los juegos permiten simular estas actividades reales de tal forma que los jugadores puedan explorar, analizar y aprender más sobre ellas en una forma más segura, controlada, y económica. Estos crean una representación dramática del problema real que debe ser estudiado por los jugadores y, hasta cierto punto, asumen roles más o menos realistas, afrontan problemas, toman decisiones, planifican estrategias y obtienen una retroalimentación sobre las consecuencias de sus acciones de forma relativamente rápida.

Como se resume en [Mic05], los beneficios generales de usar juegos serios en la educación son:

- Capacidad para modelar sistemas más complejos.
- Mayor compromiso con el material.
- Ventajas de la interactividad.
- Cercanía a estrategias de aprendizaje fundamentadas en el "constructivismo".
- Ahorro en costes al reducir el tiempo de entrenamiento de los participantes en las tareas reales y además en entornos más económicos que en escenarios físicos reales.

No obstante, todo no son ventajas y también hay algunos inconvenientes que se ponen de manifiesto en los anteriores estudios. En general, los productos

que se utilizan en los anteriores estudios son videojuegos comerciales y eso significa que el contenido de los juegos apenas se ajusta al contenido curricular y además el contenido no puede ser generalmente modificado porque el software es vendido tal cual. La actividad de jugar debe ser estrictamente planificada para dar cabida al limitado tiempo disponible y para centrarse en la parte del juego que mejor se ajuste al propósito de la sesión de juego. Mayormente, los juegos evaluados son juegos de dos jugadores, y a pesar de que existen comunicaciones en línea para los que sí que son potencialmente multijugador, acostumbra a haber una carencia importante de comunicación directa y por tanto la discusión y la reflexión se pospone al final de las sesiones.

Además, los ejercicios propuestos no están relacionados al juego propiamente dicho, sino con tareas convencionales de práctica curricular, sólo que esta vez se basan o están inspiradas en el contenido del videojuego. Como ya hemos comentado anteriormente, la alternativa obvia de diseñar juegos específicos que se ajusten a cada situación del currículum sería sencillamente muy costosa y no viable. De forma similar, como los estudios mencionados arriba, el software educacional convencional podría ser una solución aunque éste cae en un diseño de la actividad claramente de instrucción directa debido al escaso ajuste de contenidos curriculares y la consecuente necesidad de añadir importantes tareas ordinarias para forzar la reflexión y la discusión. En nuestra opinión, las plataformas de juego del futuro para dar soporte a actividades de aprendizaje deberían proveer, atendiendo a lo expuesto anteriormente, escenarios preestablecidos que sean fácilmente editables por los profesores para adaptarlos a los contenidos curriculares, y un espacio real para la experimentación, discusión y reflexión para los profesores y para los estudiantes. Por tanto, la investigación futura en juegos serios para el aprendizaje no debería ser enfocada a la producción de juegos cuyo contenidos se ajusten a todos los posibles temas educativos y estilos de aprendizaje, sino en dar a los profesores y a sus alumnos las herramientas para producir ellos mismos juegos que provean un mejor ajuste, prestando especial atención a los problemas mencionados.

Esta idea no es en sí misma revolucionaria porque Abt a finales de los 60s ya pensó en considerar el proceso de construcción de un juego como una actividad importante desde el punto de vista del aprendizaje. En [Abt70], se destaca que la primera fase del aprendizaje a través de juegos, es decir, la fase de diseño y preparación, puede dividirse en dos tipos de actividades: a) una actividad relativamente pasiva de preparación para el juego activo; b) y el diseño real del juego que va a ser jugado. La primera actividad es más común e implica el aprendizaje del trasfondo del material que tiene que ser simulado en el juego, sus reglas, los roles, los conceptos, etc. La segunda actividad es probablemente una forma más gratificante y provechosa de preparar el juego. En ella el diseñador del juego está realmente inventando un modelo de simulación del proceso que tiene que ser recreado. Durante el

curso de esta actividad, el estudiante debe identificar las variables relevantes implicadas, las relaciones entre ellas, y la dinámica de las interacciones. Para hacer esto de forma satisfactoria se requiere por tanto la comprensión del proceso que tiene que ser simulado. Por tanto implicando a los estudiantes en este proceso aumenta sus conocimientos, aprendiendo no sólo contenido factual sino también los procesos y las interacciones implicadas. Una consecuencia directa de las ideas de Abt es que lo más importante no es qué puede ser aprendido por jugar a juegos sino qué es aprendido al jugar a construir juegos, y en última instancia a jugarlos también.

En los siguientes puntos trataremos algunos asuntos psicológicos y pedagógicos que deben ser tenidos en cuenta antes de continuar con nuestra discusión sobre cómo debería evolucionar la investigación sobre juegos serios, y en particular, para soportar el aprendizaje creativo.

2.2 Teorías del Aprendizaje

Existe una infinidad de teorías de aprendizaje, fundamentos pedagógicos, y estrategias de aprendizaje. En términos generales, por *aprendizaje* se entiende "un cambio más o menos permanente de conducta como resultado de la práctica" [Hil73]. Otra definición más acorde sería que el aprendizaje supone un cambio relativamente permanente del comportamiento que refleja una adquisición de conocimientos o habilidades a través de la experiencia. Algunas de las teorías de aprendizaje muestran postulados totalmente enfrentados, sin que ello suponga que una u otra sea completamente inválida, ya que todo ello dependerá del enfoque, el objeto, y el punto de vista desde el que se trate. De hecho, otras muchas comparten gran parte de los postulados o proposiciones, y sencillamente añaden, retiran o refinan otros supuestos. Eso nos da una idea de la complejidad de las teorías de aprendizaje, que es un hecho razonable dada la alta complejidad de la condición humana y de todos los procesos psicológicos implicados que participan en el proceso de aprendizaje. Por tanto, no pretendemos en este punto dar un análisis exhaustivo de las teorías de aprendizaje, sino proveer el trasfondo filosófico básico y necesario para poder reflexionar sobre determinados aspectos importantes en el aprendizaje y en la educación que nos permita valorar y tomar decisiones sobre el diseño de actividades lúdicas para el aprendizaje.

No obstante, en el siglo XIX y XX varias teorías de aprendizaje fueron más predominantes que otras y son consideradas como las más influyentes, establecidas, aceptadas y destacadas. En primer lugar, la corriente *conductista* (del término en inglés *behaviourist*) propone un modelo en el que el aprendizaje se traduce en la modificación del comportamiento externo o conducta del individuo, el cual se convierte en un mero receptor de

información. En esencia, el conductismo sugiere que el individuo da respuestas ante unos estímulos determinados de modo que cualquier comportamiento puede aprenderse o extinguirse. Lo relevante es que la exposición a los estímulos adecuados condiciona ciertas respuestas de tal forma que se pueda inculcar en el individuo la conducta deseada. Posteriormente, esta idea de condicionamiento clásico [Wat13] evolucionó hacia lo que se conoce como condicionamiento operante [Ski53][Tho13]. El condicionamiento clásico se centró principalmente en estudiar las respuestas innatas, mientras que el condicionamiento operante se centró mayormente en respuestas voluntarias por parte de los individuos. La diferencia radica en la inclusión por parte del condicionamiento operante de estímulos, en forma tanto de refuerzos positivos (recompensa) como negativos (castigo), que permiten instrumentalizar el proceso de aprendizaje para poder inculcar las respuestas deseadas en la conducta, partiendo de la premisa que la conducta que satisface al individuo es la más probable en el futuro. Desde un punto de vista de la instrucción, esta corriente soporta una aproximación de *instrucción directa*. Ésta consistiría principalmente en introducir algún concepto nuevo, y entonces dar la posibilidad a los estudiantes de realizar ciertas actividades para practicar sobre ese nuevo concepto, para finalmente proveerles de retroalimentación con refuerzos positivos o negativos según el caso. Si las respuestas no son correctas se optaría por proveer pistas, o guiar hacia la solución correcta hasta que sea encontrada. Como se plantea en [Joy92] los eventos de instrucción relevantes típicos serían la orientación, la presentación, la práctica estructurada, la práctica guiada, y la práctica independiente. Otra vertiente es la que se conoce como aprendizaje por "modelamiento" o aprendizaje "social", o sencillamente aprendizaje por "observación". La idea es que al observar la conducta de otro individuo se produce un cambio en la conducta propia, y cuando se recurre a dicha conducta adquirida se puede producir un reforzamiento de la sociedad que la haga permanecer o por el contrario un rechazo que la haga extinguirse.

Las teorías conductistas tratan el proceso de aprendizaje como una "caja negra" en el cual los estímulos son las entradas, y las repuestas son las salidas, y no se preocupan del conocimiento propiamente dicho, ni de su tratamiento ni procesamiento. En oposición a esta corriente se encuentran las teorías *cognitivistas* (en inglés *cognitivist*). Se podría decir que las teorías cognitivistas sostienen que la mente no es una caja negra como el conductivismo había tratado hasta entonces, sino que el conocimiento debe tener alguna forma de representación de algún tipo, e.g. simbólica, esquemática, etc., y que por tanto el aprendizaje ya no consiste en inculcar ciertas respuestas a determinados estímulos, sino en internalizar nuevas estructuras de conocimiento en la construcción mental del individuo, generando así una nueva representación ampliada que incorpora el conocimiento previo y el adquirido.

Una de las principales corrientes del cognitivismo es el *constructivismo cognitivo* o simplemente *constructivismo* (en inglés *constructivism*). Esta corriente es realmente amplia [Bru60][Pia67]. En esencia, las teorías de aprendizaje constructivistas postulan que el conocimiento es construido por el aprendiz, y no suministrado por el profesor. Piaget propugna que el aprendizaje ha de venir del interior del individuo, el cual de forma activa construye su conocimiento en base a sus habilidades y su madurez, es decir, aprender es construir, relacionando unos conocimientos con otros. En la misma línea, Bruner enfatiza que durante el aprendizaje se lleva a cabo una asimilación de nuevos contenidos, que podrían ser contradictorios a los ya conocidos por el individuo, y por lo tanto tendría que "acomodarlos" en sus estructuras cognitivas mediante refinamiento u ordenación de ideas. Dicha adaptación consiste en el procesamiento del conocimiento y es especialmente importante por llevarse a cabo su integración en el modelo interno del individuo que permita la asimilación de nuevos conocimientos en el futuro. En cuanto a la instrucción, en el constructivismo los profesores tienden a realizar conexiones a conceptos que los alumnos conocen para promover su entendimiento. Los profesores adaptan sus estrategias educativas de acuerdo a las respuestas de los alumnos, animándoles a analizar, interpretar y predecir, y promueven el dialogo y la discusión. Estas teorías constructivistas se centran en el individuo como una entidad autónoma y "aislada" del entorno y de la sociedad capaz de construir conocimiento. Podríamos decir sencillamente que la regulación del aprendizaje es individual o interna.

Pero ya en la doctrina aristotélica se dice que el hombre es un "animal social". En ese sentido, otras teorías incorporan una perspectiva más ambiental y sociocultural con corte constructivista en la que la regulación del aprendizaje es esencialmente social o externa más que individual.

Desde la perspectiva ambiental, la teoría del *aprendizaje experiencial* (*experiential learning*) es una de las más destacadas. Durante el aprendizaje experiencial los profesores intencionadamente comprometen a los estudiantes en experiencias directas y se centran en la reflexión sobre las observaciones para aumentar el conocimiento, habilidades, y valores. La experiencia sucede como resultado de las interacciones entre seres humanos y el entorno en diferentes formas como pensar, sentir, manejar y hacer [Dew63]. De esta forma se sugiere que el conocimiento se construye, no se transmite, como consecuencia de la experiencia y la interacción con el entorno. La teoría experiencial de Kolb [Kol94] propone que el aprendizaje experiencial tiene lugar cuando el alumnado observa y reflexiona sobre una experiencia previa y realiza algún tipo de abstracción integrando esas reflexiones en sus conocimientos previos utilizados así como guías para acciones posteriores. De esta forma, el aprendizaje experiencial describe la adquisición de conocimientos en un ciclo de aprendizaje de cuatro etapas sucesivas (ver Figura 2), en el que las experiencias son traducidas en

conceptos por medio de la reflexión, y estos a su vez, son usados como guías para la retroacción o experimentación activa y la planificación que permitirá seguir adquiriendo más conocimientos por medio de más experimentación. Estas teorías sostienen cuatro principios básicos: el aprendizaje implica participación en el mundo real; existen relaciones íntimas entre experiencia y educación; el entendimiento se deriva de y se modifica por medio de la experiencia; el aprendizaje significativo consiste de la acción y la reflexión.

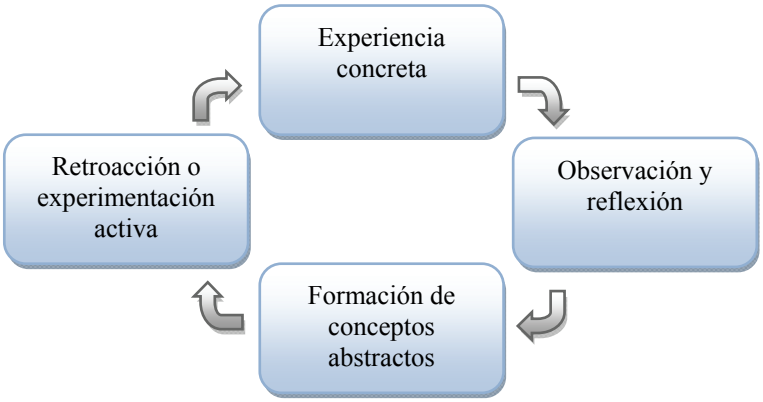


Figura 2. Aprendizaje experiencial según Kolb

Desde la perspectiva social, la teoría más influyente es sin duda *la teoría del desarrollo y del aprendizaje social* de Vygotsky en la que se enfatiza que la comprensión emerge a partir de la interacción entre individuos y su contexto sociocultural. Vygotsky habla de la mediación, donde diferentes categorías lingüísticas y la interacción entre individuos mediante objetos físicos permite la comprensión y la interpretación del mundo real [Vyg78]. Sobre las teorías de Dewey y Vygotsky surge lo que se conoce como "cognición situada" (*situated cognition*). La *cognición situada* sugiere que el conocimiento se sitúa en su contexto, es decir, el conocimiento en cierta medida es el producto de su contexto, actividad y cultura dentro del cual éste se desarrolla y es usado [Bro89]. En general la instrucción bajo estas teorías se caracteriza por tratar de adaptar en todo momento la dificultad de las actividades así como la ayuda proporcionada y prestada entre los alumnos y los profesores (andamiaje) para permitir el progreso en las actividades que por sí solos los alumnos no podrían abordar, e incrementar la interacción entre los individuos ya que en cierta medida todos persiguen objetivos similares por el hecho de pertenecer a una comunidad.

Desde una dimensión psicológica, independientemente del método de instrucción y de las teorías de aprendizaje subyacentes, la motivación se revela como un factor de indudable importancia en los procesos de aprendizaje implicados. El estudio es una de las tareas principales tanto para niños, adolescentes e incluso universitarios de cualquier cultura al que se dedica una parte significativa de la actividad diaria [Bro02]. Los alumnos han de desarrollar dicha actividad con otros compañeros de clase en presencia de sus educadores siguiendo reglas sociales de comportamiento

definidas por adultos y que en un gran número de casos resultan en modos de interacción en los que el alumno no controla la situación y que están caracterizados por bajos niveles de motivación [Del02]. Este es el típico caso de las lecciones magistrales en las que el profesor desarrolla un discurso haciendo uso de la pizarra o de otro material de apoyo como las transparencias y en las que el alumno se limita a responder preguntas a iniciativa del profesor en unas condiciones que en ningún caso pueden considerarse ideales para el aprendizaje.

Por el contrario, diversos experimentos demuestran que las experiencias más positivas son aquellas que tienen lugar durante la práctica de deportes, juegos, actividades artísticas y hobbies que incorporan en la justa medida la diversión, la concentración y el establecimiento de objetivos [Ver03]. Particularmente relevante en estas actividades es la identificación de lo que se ha venido denominando como experiencia óptima o "flow" [Csi88] que se caracteriza por la percepción de retos en el entorno y la existencia de capacidades personales adecuadas para alcanzarlos, de altos niveles de concentración, de disfrute y compromiso, de inmersión o pérdida de la conciencia propia, de control sobre la situación, de atención focalizada, de realimentación positiva, de motivación intrínseca y de ideas claras sobre los objetivos de la actividad [Dec85]. Este aspecto psicológico está siendo estudiado por la llamada *Psicología Positiva* que pretende el estudio científico de las fortalezas y virtudes humanas. La situación de "flow" la podríamos describir informalmente como aquella en la que un sujeto alcanza un estado en el cual se halla tan involucrado en la tarea que nada le parece más importante, con altos niveles de interés, atención, y satisfacción, sintiendo a la vez una sensación de progreso en la productividad, pero de no percepción del paso del tiempo, llamada en ocasiones "falta de conciencia propia". Según Csikszentmihalyi, la experiencia óptima requiere cierto equilibrio entre los desafíos o retos percibidos y las habilidades del sujeto (ver Figura 3). Cuando ese equilibrio no puede ser alcanzado, el sujeto puede experimentar dos sensaciones igualmente indeseables para el proceso de aprendizaje. Si los retos percibidos superan las habilidades del sujeto, éste entra en un estado de ansiedad debido al exceso de dificultad, ya sea real o percibida. Por otra parte, si las habilidades superan con creces los retos, resulta en una sensación de aburrimiento que deriva en una escasa motivación por parte del sujeto. Además del equilibrio entre retos y habilidades, el sujeto debe encontrar la tarea interesante, las metas deben estar claramente definidas, sentir que tiene suficiente tiempo para realizarla, obtener retroalimentación inmediata, sentir el control sobre las acciones, y por supuesto poder concentrarse sin excesivas interrupciones. Por lo tanto, para la continua realización del "flow", y consecuentemente para que la actividad llegue a ser motivada intrínsecamente, es necesaria la introducción regular de nuevos retos a medida que se vayan adquiriendo más habilidades para seguir desarrollando y adquiriendo otras nuevas y mantener el resto de factores en los niveles de percepción adecuados.

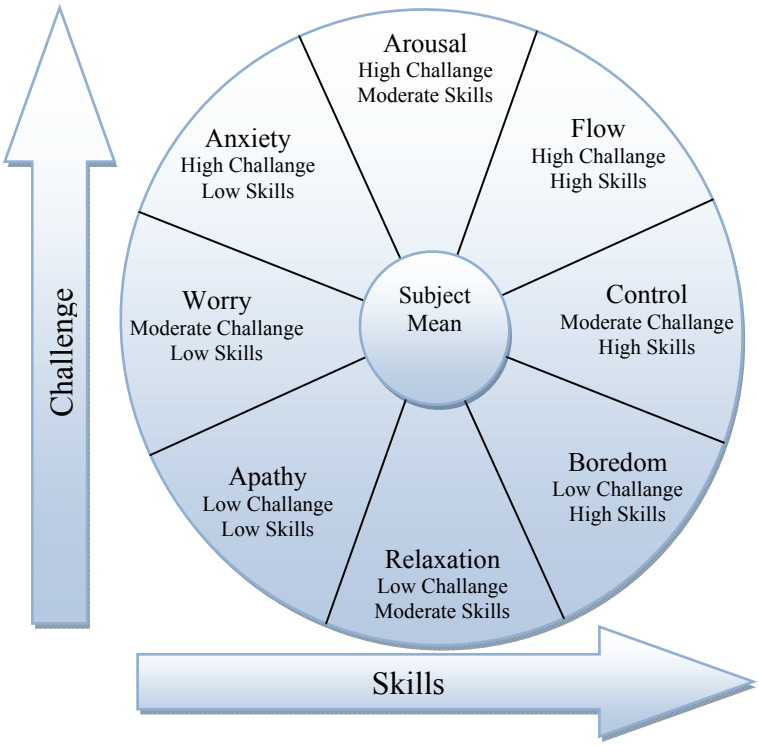


Figura 3. Teoría del "Flow"

En torno a las teorías activas constructivistas, experienciales y situadas, ha surgido con notable fuerza la idea de *aprendizaje creativo*. La creatividad se define como la capacidad o la facultad de crear. Sin embargo, esa definición de diccionario no revela la complejidad inherente asociada el término "creatividad". Para hacernos una idea de la problemática que lleva consigo el término y los diferentes puntos de vista tan sólo hay que considerar algunos estudios previos. Por ejemplo, en [Tre96] se presentan más de 100 definiciones encontradas en la literatura, y en [Ale00] se discuten 101 definiciones actuales proporcionadas tanto por niños como por adultos. A pesar de esta variabilidad y de que el estudio de marcos teóricos que definan la creatividad es una línea de investigación todavía en desarrollo, en [Tre02] se proporciona una detallada categorización entre la que destaca la propuesta por Teresa Amabile [Ama83] por su simplicidad y por su capacidad para integrar aspectos propuestos de forma separada en otras aproximaciones. Según Amabile, la creatividad emerge como consecuencia de la confluencia de tres factores: conocimiento, pensamiento creativo y motivación.

El *conocimiento* es toda la información que los individuos poseen para abordar la resolución de un problema. Gardner identifica dos tipos de conocimiento, el relacionado con la comprensión profunda de un determinado dominio (conocimiento en profundidad) y el relacionado con un entendimiento más superficial pero de múltiples y variadas áreas (conocimiento en amplitud) [Gar93]. La combinación equilibrada de ambos

tipos de conocimiento es fundamental para maximizar el potencial creativo del individuo.

Por otra parte, el *pensamiento creativo* puede resumirse como la presencia en mayor o menor grado de las siguientes capacidades en el individuo: la capacidad de estar en desacuerdo con los demás y experimentar con soluciones dispares a las propuestas por la mayoría; la capacidad de combinar conocimiento existente en áreas separadas; la capacidad de perseverar ante situaciones difíciles o problemáticas; y finalmente, la capacidad de incubación de ideas durante periodos en los que nos apartamos del problema para volver a él con una perspectiva novedosa. En este sentido, podríamos concluir que el pensamiento creativo se desarrolla mediante la facilitación de procesos de discusión colectiva, reflexión y experimentación de nuevas ideas.

Finalmente, la motivación es considerada por un gran número de investigadores como el factor más importante que contribuye al desarrollo creativo, y tal y como ya se describió con la Teoría del Flow, tiene una gran importancia la motivación intrínseca entendida ésta como la dirigida por el interés, la satisfacción y los propios retos del trabajo a desarrollar para resolver un problema, y no tanto por la existencia de "presiones externas" o motivación extrínseca.

En definitiva, el aprendizaje creativo persigue el desarrollo de habilidades así como la adquisición de nuevo conocimiento incentivando la innovación, la generación de ideas y su exploración. En [Ste99] se defiende la idea de que las nuevas técnicas pedagógicas para el desarrollo y evaluación del pensamiento creativo deben incidir en los siguientes aspectos: reducir el número de reglas que limiten o controlen los procesos de pensamiento pues el establecimiento de un gran número de restricciones limita la motivación intrínseca, favorecer al máximo las oportunidades de elección tanto en la definición de la naturaleza de los problemas a resolver como en la formulación de soluciones y experimentación de las mismas, involucrar a los alumnos en la evaluación de su propio trabajo y hacer que no teman al fracaso y que comprendan que en las situaciones de fracaso de las ideas propuestas aparecen las fuentes de generación de nuevo conocimiento, favorecer la colaboración creativa entre diferentes sujetos, buscar entornos de aprendizaje estimuladores, facilitar la expresión explícita de los puntos de vista y soluciones de los demás, habilitar mecanismos para la combinación de ideas y para la crítica.

En los párrafos anteriores hemos hecho un repaso general de las principales teorías de aprendizaje, así como también del impacto de la motivación en la psicología del aprendizaje, y las principales dimensiones consideradas en el aprendizaje creativo. Como conclusión se podría decir que su evolución ha sido conducida hacia teorías más y más complejas que introducen un mayor número de variables y factores, por ejemplo las socioculturales, con el

propósito de dar cabida a un mayor rango de perfiles de los individuos, áreas de aplicación y sobretodo buscando su eficacia. En ellas se consideran muy importante las tareas de reflexión y discusión como medio para construir nuevos conocimientos, y la interacción, ya sea con los artefactos y conceptos, con los participantes en las actividades, o con el entorno y contexto, y desde la perspectiva psicológica de la teoría del *flow*, se persigue el mantener en equilibrio las capacidades y los retos presentados a los alumnos para hacer óptima la experiencia de aprendizaje.

2.3 Soportes Tecnológicos para el Aprendizaje

A lo largo de la historia hemos presenciado que a medida que la ciencia y la técnica han evolucionado, ésta ha llevado consigo cierto desarrollo tecnológico que ha propiciado una revolución. Una revolución que no sólo tiende a mejorar lo existente sino a cambiar completamente la forma de proceder en los diferentes aspectos impactados por la misma. Podríamos decir que no complementan lo ya existente sino que lo sustituyen con la nueva tecnología, aunque evidentemente siempre quedan tradiciones y vestigios de lo establecido anteriormente. Por ejemplo, con la revolución industrial en el siglo XVIII y XIX, nuevas tecnologías y mecanismos potenciaron el uso de máquinas. En particular, pensemos en la máquina de coser, que permitió la producción masiva sin que ello implicase coser todo a máquina. Por supuesto, su introducción no supuso que la gente no tuviera que coser a mano, pero desplazó ese trabajo a tareas residuales[2] y que resulta casi imposible que la máquina pueda acometerlas. Otro ejemplo quizás más sonado sea el de la imprenta de Gutenberg en el siglo XV. La introducción tecnológica de la imprenta en Europa permitió la divulgación de conocimiento en forma de libros masivamente a mucha mayor velocidad y a un mayor público de lo que se había hecho hasta entonces a través de los copistas y religiosos. Sin embargo, con la aparición de la tecnología informática, no acaba de producirse una revolución a nivel de enseñanza y aprendizaje. Es decir, lo que se hacía con el papel y lápiz, sigue haciéndose así, y la introducción de tecnología ocupa un segundo lugar de apoyo o de enriquecimiento para tratar de mejorar los procesos de aprendizaje. Quizás, como se expone en [Res02], de la misma forma que la aparición de forma masiva de los ordenadores y de la Web han dado como resultado la *sociedad de la información* y posteriormente la *sociedad del conocimiento*, sea necesario dar un paso más allá sobre el papel que deben desarrollar las tecnologías en la educación para poder dar paso a lo que se denomina la *sociedad creativa*. Lo que está claro es que los ordenadores, en su forma más

[2] Se refiere en términos relativos ya que se pasa de coser prácticamente todo a mano, a coser a mano sólo aquello que no puede ser realizado con ayuda de la máquina.

variada, no sólo PCs, pueden ser vistos como materiales de construcción, extendiendo lo que la gente puede crear y lo que ellos pueden crear en ese proceso [Res98]. Las investigaciones han mostrado que muchas de nuestras mejores experiencias de aprendizaje vienen cuando estamos implicados en el diseño y creación de "cosas", especialmente con las cosas que tienen algún significado para nosotros o para otros en nuestro entorno [Pap93]. Antes de precisar el concepto hacia el cual queremos movernos para formar parte del tejido que contribuya a la construcción de la sociedad creativa analicemos cuál ha sido la evolución de la tecnología en el ámbito del aprendizaje.

Toda taxonomía o clasificación viene determinada por una dimensión principal que guía el desarrollo de la misma. Por ejemplo, en [Kin02] se muestra una posible clasificación de los sistemas de *elearning* atendiendo a la naturaleza de la actividad a realizar dando como resultado al aprendizaje típico mediante medios electrónicos asíncronos, basados en escenarios, en simulaciones, y por último en juegos, pero dicha clasificación para nuestros propósitos no es demasiado específica. En este punto sería más idóneo, sin embargo, realizar una taxonomía atendiendo a criterios como las teorías de aprendizaje y las estrategias de instrucción subyacentes de forma similar a como se realiza en [Keb08]. No obstante, dada la orientación altamente tecnológica del presente trabajo, estamos más interesados en repasar y reflexionar sobre los diferentes soportes "tecnológicos" que tanto tradicionalmente como de forma innovadora han sustentado las aplicaciones de aprendizaje. Hemos considerado las siguientes categorías tecnológicas relevantes:

- ***Sin "tecnología"***. En esta categoría se encontrarían todas aquellas actividades y juegos pensados para el aprendizaje en cualquiera de sus formas tradicionales. Desde la infancia hemos participado activamente de este tipo de experiencias y forman parte de la etapa inicial de nuestro aprendizaje. Por ejemplo, en actividades más plásticas o artísticas normalmente el objetivo principal es el de "crear", desarrollando ciertas habilidades tanto manuales como mentales asociadas a la actividad. En actividades más físicas y también más sociales, el objetivo es el de participación en grupo, cooperación, competición, respeto a las reglas establecidas y aceptadas, valoración de metas comunes frente a las metas individuales, etc., tratando de inculcar de esta manera ciertos valores socioculturales y fomentar la confianza, estima, contacto, comunicación, convivencia y discusión entre los participantes. La no existencia de tecnología impide la observación de comportamientos de entidades no reales en ecosistemas visualmente atractivos. Este hecho tiene implicaciones en el grado de motivación intrínseca del alumno y en la capacidad y productividad para analizar de forma crítica mediante experimentación los resultados de las decisiones tomadas.

- ***Recursos didácticos multimedia***. Esta categoría es bastante simple, y típicamente se entregan recursos en soportes físicos como CDs o DVDs, que contienen todo tipo de material multimedia diseñado para el aprendizaje, tal como recursos de audio y/o vídeo, presentaciones multimedia, y documentos. Se caracteriza por ser recursos que los usuarios deberán procesar y en general se dice que serán considerados como simples receptores de información, no desarrollando por lo general ninguna actividad de refuerzo o evaluadora del aprendizaje.

- ***Aplicaciones educativas multimedia***. En primera instancia es la evolución natural de los clásicos recursos multimedia. Los recursos son integrados en una aplicación de forma amigable y cohesionada pudiendo reproducirse. Este tipo de soportes tecnológicos permiten un mayor grado de sofisticación de las actividades a realizar ya que en general se diseñan para ser utilizados en un PC con teclado y ratón. Permiten la inclusión de interactividad, la posibilidad de definir caminos de aprendizaje que el usuario deberá respetar, evaluación de su avance y proveerle retroalimentación inmediata, orientar las actividades no sólo a la recepción de información sino también a la realización de ejercicios, resolución de problemas, escenarios, juegos, etc. Pese a la diferencia cualitativa que supone esta tecnología respecto a los recursos didácticos multimedia siguen siendo una tecnología centrada en el individuo, y las técnicas de interacción para permitir la práctica de conceptos y su reflexión es limitada. Aunque permite enfocar el aprendizaje hacia la construcción de conocimientos a partir de la creación de artefactos, lo cual es un elemento importante para ser creativo, la interacción se limita a la relación de un usuario, o en el mejor de los casos dos compartiendo los mismos dispositivos de interacción, con la aplicación. Esto dificulta por un lado, el establecimiento de conocimiento en amplitud, la colaboración creativa entre diferentes sujetos y la expresión explícita de los puntos de vista y soluciones de los demás lo que limita la diversidad en el pensamiento que haga potenciar el aprendizaje por medio de la exploración de ideas.

- ***Basados en Web***. En esta categoría tecnológica yacen la mayoría de las plataformas para el aprendizaje puramente instruccional y formal que hoy en día se encuentran en producción. Su funcionalidad depende mucho de qué plataforma se esté considerando. Por ejemplo, la evolución de los recursos didácticos hacia el marco tecnológico basado en web serían los llamados repositorios de objetos de aprendizaje o incluso de contenidos en general (en este caso sólo cambia el continente de los recursos). Los sistemas de gestión del aprendizaje van un poco más allá y suelen incluir opciones de seguimiento del progreso de los usuarios por parte del tutor, expresar

sus opiniones e impresiones con sus compañeros y permitir la discusión de las actividades dando así soporte en cierta medida al pensamiento crítico y divergente. Sin embargo, las actividades están en general basadas en la instrucción clásica y, cuando están orientadas a la creatividad, de nuevo se plantean como ejercicios de carácter individual. Existen excepciones como las aplicaciones web colaborativas pero en las que dada la no co-presencia física de los participantes en la actividad los mecanismos de discusión, crítica y aporte de nuevas soluciones son en general primitivos y basados únicamente en información textual tanto síncrona mediante "chat" como asíncrona mediante blogs, redes sociales, wikis, comunidades y foros. De nuevo, se plantean problemas similares a los discutidos en las aplicaciones multimedia offline desde el punto de vista de las estrategias pedagógicas creativas levemente mitigados por la existencia de ciertos niveles de comunicación entre los participantes.

- ***Mundos Virtuales***. Esta categoría es sin duda de las más importantes en la actualidad ya que en ellas se reflejan la mayoría de las plataformas atractivas e interesantes para los usuarios, y además representan un sector del mercado de la industria importante. En esta categoría tienen cabida los videojuegos (los que no pertenecen ya a la categoría de aplicaciones educativas multimedia que ya hemos visto que son más restringidas), las simulaciones que acostumbran a acontecer en mundos virtuales, mundos virtuales 3D, etc.

  Aunque el concepto de mundo virtual 3D y la idea romántica de "Metaverse" que Neal Stephenson describió en su libro de ciencia-ficción "Snow Crash", en el que los humanos por medio de sus avatares interaccionaban entre ellos y con agentes software en un mundo tridimensional como metáfora de la realidad, ya lleva muchos años en marcha, no fue hasta mediados del 2000 cuando se dieron todas las condiciones necesarias, tanto técnicas como sociales, y se produjo el lanzamiento de Second Life causando un gran impacto mediático en nuestra sociedad [Secw3]. Second Life se define a sí mismo como un mundo virtual 3D creado por sus residentes. Ofrece un alto grado de interacción social y de posibilidades creativas del mundo del que se es propietario pudiendo exhibir cualquier tipo de creación y poner a disposición del resto de usuarios objetos virtuales interactivos, juegos, actividades, etc. Bajo esta misma idea existen iniciativas libres para la creación de mundos virtuales [OSiw3][Olgw3] basados en la misma idea de simulación [OSiw3]. Esto significa que este tipo de plataformas no son precisamente concebidas para el aprendizaje, pero potencialmente pueden llegar a serlo dado que ofrecen un amplio abanico de posibilidades. De hecho, se está potenciando la entrada de instituciones para su uso en el ámbito del aprendizaje, para proveer lugares de encuentro, discusión

síncrona y asíncrona, creación y manipulación de artefactos, difusión de recursos educacionales, etc., dentro del propio mundo virtual. Estos entornos educativos necesariamente resultan atractivos y motivadores para los participantes pero dada su naturaleza tridimensional y virtual plantean dos problemas respecto al aprendizaje creativo. En primer lugar, la naturaleza puramente virtual de dichos entornos desacopla a los participantes del mundo real y físico en el que se producen de manera efectiva las discusiones, y el análisis de las propuestas creativas de los diferentes participantes y, en segundo lugar, la naturaleza tridimensional de los elementos del mundo virtual hace difícil el modelado por parte de los usuarios no expertos de entidades que formen parte de la solución y que exhiban un cierto comportamiento en dichos mundos virtuales. En este sentido se están realizando avances mediante propuestas creativas como el reciente juego para la plataforma PlayStation 3 denominado "Little Big Planet" en el que se deja a los usuarios concentrarse en un reducido número de comportamientos definidos a priori sin necesidad de tener que modelar nuevas entidades que participen en la acción. Si bien estas propuestas plantean un cierto avance en términos de aprendizaje creativo, siguen limitándose a actividades individuales con escasa participación de múltiples agentes donde se pueda favorecer los procesos de crítica, divergencia y posteriormente convergencia. Además, en este y otros juegos similares, la línea argumental está predefinida en forma de secuencia de problemas a resolver y en los que la solución aunque se obtenga de manera creativa es única y definida a priori.

- ***Móviles***. Con la llegada de dispositivos móviles y su popularización, en forma de teléfonos celulares, PDAs, y UMPCs surge la posibilidad de migrar algunas de las aplicaciones educativas ya existentes a este tipo de tecnología. Sin embargo, las posibilidades de ésta da para mucho más que sólo adaptar recursos, y aplicaciones a la restringida interfaz de estos dispositivos. Dependiendo de las características de la conexión y el ámbito de uso, las posibilidades del aprendizaje móvil (*m-learning*) pueden diferir. El diseño de las actividades pueden incorporar un aprendizaje basado en la localización y el resto de información de contexto que pueda haber disponible, se puede contemplar el uso de redes sociales en busca de la cooperación tomando el sentido natural de la movilidad, y se proveen de mecanismos de entrega de información "just-in-time". Un ejemplo notable de esta categoría es *Savannah* [Fac04] donde los participantes mediante el uso de dispositivos móviles aprenden el comportamiento de los animales de la sabana teniéndose que mover y actuar en un espacio físico real como si se tratasen de animales en una sabana real. De esta forma emergen de manera natural soluciones que son diseñadas, discutidas y experimentadas por un

colectivo de participantes siguiendo los principios del aprendizaje creativo en los que los alumnos generan y discuten ideas de forma colectiva y dinámica sin las barreras que proporcionan las tecnologías basadas en la web o en los mundos virtuales. Las limitaciones de este tipo de entornos vienen dadas por las propias limitaciones de interacción con los dispositivos y por el hecho de que la información de naturaleza virtual no puede ser fácilmente manipulada de forma colectiva puesto que ésta es presentada en dispositivos móviles independientes para cada participante. Esto limita en gran medida el tipo de acciones creativas que se pueden desarrollar mediante el uso de estas tecnologías.

- ***Realidad Aumentada***. La realidad aumentada va más allá del puro mundo virtual y combina las características de la movilidad para crear un espacio real aumentado con objetos digitales. Esta tecnología aboga por una interacción más real y física, tanto con las entidades físicas como reales que coexisten en el espacio aumentado, y para ello introduce el uso de interfaces y objetos físicos que proveen una retroalimentación más realista. Además, el hecho de interaccionar con objetos físicos tangibles que permiten disparar mecanismos más naturales y propios de los humanos para el aprendizaje, tal y como Vygostky comenta, potencia el aprendizaje por parte de sus usuarios. Además, se incorporan todas las características que ya se habían desarrollado con la realidad virtual y los mundos virtuales, y en la medida de lo posible se integran también las ventajas que ofrece la movilidad cuando se visualizan los entornos aumentados sobre dispositivos móviles.

- ***Robóticos y mecánicos***. Esta categoría se centra principalmente en la tecnología basada en dispositivos electrónicos programables dotados de sensores y actuadores (robots). De especial relevancia ha resultado el concepto de bloques programables y la combinación en ellos del lenguaje Logo y el kit de construcción LEGO [Res88][Res96] que han inspirando los kits de construcción *PicoCricket* [Picw3] que se centra en la creaciones artísticas con luces, sonidos, música y movimiento, y *Lego MindStorms NXT* [Legw3] que se centra más en la construcción de robots. *Lego MindStorms* es la familia de dispositivos más popular debido a su facilidad de uso y a su diseño en forma de componentes que pueden ser fácilmente ensamblados y programados. Otra sonada iniciativa educativa y de investigación del uso de robots es la popular *RoboCup* [Robw3] en la cual se organizan diversas competiciones en sus diferentes categorías. De nuevo, mediante la aproximación del uso de robots, se pretende potenciar el aprendizaje por medio del montaje y configuración de objetos físicos tangibles, y potenciar la creatividad ya que se permite realizar prácticamente cualquier tipo de actividad con ellos. En la actualidad,

la principal limitación de estos sistemas es todavía su elevado coste, la escasa interactividad de dichos sistemas en los que los robots tienen un número predefinido de tipos de movimiento, y la dificultad de programarlos si no se poseen conocimientos avanzados.

- ***Superficies interactivas.*** Se trata de una tecnología consistente en una mesa con una superficie multi-táctil que a la vez actúa como pantalla, y de forma opcional soporta artefactos físicos tangibles que enriquecen la interacción. Esta tecnología fue concebida originalmente para soportar colaboración entre los usuarios presentes en torno a la mesa por medio de interacción natural con las manos y otros artilugios tangibles. En el ámbito de la educación el uso de esta tecnología ha estado orientada principalmente al consumo de juegos interactivos reproduciendo en muchos casos juegos de mesa no tecnológicos basados en cartas, muñecos, puzles, etc. En este sentido en [Mag03] y [Mag04] se explora esta tecnología para su uso en juegos, y de forma similar en [Maz07] se explora el desarrollo de un juego de rol utilizando como base la plataforma presentada en [Maz06] que pretende ser lo suficiente generalista para soportar el desarrollo de un amplio rango de actividades. En el ámbito del aprendizaje, en [Pip06] se utiliza esta tecnología para ayudar a adolescentes en terapias de grupo sociales especialmente orientada a individuos que padecen el síndrome de Asperger mientras que en [Slu04] se busca el desarrollar las habilidades de lectura en los jóvenes. En [Sca02] y [Kha07] se presentan algunas actividades educativas específicas en el contexto de los primeros cursos escolares de educación infantil. Algunas experiencias de uso en un museo se reportan en [Hor08], y en [Gab08] se explora el uso de un juego de cartas en un centro de mayores incorporando la mediación mediante agentes inteligentes para motivar a los jugadores.

2.4 Juegos sobre Superficies Interactivas para el Aprendizaje Creativo

De acuerdo a lo estudiado sobre teorías de aprendizaje se podría decir que en la actualidad las vertientes constructivistas y socioculturales tienen un importante peso ya que actualmente la práctica como medio para internalizar el conocimiento, y la interacción entre sujetos impactan de forma relevante en la instrucción. Es más, como hemos estudiado anteriormente, según la teoría del *flow* las actividades diseñadas para fomentar el aprendizaje han de encontrar un adecuado equilibrio entre las capacidades de los sujetos y los retos planteados. En nuestra opinión dichas capacidades pueden evolucionar de forma constructiva mediante un proceso en el que se faciliten los procesos de reflexión y creación cooperativa en la

que múltiples sujetos construyan y discutan de forma colectiva acerca de sus propios modelos internos que son creados y contrastados de forma social.

El objetivo que perseguimos es el de la facilitación del aprendizaje creativo, constructivo y social mediante el uso de las tecnologías de la información dando soporte a los siguientes requisitos fundamentales:

- *Proveer de un espacio físico común para la comunicación directa, acción, discusión, y reflexión*. El proveer de un espacio físico común tal y como se introdujo en el punto 2.2 parece un requisito esencial ya que se podría abordar tareas de discusión y reflexión manteniendo una comunicación directa entre los participantes.

- *Introducir elementos tangibles para la interacción*. Además de las necesidades de interaccionar de forma natural y más física, como siempre persigue el área de interacción hombre-máquina, la introducción de elementos tangibles que permitan realizar acciones y manipular el entorno de aprendizaje es una forma de aproximarse a la idea de *mediación* introducida por Vygotsky.

- *Potenciar las habilidades sociales para conducir el aprendizaje y soportar mecanismos de refuerzo (recompensa-castigo).* Aprovechando el requisito de disponer un espacio físico común, las teorías de corte social promueven el desarrollo de habilidades sociales en un grupo o comunidad, como siempre lo han hecho las actividades lúdicas de aprendizaje tradicionales en grupo. En particular, se debería soportar la participación, la cooperación, la colaboración, la competición y la comunicación. Además, la existencia tanto de refuerzos positivos como negativos ha sido siempre de mucha importancia como mecanismo para reconducir la conducta. El hecho de interaccionar en una comunidad puede potenciar el aprendizaje por observación reforzando o rechazando de manera natural determinados comportamientos y decisiones tomadas "in situ".

- *Situar el aprendizaje en un "ecosistema"*. La mayor parte de las teorías de aprendizaje presentadas anteriormente defienden el aprendizaje por medio de la acción, de la interacción con artefactos, objetos, entidades y/o el entorno. Por tanto debemos ir al concepto de ecosistema. En particular, ecosistemas en la que residen entidades interactivas, basadas en un modelo simple de relaciones acción-reacción y causa-efecto, que permitan exhibir comportamiento, y que interaccionan entre ellas y a las acciones de los participantes. Los términos simulación y escenario se encuentran fácilmente en el área de los juegos serios para el aprendizaje precisamente por el hecho que este tipo de aproximaciones reflejan determinados procesos de la vida real y cuotidiana sobre los que se quiere aprender de manera natural.

- *Soportar la creación y construcción de escenarios interactivos de forma colaborativa (responsabilidades), dando paso no sólo la fase de juego propiamente dicha sino que también la fase de creación y diseño del mismo*. No estamos hablando de desarrollar un juego que cumpla con los requisitos anteriores por medio del juego (*play*), sino que hay que ir un paso más allá proveyendo de una plataforma que soporte las fases de creación y diseño de un juego (*game*) como una fase enriquecedora para el aprendizaje y como forma para permitir cubrir de forma poco costosa el abordaje de aspectos curriculares que con juegos convencionales sería difícil cubrir como se discutió en los estudios introducidos en el punto 2.1 y 2.2. Esto supone sobre el concepto de ecosistema, la creación del ecosistema por parte de los participantes en el juego. Cada participante tendría la responsabilidad de crear y diseñar una parte del ecosistema, en general una entidad que debería comportarse de acuerdo a una serie de directrices básicas para lograr los objetivos establecidos por el tutor en el desarrollo global de la actividad de juego. El hecho que el tutor, y el resto de participantes estén presentes y colaborando permite guiarse durante el diseño, y permite la retroalimentación durante esta fase.

- *Jugar al juego diseñado*. Al final, tras tener el ecosistema diseñado de acuerdo a lo propuesto por los tutores, sería momento para jugar, interaccionar, experimentar en él. Observar las reacciones, reflexionar sobre ellas, e internalizar nuevas ideas de esta forma. El hecho de estar hablando de un ecosistema accionable y que reacciona, permite obtener una retroalimentación inmediata de la acciones durante el juego.

En definitiva, la investigación en los juegos serios en educación no debería enfocarse en producir juegos que cubran todos los posibles contenidos curriculares sino que deberían proporcionarse a los profesores y a los alumnos las herramientas para producir por ellos mismos juegos que sean de su provecho. Un primer paso hacia el soporte para estos entornos de aprendizaje del futuro es encontrar, en gran medida, los principios pedagógicos que sean adecuados, y de acuerdo a ellos proveer de una herramienta que a) permita a los profesores diseñar escenarios de aprendizaje básicos de acuerdo a los objetivos curriculares y las habilidades en las que están interesados; b) que los profesores y los alumnos tengan la capacidad de construir (crear) colaborativamente los mundos de juego que concreten los escenarios consistentes de entidades interactivas como espacios adecuados para la experimentación, exploración, reflexión, y discusión tanto en el proceso de creación como de juego.

Desde un punto de vista tecnológico y en base a las distintas propuestas tecnológicas analizadas con anterioridad, las superficies interactivas presentan a priori las características necesarias para dar soporte a los

requisitos anteriormente enumerados. Además, éstas no han de ser entendidas como meros elementos hardware a desarrollar sino como complejos sistemas de información que han de dar soporte dichos requisitos. Recordemos que en esencia se trata de una mesa que potencialmente permite la interacción multi-táctil y multiusuario sobre su superficie, así como la interacción mediante objetos físicos tangibles, tomando además la propia superficie como dispositivo de visualización (ver Figura 4). En torno a las posibilidades que esta plataforma ofrece se espera ser capaces de cubrir los requisitos identificados.

Figura 4. Ejemplo de superficie interactiva

Los elementos tecnológicos necesarios para el desarrollo de dichas infraestructuras educativas son los que describimos a continuación:

La infraestructura hardware que permita la interacción multi-táctil sobre una superficie horizontal.

Este elemento comprende dos partes claramente diferenciadas. Por una parte se encuentra el soporte físico (hardware) que implica la superficie interactiva así como todos los periféricos y dispositivos que la componen, y por otra parte el software básico que interpreta la interacción en la superficie de forma que permita "sentir" lo que ocurra sobre la misma.

Sobre la infraestructura hardware para superficies interactivas hay diversas aproximaciones, como por ejemplo las basadas en sensores capacitivos y resistivos, o las basadas en sistemas de visión teniendo en cuenta propiedades de refracción de la luz en determinadas materiales, como es el caso del principio de Iluminación Difusa (DI por *Diffuse Illumination*), o el principio de Reflexión Interna Total Frustrada (FTIR por *Frustrated Total Internal Reflection*). En nuestro trabajo estamos interesados en utilizar una infraestructura hardware basada en FTIR ya

que es una aproximación efectiva de bajo coste y permite el uso de retroproyección para visualizar las imágenes sobre la superficie [Han05].

El principio de FTIR aplicado a superficies interactivas funciona como sigue. La superficie interactiva se compone de varias capas: a) capa "sensora" (*sensing*); b) capa de confort y amoldamiento (*complaint*); c) capa de proyección (*projection*).

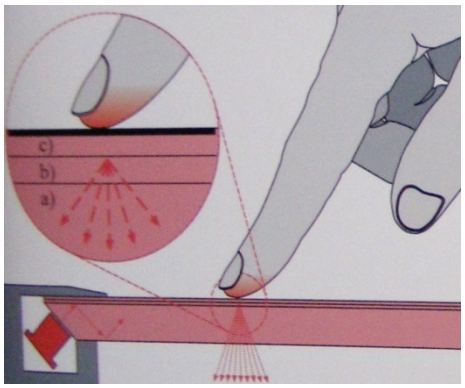


Figura 5. Principio FTIR

La capa *sensora* consiste de una lámina de policarbonato o acrílico de unos 10 mm. sobre la que se inyecta luz infrarroja emitida por una serie de LEDs, de tal forma que se produce el efector TIR. Cuando la superficie es perturbada, por ejemplo al interaccionar con los dedos, el efecto TIR se ve interrumpido y entonces en esos puntos de frustración de la reflexión interna la luz sale refractada hacia fuera del material de la capa *sensora* lo cual es aprovechado por el sistema de visión para analizar la forma de dichas perturbaciones (ver Figura 5). La capa de amoldamiento se requiere por cuestiones de ergonomía durante la interacción así como para mejorar los resultados del efecto FTIR y se ubica encima de la capa *sensora*. Típicamente consiste de un recubrimiento de silicona. Por último, la capa de proyección consiste de una fina lámina (*pvc* rígido o papel de trazado) que actúa como difusor para la luz proyectada proveniente del proyector (ver Figura 6).

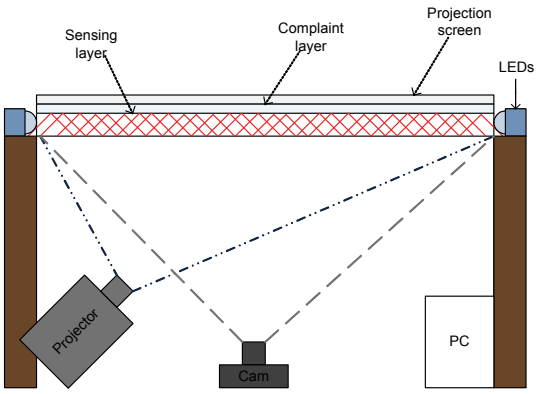


Figura 6. Esquema superficie interactiva

En cuanto el software, es necesario el uso de sistemas de visión por computador que extraigan las interacciones que se produzcan a partir de las imágenes capturadas por la cámara desde debajo de la superficie, y analizar dicha información para extraer acciones gestuales que se hayan realizado.

Un motor de medios (gráficos y sonido) para aumentar la mesa interactiva.

Será necesario el disponer de un motor de medios y de física, dotado en especial de ciertas capacidades 2D que permita sobre él el desarrollo rápido y eficaz de animaciones, representaciones visuales y/o sonoras, interacciones entre representaciones visuales de objetos, etc. Sobre este elemento recae la responsabilidad en última instancia de aumentar la superficie interactiva tanto visual como sonoramente.

La infraestructura software que permita el seguimiento de objetos tangibles sobre la superficie, y permita la interacción a través de los mismos.

Dado que queremos interactuar con objetos físicos, además de con nuestras manos y dedos, será necesario utilizar algún mecanismo que lo permita. Cabe la posibilidad de utilizar diversas tecnologías como pueden ser cualquiera que utilice dispositivos electro-mecánicos dotados de alguna interfaz de comunicaciones (e.g. basados principalmente en RFID, en Bluetooth o incluso en WiFi), o bien, dado que estamos ante una habilidad sensitiva basada en algoritmos de *visión* se puede utilizar el seguimiento de determinados patrones sobre la superficie interactiva. En nuestro caso estamos interesados en utilizar la tecnología de seguimiento basada en marcadores que tradicionalmente se ha utilizado en sistemas de Realidad Aumentada. Esta aproximación resultará extremadamente barata ya que sólo requerirá la impresión en papel de los marcadores y su adhesión a la superficie inferior de los objetos tangibles.

La infraestructura software que soporte la creación y animación de ecosistemas reactivos.

Dado que queremos centrarnos en el desarrollo y en el juego sobre ecosistemas reactivos necesitaremos una infraestructura software que soporte las entidades del ecosistema, así como un motor de simulación que haga evolucionarlo de acuerdo a las acciones de los usuarios, las reacciones de las entidades, y se comunique con el motor de medios para llevar a cabo la representación del mismo.

Herramientas de autoría amigables y usables que utilicen las diferentes infraestructuras, toolkits y motores para el diseño de los juegos sobre la plataforma.

Dado que queremos soportar la creación del juego como una fase más del propio juego, será necesario proveer de herramientas adecuadas para la creación, teniendo en cuenta todos los motores implicados en la infraestructura (simulador, medios, etc.). El proveer de herramientas de autoría es una tarea en general difícil ya que se tiene que tomar en consideración todos los factores que queremos que puedan ocurrir en la "creación" resultante. Además, en nuestra propuesta se plantea una infinidad de cuestiones concernientes a la usabilidad, e interacción hombre-máquina ya que estamos planteando que la creación debería ser soportada en una plataforma poco explorada como son las superficies interactivas, y por tanto las guías de diseño convencionales para la interacción no son en general de aplicación directa.

■

En esta tesis de máster se han abordado prácticamente dos de los anteriores elementos. En primer lugar, en el capítulo 3 se presentará el desarrollo de una infraestructura para el seguimiento de objetos tangibles. Ésta está basada en el concepto de marcadores de realidad aumentada que nos permitirá añadir interacción mediante determinados objetos físicos que lleven incrustados los correspondientes marcadores. En segundo lugar, el capítulo 4 presenta la infraestructura software que dará soporte a la creación y animación de ecosistemas reactivos. En particular se trata de un subsistema de gestión de eventos semántico que descansa sobre OWL para describir los eventos y gestionar la resolución de las suscripciones con los eventos sobre el cual trabajará el motor de simulación.

2.5 Conclusiones

En este capítulo se han presentado los conceptos fundamentales del entorno de aprendizaje al que se pretende dar soporte en un futuro. Así, se ha introducido a la noción de *juego* y *juego serio*, tanto en el ámbito tradicional como en el tecnológico. También se introdujo la posibilidad de utilizar juegos comerciales en el ámbito de la educación así como las estrategias y tareas que se consideran, resaltando la importancia de la reflexión, de la actividad y de la discusión en el aprendizaje. Tareas, que como se mostró en la exposición sobre teorías de aprendizaje, forman parte de gran parte de las teorías de aprendizaje de corte constructivista aunque materializadas de formas diversas.

También se ha hecho un recorrido por los diferentes soportes tecnológicos existentes para la educación, y las características básicas de cada uno de ellos en cuanto a su efectividad en el aprendizaje.

Finalmente se han detectado los requisitos para el entorno de aprendizaje que pretendemos que son los siguientes:

- Proveer de un espacio físico común para la comunicación directa, acción, discusión, y reflexión.

- Introducir elementos tangibles para la interacción.

- Potenciar las habilidades sociales para conducir el aprendizaje y soportar mecanismos de refuerzo (recompensa-castigo).

- Situar el aprendizaje en un "ecosistema".

- Soportar la creación y construcción de escenarios interactivos de forma colaborativa (responsabilidades), dando paso no sólo la fase de juego propiamente dicha sino que también la fase de creación y diseño del mismo.

- Jugar al juego diseñado.

Y se han descrito los principales elementos tecnológicos que necesitamos para cubrir tales requisitos, girando estos en torno al concepto de *Superficie Interactiva* como plataforma que mejor facilite la consecución de los requisitos:

- La infraestructura hardware que permita la interacción multi-táctil sobre una superficie horizontal.

- Un motor de medios (gráficos y sonido) para aumentar la mesa interactiva.

- La infraestructura software que permita el seguimiento de objetos tangibles sobre la superficie, y permita la interacción a través de los mismos.

- La infraestructura software que soporte la creación y animación de ecosistemas reactivos.

- Herramientas de autoría amigables y usables que utilicen las diferentes infraestructuras, toolkits y motores para el diseño de los juegos sobre la plataforma.

Por tanto, la propuesta general que se presenta en este capítulo trata de aunar las características clave de cada una de las teorías de aprendizaje

persiguiendo el desarrollo de un entorno de aprendizaje creativo eficaz y motivador que mejore el proceso de aprendizaje.

CAPÍTULO III

UN SISTEMA DE REALIDAD AUMENTADA BASADO EN MARCADORES

En este capítulo se describe la infraestructura que nos permitirá abordar la tarea de seguimiento e interacción mediante objetos tangibles. Esta tarea se ha enfocado al desarrollo de un sistema de visión por computador de Realidad Aumentada basado en la detección de marcadores (*fiducial markers*). El objetivo es triple. En primer lugar se busca cubrir el seguimiento de forma general de objetos tangibles en las superficies interactivas que será la plataforma final. En segundo lugar, ahondar en el conocimiento de este tipo de sistemas de seguimiento, implementando uno con características relativamente sencillas, que nos permita su rápida adaptación ante posibles cambios futuros en las características de los marcadores debido a posibles nuevos escenarios y requisitos que surgieran en el momento de la integración de todas las tecnologías implicadas en nuestro concepto de *superficie interactiva*. Y por último, dado que el seguimiento es un problema habitual abordado en sistemas móviles y de realidad aumentada, hemos dado un enfoque generalista a esta infraestructura de forma que pueda ser utilizada fácilmente en aplicaciones de realidad aumentada tanto en dispositivos móviles como en los utilizados en los basados en gafas de visión artificial. De hecho, a efectos de pruebas y de evaluación del desarrollo se ha integrado en sistemas de realidad aumentada de los dos tipos con éxito.

El capítulo se estructura de la siguiente forma. Dado que se ha dado un enfoque general se introducirá en primer lugar el concepto de realidad aumentada basado en marcadores y las diferentes etapas de procesamiento que este tipo de software realiza, y se introducirán algunos *toolkits* ya existentes. A continuación se describirá el funcionamiento del subsistema que hemos desarrollado con bastante nivel de detalle. Finalmente se comentarán detalles de implementación y se expondrán las conclusiones.

3.1 Realidad Aumentada basada en Marcadores

La "Realidad Aumentada" (RA) consiste en añadir gráficos virtuales en tiempo real al campo de visión de una persona. Su finalidad es superponer en el entorno real la información que interesa visualizar. Se diferencia de la Realidad Virtual en que mientras ésta pretende reemplazar el mundo real, la RA lo complementa con información adicional. Véase la Figura 7.

A diferencia de la Realidad Virtual, que introduce al usuario en un ambiente informático artificial, la RA no lo aleja de la realidad, sino que la complementa, manteniéndole en contacto con ella, al mismo tiempo que se interactúa con objetos virtuales. En la práctica, la Realidad Aumentada es una interfaz alternativa a la pantalla del ordenador que se aplica en diversos campos como la medicina, ocio, mantenimiento de maquinaria, arquitectura, robótica, industria, etc.

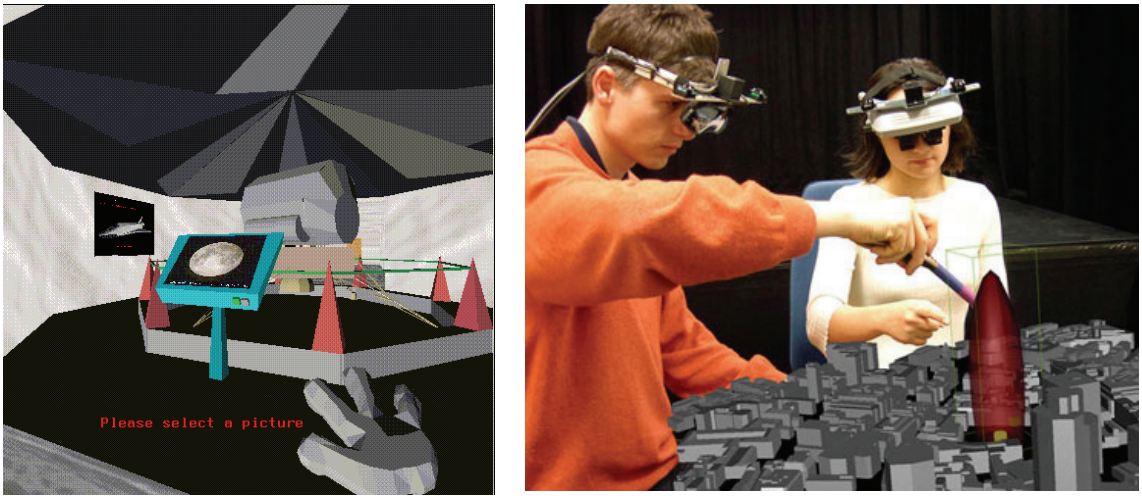
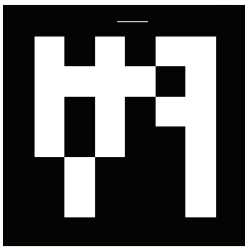


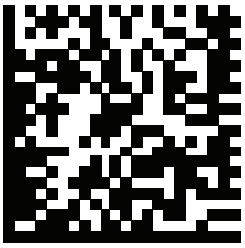
Figura 7. Realidad Virtual vs. Realidad Aumentada.

Cuando hablamos de RA basada en marcadores nos referimos a que se utiliza algún mecanismo basado en lo que se conoce como marcadores físicos (*fiducial markers*) para realizar algún tipo de seguimiento de objetos. Un sistema de marcadores consiste en una serie de patrones que son introducidos en el entorno físico y automáticamente detectados mediante un sistema de procesamiento de imágenes digitales. El tipo de marcador que se utilice determina las características del sistema de detección. No obstante, la mayoría de los que se han desarrollado en la comunidad se tratan de marcadores 2D cuadrados que constan de un borde diferenciado y de una imagen identificadora en su interior. Los siguientes ejemplos se corresponden a los tipos de marcadores más relevantes usados por diversos sistemas existentes:

Marcadores plantilla. Este tipo de marcadores se caracterizan por tener un amplio borde que da contraste facilitando su detección. En su interior figura una imagen que actúa como imagen identificadora. Este tipo es el usado por el software ARToolkit [Artw3].



Marcadores ID. Estos se componen de un borde cuadrado y en su interior se codifica una palabra binaria identificadora. Las codificaciones pueden ser sencillas como un simple identificador o un código BCH, o por el contrario puede tratarse de un código mucho más elaborado diseñado mediante técnicas de detección y corrección de errores [Rek00]. Ese es el caso de nuestro subsistema de marcadores.



Marcadores DataMatrix. DataMatrix es un sistema industrial de codificación bidimensional que permite la generación de un gran volumen de información en un formato muy reducido, con una alta fiabilidad de lectura gracias a sus sistemas de información redundante y corrección de errores. Puede considerarse como un caso particular de los marcadores basados en identificadores digitales salvo que no tienen un marco totalmente cerrado. Corresponde al ISO/IEC16022.

Marcadores de marco digital. Este tipo de marcador se caracteriza por la codificación de forma discreta del identificador digital del marcador a lo largo de la parte interna del borde. Esto permite utilizar la región interna para la codificación de imágenes que sean inteligibles para las personas por razones de usabilidad y a la vez se descansa en técnicas digitales de codificación en el marco.

Marcadores de marco partido. Este tipo sigue el mismo principio que los marcadores de marco digital, salvo que no consideran el uso de un borde completamente cerrado siendo especialmente adecuado para aplicaciones en las que los marcadores deben ser manipulados con las manos ya que permite su utilización a través de los lados sin borde sin interrumpir su detección.

Marcadores circulares. La forma de este tipo de marcadores es completamente diferente y consecuentemente su detección recae en técnicas bastante diferentes de las que se podrían utilizar en todos los marcadores anteriores.

Algunos sistemas que utilizan marcadores son los presentados en [isew3] y [Lop02].

Marcadores amorfos. En esta categoría cae cualquier tipo de marcador que no posee propiedades regulares aunque evidentemente sí que responden a principios matemáticos no triviales que permitan su adecuada detección e identificación. Este es el caso de los marcadores *Ameoba* utilizados en *reacTIVision* [Kal07] que se están utilizando ampliamente en aplicaciones sobre superficies interactivas.

3.1.1 Algunos sistemas de marcadores

Sin duda alguna uno de los sistemas de marcadores más influyentes y pionero ha sido *ARToolkit* [Artw3]. Su éxito fue posible gracias a la popularización y democratización de este tipo de tecnología al ponerse al alcance de la comunidad, así como también debido a su facilidad de uso que hizo proliferar cantidad de aplicaciones explorando su utilización. ARToolkit utiliza marcadores de plantilla con un borde negro llevando en su interior una de la imágenes que el sistema haya registrado previamente como identificador. Estos marcadores son ventajosos desde el punto de vista de la usabilidad y amigabilidad ya que los patrones tienen un significado para los usuarios, sin embargo su detección se basa completamente en técnicas de correlación de imágenes lo que hace que en ocasiones se produzcan errores de detección debido a la confusión entre las plantillas que los usuarios han registrado en el sistema.

Otro sistema de relevancia que ha influido en el desarrollo de otros sistemas de seguimiento es el conocido como *ARTag* [Fia04]. ARTag cambia totalmente la aproximación a la detección de los marcadores. Por una parte utiliza técnicas algo diferentes a ARToolkit para extraer los bordes, permitiendo incluso la detección de marcadores bajo ciertas condiciones de oclusión gracias a algoritmos de enlazado de aristas. Por otra parte, el marcador sigue teniendo un borde cuadrado cerrado pero en su interior no se contempla el uso de una imagen establecida arbitrariamente sino de un código digital específicamente diseñado para ser robusto y soportar la oclusión permitiendo incluso su correcta decodificación gracias a técnicas de corrección de errores en códigos digitales. Esto lo hace más adecuado ante escenarios más exigentes aunque el precio a pagar es la pérdida de amigabilidad en su uso ya que la identificación por parte de las personas ya no es posible.

Finalmente, *ARToolkitPlus* [Wag07] combina las ideas de ambos sistemas de seguimiento anteriores pero aplicados a dispositivos móviles en lugar de aplicaciones convencionales de Realidad Aumentada. Este sistema ha sido

reimplementado y evolucionado a lo que actualmente se conoce bajo el nombre de *Studierstube Tracker* [Stuw3], no restringiéndose a marcadores de plantilla y de ID, sino soportando una más amplia variedad de tipos de marcadores, como los de marco digital y los partidos.

Visto lo que es un marcador en nuestro contexto y algunos de los sistemas de seguimiento más relevantes se introducen a continuación las tareas que un sistema de marcadores debe abordar para cumplir su función desde un punto de vista general del reconocimiento de formas de forma que nos sirva como punto de partida en la explicación de nuestro sistema.

3.1.2 La perspectiva del Reconocimiento de Formas

La detección de marcadores es una tarea de visión artificial que puede verse desde un punto de vista clásico del reconocimiento de formas como se muestra en la siguiente figura.

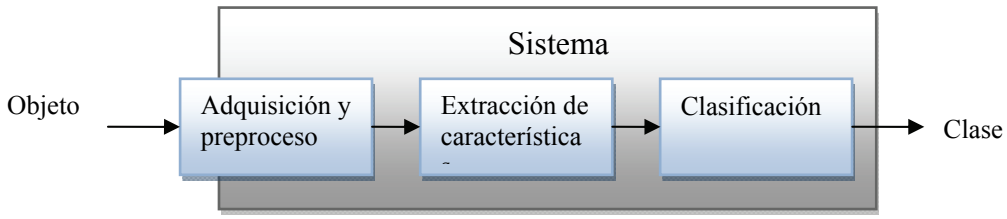


Figura 8. Visión artificial

La etapa de adquisición y preproceso consiste en tomar los datos *brutos* de entrada y someterlos a posibles tratamiento de mejora (eliminar ruido, compensación,…). Esta etapa realmente está en la frontera del sistema y dependiendo de la tarea a resolver forma parte del sistema o no. Lo que se obtiene tras el preproceso generalmente está en el mismo dominio que los datos de entrada.

La etapa de Extracción de Características consiste en obtener una representación adecuada de los datos de entrada. El objetivo es transformar los datos brutos en información, con un formato adecuado para que pueda ser tratado por la etapa de clasificación. En esa etapa de clasificación se obtiene como salida la información de alto nivel que indica la clase del objeto reconocido.

En la práctica todo esto es mucho más complejo ya que las etapas no están tan claramente diferenciadas: algunas veces por eficiencia, otras veces por tener subsistemas de reconocimiento para formar el sistema principal. Todo depende de la tarea que se esté considerando, pero la idea básica es tal como se ha comentado.

En el caso de sistemas de visión, esas tres etapas en general se enlazan con los siguientes temas:

- Mejora de la imagen, segmentación, filtrado, umbralización, binarización, etiquetado…
- Representación y descripción: contornos, vectores de características…
- Reconocimiento de objetos: cualquier proceso de inteligencia artificial que dé solución final a partir de la descripción del problema obtenida (HMM, NN, Bayes,…).

De una manera aplicada, el sistema de visión de Realidad Aumentada desde el punto de vista del área de reconocimiento de formas, podría esquematizarse como presenta la Figura 9.

Un dispositivo con cámara (imagen superior izquierda) captura una imagen y la muestra en pantalla como fondo. La imagen captada pasa al sistema de detección de marcadores (imagen superior central) y se le aplica un proceso de segmentación y detección de marcadores (imagen superior derecha). Cuando un marcador es detectado, el sistema calcula la posición y orientación de la cámara con respecto al sistema de coordenadas del marcador (imagen inferior izquierda). Con esta información es posible posicionar objetos 3D de manera precisa encima del marcador. La imagen final compuesta por la imagen capturada de fondo y el objeto 3D superpuesto se muestra en la pantalla del dispositivo (imagen inferior derecha).

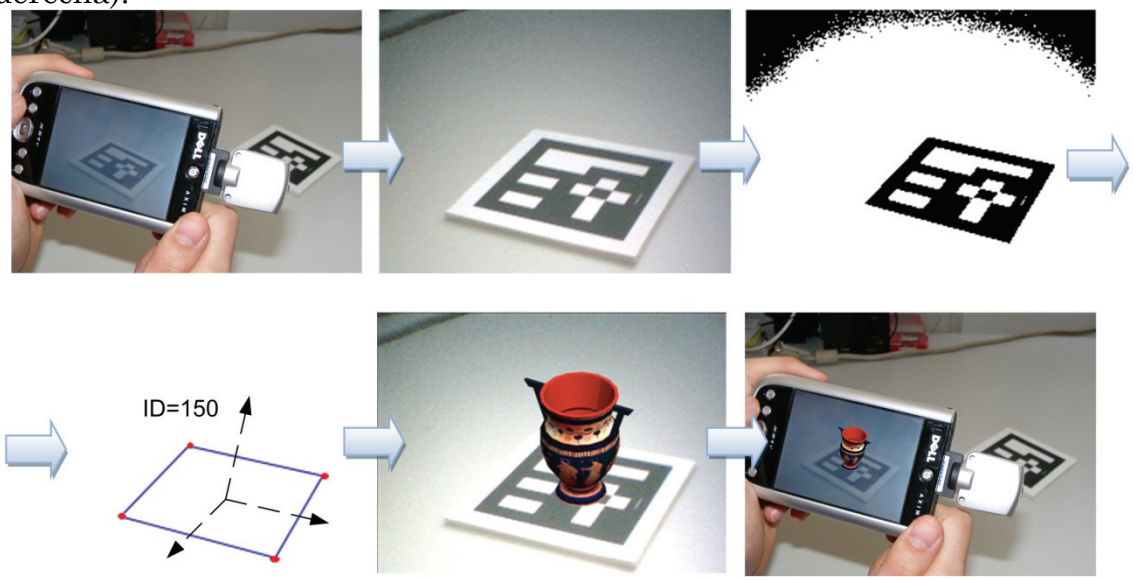


Figura 9. Etapas típicas de una aplicación de RA basada en marcadores

3.2 Funcionamiento del subsistema de seguimiento

La presente sección tiene como objetivo explicar cuáles son las distintas fases en las que se divide el subsistema de Realidad Aumentada basado en marcadores. Inicialmente se expondrá una visión global del funcionamiento del sistema apoyado en un diagrama de flujo global. En los siguientes puntos se ampliará la información de las etapas explicadas en aquí y se abordarán de una manera más técnica y específica de acuerdo a nuestra implementación. Hacia el final se abordan conceptos generales de geometría de visión monocular para poder explicar finalmente la estimación de pose y orientación y la calibración de la cámara, que se trata de una tarea indispensable para que el sistema funcione, teniendo la base matemática necesaria.

El subsistema puede agruparse en tres fases diferenciadas, la *Detección*, la *Obtención de los Identificadores,* y la *Estimación de Pose y Orientación*. El funcionamiento general del sistema consiste en capturar imágenes mediante una cámara para ser pasadas posteriormente por varias fases. El objetivo es el de detectar la presencia de marcadores en las imágenes capturadas. En caso negativo la aplicación mostraría la imagen capturada en pantalla sin ningún objeto sintético superpuesto derivado de la detección e identificación de algún marcador. En caso positivo la imagen capturada se mostraría en pantalla y además se superpondría información proveniente del sistema de RA.

Cada imagen capturada entra en la fase de detección donde, en primer lugar se lleva a cabo la segmentación. En esta etapa de la detección se prepara la imagen mediante umbralización y etiquetado para averiguar si existen objetos en ella. Si efectivamente hay objetos que analizar, estos objetos entrarán en la etapa de detección de marcadores candidatos.

El objetivo de la detección de marcadores candidatos es decidir si los objetos presentes en la imagen son marcadores o no. Para ello se extraen características como el contorno y se evalúan ciertas características como el tamaño o la posición. También se analiza la forma del objeto para averiguar si es cuadrado. La detección de marcadores candidatos se explica más a fondo en la sección 3.2.1.2

Una vez finalizada la fase de Detección es posible saber cuántos marcadores definitivos hay presentes en la imagen, la posición de sus vértices y el centro del marcador. Con estos datos es posible obtener el identificador del interior del marcador.

La última fase consiste es la estimación de la pose y la orientación del marcador. Esta fase consiste en reconstruir la posición de la cámara a raíz

de la imagen del marcador detectado y es imprescindible para poder mostrar las imágenes superpuestas de manera coherente con la posición de la cámara y la imagen tomada.

El diagrama de flujo global de la Figura 10 sirve de apoyo para una mejor comprensión de las etapas introducidas anteriormente. En los puntos sucesivos de este capítulo se explicará en detalle el funcionamiento de las fases y etapas.

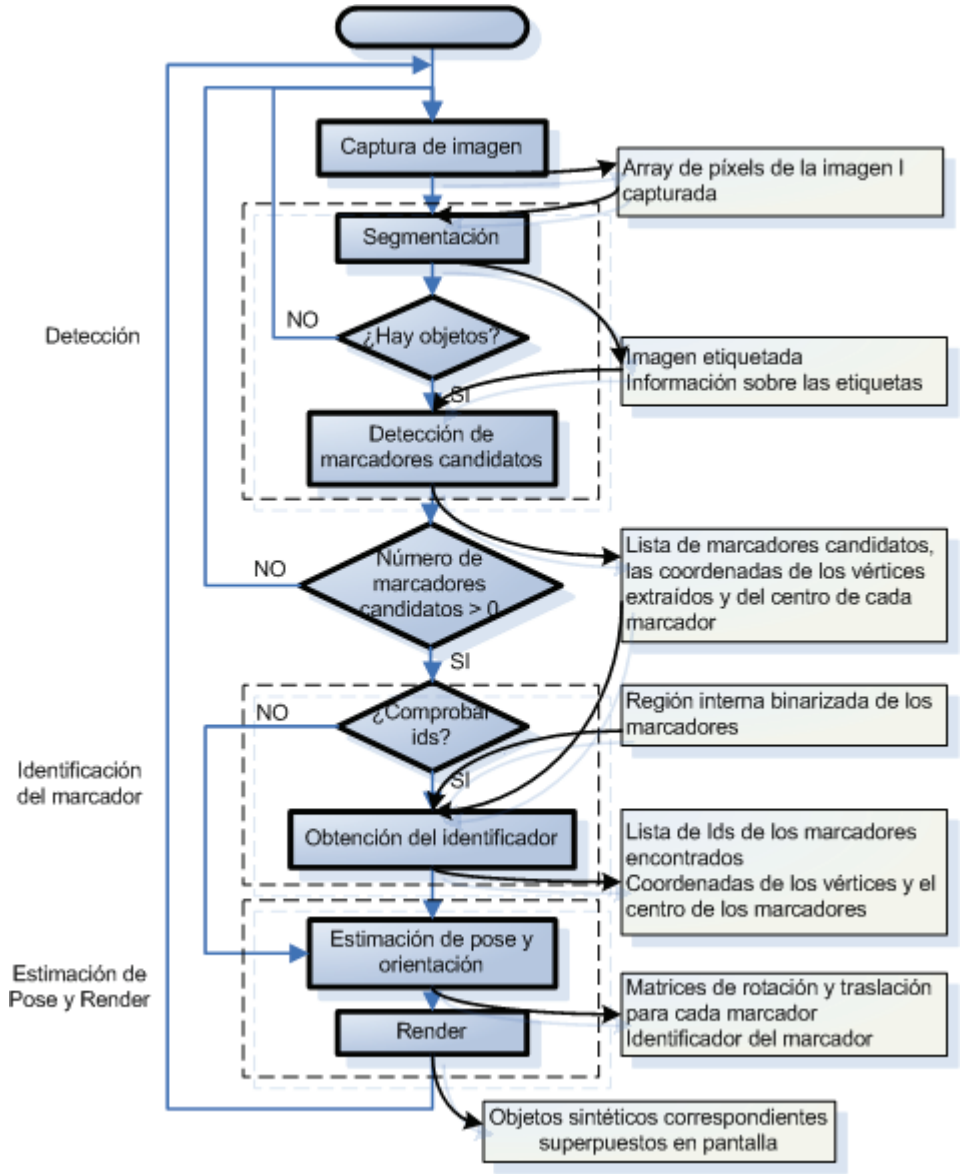


Figura 10. Diagrama de flujo global de AR con marcadores

3.2.1 Detección

Una vez obtenida la imagen a analizar, comienza el procesamiento de la misma con el objetivo final de detectar la presencia de marcadores en ella. Esta fase del reconocimiento de marcadores es en la que activamente se

analiza píxel a píxel la imagen de entrada obteniendo finalmente los marcadores contenidos en ella y sus identificadores. Esta fase a su vez está dividida en varias etapas que se explicarán a continuación.

3.2.1.1 Segmentación

La segmentación consiste en utilizar técnicas que sean capaces de diferenciar en una imagen lo que es "fondo" de lo que son "objetos". Cómo se haga la segmentación depende de la tarea de visión por ordenador que se vaya a realizar.

Habitualmente la segmentación se lleva a cabo mediante dos fases. Una primera fase de umbralización, utilizándose la más conveniente para cada caso, donde se separan los objetos del fondo y una segunda fase de etiquetado de componentes conexas atendiendo a la adyacencia de los píxeles para diferenciar dos objetos distintos.

3.2.1.1.1 Umbralización

La umbralización o *thresholding* va muy ligada a la binarización de imágenes y es el proceso por el cual una imagen pasa a estar en escala de grises o a ser una imagen binaria atendiendo a una serie de valores umbrales.

Existen distintas aproximaciones a la umbralización. Unas basadas en umbrales fijos y otras más sofisticadas se basan en la información que se deriva del histograma (umbrales adaptativos).

Las imágenes obtenidas mediante el proceso de captura pueden ser de diferente naturaleza. Esto incluye imágenes en color RGB, color YUV, escala de grises, blanco y negro, etc. Éstas vienen definidas por cómo se almacena la información de cada píxel, cuántos bits se van a utilizar para almacenar esa información o qué información queremos reflejar (color, solo luminosidad,…). En las sucesivas etapas, para poder llevar a cabo el análisis necesario para nuestro objetivo es necesario que las imágenes estén en blanco y negro. En el caso de que la imagen ya esté en ese formato este paso será omitido y la imagen pasará directamente a la siguiente etapa del análisis. En cambio, si no es así, será necesario hacer la transformación pertinente. A este proceso se llama binarización: a cada píxel se le asignará el valor 0 representando el fondo, o 1 representando un objeto según el valor real del píxel y un valor umbral, un ejemplo puede verse en la siguiente figura.

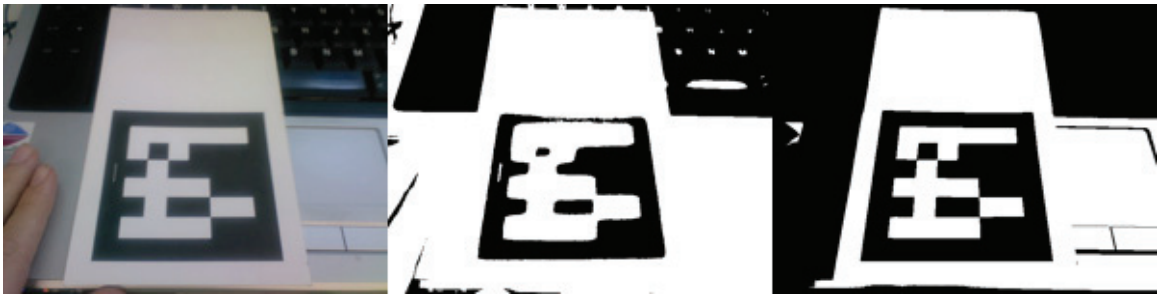


Figura 11. Umbralización: imagen original, con valor umbral = 70, con valor umbral = 140.

Una manera de realizar esta transformación es fijando un valor umbral con el que comparar el valor de cada píxel de la imagen fuente. Si el valor del píxel analizado es mayor que este valor umbral, esté píxel será considerado como objeto y tendrá valor 1. Si en cambio el valor es menor o igual, el valor de ese píxel en la imagen resultante será cero y se considerará un píxel del fondo. Analíticamente:

Se escoge un umbral inicial (T) de manera aleatoria o según otro método.

La imagen es segmentada creando dos conjuntos:

$G1 = \{ I(x, y) : I(x, y) > T \}$ (píxeles objeto)

$G2 = \{ I(x, y) : I(x, y) \leq T \}$ (píxeles fondo)

Donde I(x, y) es el valor del píxel en la columna x fila y.

Este proceso de umbralización tiene ventajas e inconvenientes. Por un lado es un algoritmo sencillo, fácil de implementar y sin mucho coste computacional, pero por otro lado es muy dependiente de las condiciones de iluminación en las que las imágenes son tomadas. Por ejemplo, si una imagen es tomada con luz tenue, un valor umbral de 180 sería demasiado alto y se perdería mucha información al hacer la transformación. En cambio para unas condiciones donde la iluminación fuera mucho más intensa este umbral sería muy bajo perdiéndose también una importante cantidad necesaria de información.

El algoritmo utilizado para esta fase se explica en el Listado 1. Hay que destacar que por motivos de eficiencia, la umbralización de la imagen se efectúa a la vez que el etiquetado aunque en este documento vamos a presentar los algoritmos por separado para mayor claridad.

Listado 1. Umbralización

```
Entradas:
Imagen de tamaño HxW: I
Valor umbral: threshold
Valor que se le asignará a los "objetos" = objectBrightness
Valor que se le asignará al "fondo" = backgroundBrightness
```

```
Salidas:
Imagen umbralizada de tamaño HxW: thresholdedImage

Inicio

Para cada píxel j en la fila 0 o W
      thresholdedImage[0,j] = backgroundBrightness
      thresholdedImage[W,j] = backgroundBrightness
FinPara

Para cada píxel i en la columna 0 o H
      thresholdedImage[i,0] = backgroundBrightness
      thresholdedImage[i,H] = backgroundBrightness
FinPara

//Establece los bordes de la imagen como fondo


Para cada píxel Idx de la imagen recorriendo de izquierda a derecha y
de arriba a abajo.

      Si I[Idx] < threshold
            isObject = true
            thresholdedImage[idx] = objectBrightness
      Sino
            isObject = false
            thresholdedImage[idx] = backgroundBrightness
      FinSi

FinPara

Fin
```

Típicamente en una canal de 8 bits la ausencia de color (negro) se corresponde con el valor 0. Mientras que una iluminación total (color blanco) toma el valor 255.

Como ya se ha remarcado anteriormente, el parámetro clave para este proceso es la elección del valor umbral. Un método más sofisticado que la elección de un valor umbral fijo es el de calcular el umbral para cada imagen procesada según los propios valores de esa imagen en concreto. Los métodos que usan umbralización adaptativa se basan generalmente en la creación de un histograma que recoja la frecuencia de aparición de las intensidades de los píxeles de la imagen. Una vez construido este histograma se realiza algún tipo de operación que en cierta medida obtendrá un valor umbral "óptimo" para esa imagen. Este proceso es más costoso que el anterior pero en condiciones de luz variables obtiene unos resultados visiblemente mejores. Es por ello que si la capacidad de procesamiento de la plataforma en la que se trabaja es suficiente, este tipo de umbralización sea el predeterminado. En particular, se implementaron algunas estrategias sencillas que no sobrecargaran el proceso de binarización pero que permitieron introducir cierta adaptación a las condiciones de iluminación.

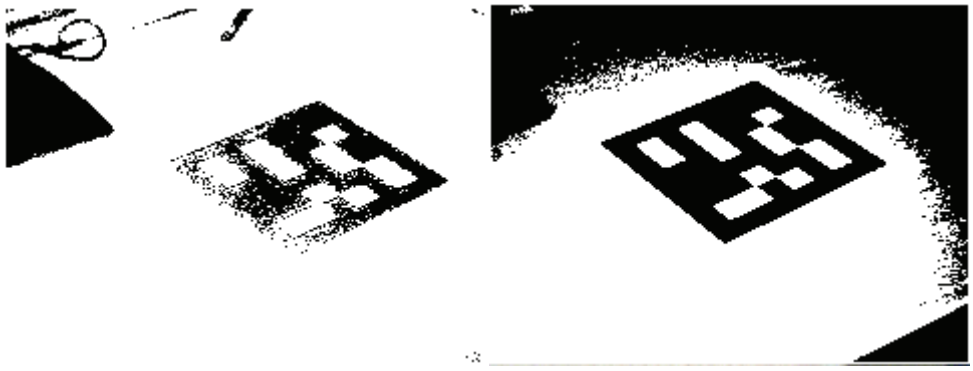


Figura 12. Resultados umbralización adaptativa

En particular, se introdujo una aproximación adaptativa para tratar de paliar en la medida de lo posible las condiciones variables de iluminación teniendo en cuenta que no se pretende invertir excesivo tiempo de cómputo en este proceso. Para ello se ha tomado siempre el histograma construido durante el análisis de la imagen capturada anterior partiendo de la hipótesis que imágenes sucesivas resultarán en histogramas en general muy parecidos. El umbral adaptativo se establece a cada imagen de acuerdo a la siguiente expresión:

$$T_{Adap} = \frac{\underset{i}{\text{argmax}}\{f_i\} + \underset{j}{arg\max}((\text{j} - \underset{i}{\text{argmax}}\{f_i\})^2 * f_j)}{2}$$

En esencia lo que persigue es obtener los dos picos máximos en el histograma de tal forma que el segundo esté lo más separado posible del primero. Esto se hace así para evitar que los dos picos sean por ejemplo dos máximos consecutivos o muy próximos que hagan que no haya apenas valores de grises entre ellos, y que consecuentemente haga que la imagen prácticamente pierda toda la información. Se probaron algunas variantes de la expresión anterior en la que se introducía un parámetro k que forzaba la distancia mínima entre picos aunque no se obtuvieron grandes diferencias en el resultado. La Figura 12 muestra los resultados de umbralizar una imagen capturada utilizando un umbral fijo (izquierda) y el resultado de utilizar la selección del umbral adaptativo. Claramente, hay situaciones como el de la figura en las que el uso de un umbral fijo no es recomendable ya que puede haber regiones que estén más iluminadas que otras produciendo efectos indeseables y pérdida de información relevante al umbralizar. Sin embargo, el uso de un algoritmo de selección del umbral adaptativo puede resolver en cierta medida esos problemas.

3.2.1.1.2 Etiquetado

La segunda fase de la segmentación es el etiquetado (*Labeling*). El etiquetado consiste en obtener la información de "objetos" en la imagen

atendiendo a su nivel de gris. En esencia, consiste en obtener las componentes conexas de píxeles de forma que cada píxel de cada componente tiene el mismo nivel de gris y representa por tanto un objeto en la imagen, distinguiendo de esta manera diferentes objetos en una imagen.

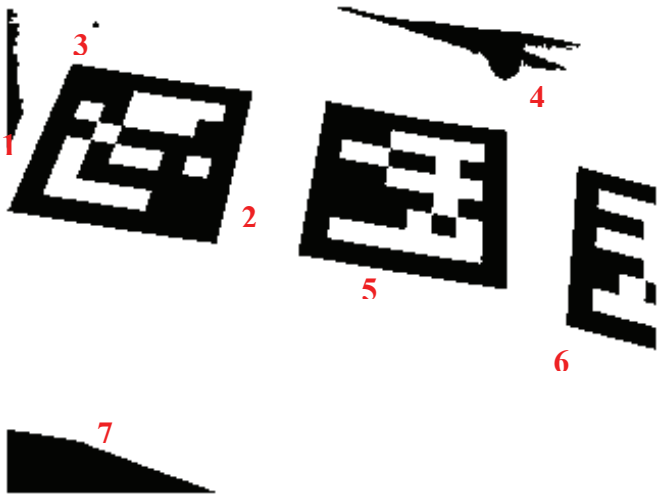


Figura 13. Etiquetado de una imagen.

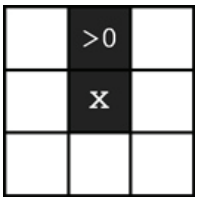
En la Figura 13, por ejemplo, se ha umbralizado y binarizado la imagen. A continuación se ha realizado el etiquetado, y como salida hemos obtenido el conjunto de componentes conexas. Sobre estas componentes ya se podría extraer las características que se requieran, como por ejemplo los contornos para decidir si se corresponde a figuras que queremos detectar.

Para construir el algoritmo de etiquetado de imágenes necesitábamos de alguna estructura de datos que nos permitiera representar fácil y eficientemente componentes conexas. Este es el caso de los MF-SETs con comprensión de caminos y fusión por rango que finalmente decidimos utilizar. Un MFSET es una estructura de datos utilizada para mantener una colección de subconjuntos disjuntos. Esta estructura soporta básicamente dos operaciones de forma muy eficiente: la operación *merge* y la operación *find*. La operación *merge* une dos subconjuntos disjuntos en único subconjunto, y la operación *find* determina a qué subconjunto disjunto pertenece un elemento del conjunto. El motivo de mantener la información de conjuntos de píxeles mediante un MFSET es que, dado que es necesario recorrer todos los píxeles de la imagen, es necesario utilizar operaciones rápidas para que el tiempo de etiquetado de la imagen sea el mínimo posible.

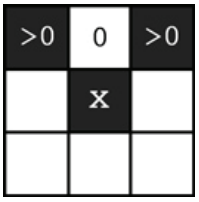
El algoritmo de etiquetado se muestra en pseudocódigo mostrando todos los casos ayudados por una figura gráfica para su mejor comprensión. Recordaremos que se hace uso de las operaciones de un MFSET *find* y *merge* definidas en el párrafo anterior además de la operación *add* que crea un nuevo subconjunto disjunto, una nueva etiqueta.

Listado 2. Etiquetado

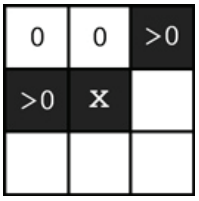
```
Entradas:
Imagen umbralizada de tamaño HxW: thresholdedImage

Salidas:
Imagen etiquetada de tamaño HxW: labeledImage
Número de subconjuntos disjuntos, número de objetos: labelCount
Información característica de los conjuntos: infoLabels

  Inicio

  Para cada píxel de la imagen recorriéndola de izquierda a derecha y de
  arriba a abajo donde i son las filas y j las columnas.

  //Inicialmente el valor todos los píxeles de labeledImage es 0.

  Si     (thresholdedImage[i,j] == objectBrightness) //es objeto
         Si     (lab = labeledImage[i - 1, j]) > 0)
```



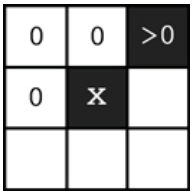

```
                CASO 1
                // asignar [i-1, j]
                labeledImage[i, j] = lab = find(lab);

         Sino
                Si     (lab2 = labeledImage[i - 1, j + 1]) > 0)
                       Si     ((lab3 = labeledImage[i - 1, j - 1]) > 0)
```



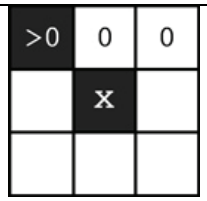

```
                              //CASO 2
                              // asignar merge [i-1, j-1] [i-1,j+1]
                              labeledImage[i, j] = lab = merge(lab2 =
                              find(lab2), lab3 = find(lab3));

                       Sino
                              Si     ((lab3 = labeledImage[i, j - 1]) > 0)
```



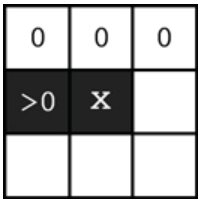

```
                                     // CASO 3
                                     // asignar merge [i, j-1][i-1, j+1]
                                     labeledImage[i, j] = lab = merge(lab2 =
                                     find(lab2), lab3 = find(lab3));

                              Sino
```



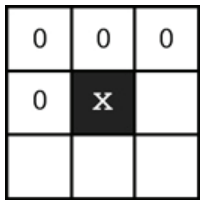

```
                                     // CASO 4
                                     // asignar [i-1, j+1]
                                     labeledImage[i, j] = lab2 = find(lab2);

                              FinSi
                       Fin
                       Si
                Sino
                       Si     ((lab = labeledImage[i - 1, j - 1]) > 0)
```




```
                    // CASO 5
                    // asignar [i-1, j-1]
                    labeledImage[i, j] = lab = find(lab);

              Sino
                    Si    ((lab = labeledImage[i, j - 1]) > 0)
```




```
                          // CASO 6
                          // asignar [i, j-1]
                          labeledImage[i, j] = lab = find(lab);

                    Sino
```




```
                          // CASO 7
                          // crear etiqueta
                          if (labelCount == labelEq.Capacity)
                               return true;
                          labelCount++;
                          add(labelCount);
                          labeledImage[i, j] = labelCount;

                    FinSi
              FinSi
        FinSi
  FinSi
Sino
  CASO 0
  thresholdedImage[i, j] = backgroundBrightness;
  labeledImage[i, j] = 0;
FinSi

Fin
```

En las figuras la casilla central marcada con una **x** representa el píxel que se quiere etiquetar en esa iteración. Las casillas marcadas con un **> 0** representan que ese píxel es considerado "objeto" y ya ha sido etiquetado, es decir, ya pertenece a un subconjunto disjunto del MFSET. Las casillas a **0** representan que ese píxel es "fondo", por lo que sus etiqueta es 0 (no pertenece a ningún subconjunto). También hay que remarcar que para asignar una etiqueta al píxel actual **x** tan solo es necesario tener en cuenta los 3 superiores y el de su izquierda, los demás no son analizados en ningún momento.

A modo de conclusión, una vez finalizado el proceso de segmentación de una imagen, obtenemos la imagen etiquetada y el número de etiquetas u objetos de la imagen así como la correspondencia de etiquetas en el MFSET. Además de manera adicional se obtienen una serie de características de los objetos tales como rectángulos de recorte y otras propiedades que serán de ayuda en la siguiente fase, la detección de marcadores candidatos.

3.2.1.2 Detección de marcadores candidatos

Tras el etiquetado de la imagen, el siguiente paso es analizar cada uno de los objetos encontrados en la misma con el objetivo de determinar si estos objetos son marcadores de la librería o no. A continuación se explicarán los detalles de cómo se realiza la detección de marcadores candidatos, apoyándose en el diagrama de flujo de la Figura 14.

La detección de marcadores candidatos puede verse como un bucle donde se inspecciona cada uno de los objetos encontrados en el etiquetado uno a uno. Si no se ha obtenido ninguna etiqueta u objeto en el proceso de etiquetado, no habrá nada que detectar y esta etapa finalizará con un número de marcadores detectados igual a 0. De lo contrario se procederá a comprobar ciertas características de los objetos para determinar si son marcadores o no.

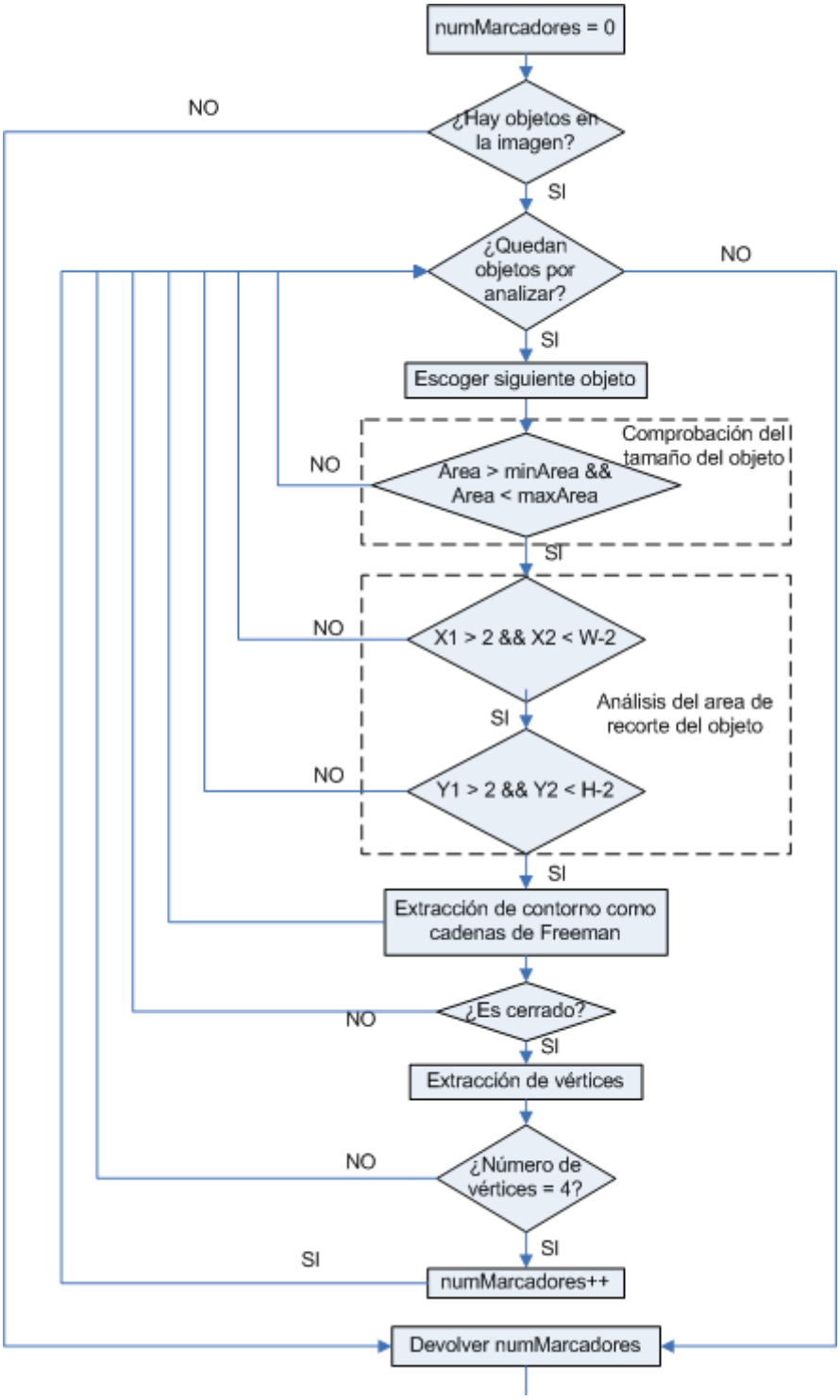


Figura 14. Diagrama de flujo de la detección de marcadores candidatos.

Al mismo tiempo que se ha ido etiquetando los distintos objetos de la imagen se han ido calculando y recogiendo ciertas características del mismo, entre ellas el tamaño y el área de recorte. De esta manera el objeto se considerará objeto si el área que ocupa dentro de la imagen está dentro de un máximo y un mínimo establecidos. También ha de comprobarse que el objeto etiquetado no está tocando ninguno de los bordes de la imagen. Estas son las primeras comprobaciones y las más sencillas. Posteriormente se pasará a tener en cuenta no sólo el tamaño o posición de un objeto, sino también su forma.

Para empezar obtenemos la descripción de los contornos de los objetos cuyo contorno es cerrado. Para aquellos que lo cumplan se trata de extraer los cuatro vértices que se supone que un marcador cuadrado ha de tener. Una vez extraídos los vértices se realiza un ajuste de los cuatro segmentos que componen el borde del marcador, con el objetivo de verificar que todos los píxeles de cada segmento se ajustan a una línea recta para finalmente obtener las esquinas definitivas de los marcadores. Todas las comprobaciones que se realizan a estos objetos son concluyentes, es decir, si una falla el objeto deja de considerarse marcador y no se le aplican más pruebas.

A continuación se explican con más detalle la obtención de la descripción del contorno y el algoritmo de extracción de vértices.

3.2.1.2.1 Descripción de contornos

Para poder determinar la forma de un objeto es necesario extraer información de su contorno. Para la descripción de contornos se ha seguido una aproximación basada en códigos de Freeman, pero obteniendo una secuencia de coordenadas de cada píxel en lugar de una secuencia Freeman ya que la posición de los píxeles del borde son datos relevantes para etapas posteriores.

El código Freeman, en su versión 8d, es un código que se emplea para la descripción sintáctica de contornos atendiendo a variaciones de su dirección relativa. El código se sintetiza en la Figura 15:

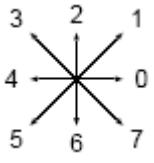


Figura 15. Código Freeman 8d

Un contorno queda descrito completamente con el punto inicial "S", y la secuencia de Freeman como puede verse en el ejemplo de la Figura 16.

La búsqueda de contorno para la descripción mediante el código de Freeman puede hacerse tanto en sentido horario como antihorario. Aquí vamos a

describir un algoritmo que es capaz de obtener la descripción del contorno en ambos sentidos (ver Listado 3).

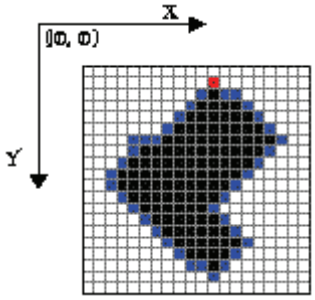


Figura 16. Contorno de un objeto.

La descripción del contorno de la figura anterior como cadena de Freeman sería 70777755555577755534333333332112001111.

La búsqueda del contorno se apoya en una tabla de "Freeman" que nos indica en cada momento en qué dirección debe seguir la búsqueda:

	Sentido Horario		Sentido Antihorario	
Dirección	**Tras Acierto**	**Tras Fallo**	**Tras Acierto**	**Tras Fallo**
0	3	7	5	1
1	4	0	6	2
2	5	1	7	3
3	6	2	0	4
4	7	3	1	5
5	0	4	2	6
6	1	5	3	7
7	2	6	4	0

Pongamos por ejemplo que la búsqueda se está realizando en sentido horario y estamos siguiendo la dirección 0. Si el píxel que se analiza siguiendo esa dirección es objeto, lo consideraremos como un acierto y escogeremos la nueva dirección de la columna *Tras Acierto* y la fila 0. Por lo tanto la nueva dirección a seguir será 3. Por el contrario si al analizar un píxel en determinada dirección resulta ser un píxel del fondo, la columna que deberemos consultar será *Tras Fallo*.

En el momento de iniciar la búsqueda, valores adecuados para la dirección inicial para el algoritmo definido pueden ser:

	Sentido Horario	Sentido Antihorario
Dirección	7, o 3 (revés)	5, o 1 (revés)

Listado 3. Extracción de contorno

```
Entradas:
imagen etiquetada de tamaño HxW: I
etiqueta de la componente conexa de la que queremos obtener su
contorno: etiq
```

```
Sentido de la búsqueda: horario o antihorario: Sentido

Salidas:
punto inicial Start: S
una secuencia de Freeman: Cadena y CadenaCoord
lógico que indica si la cadena es Cerrada: Cerrada

Inicio

Recorrer por filas la imagen hasta encontrar S, el primer punto
etiquetado con etiq.

P_ult = S;
P = S;
Acabar = falso;
Dirección = FreemanInicial (Sentido);
Cadena = {};
CadenaCoord = {};
Cerrada = falso;

Mientras (no acabar)
Hacer
      Dirección = Freeman (Dirección, Sentido, "Tras acierto");
      i = 1;
      encontrado = falso;
      Mientras (i<=8 && !encontrado) hacer
            P = Ady (P_ult, Dirección);
            Si (Etiqueta(P) = etiq) entonces
                  encontrado = cierto;
                  Si (P no es S) entonces
                        P_ult = P;
                        Cadena = Cadena · {Dirección};
                        CadenaCoord = CadenaCoord · P_ult(x,y)
                  Sino
                        acabar = cierto;
                        Cerrada = cierto;
                  FinSi
            Sino
                  Dirección = Freeman (Dirección, Sentido, "Tras
fallo");
                  i = i + 1;
            FinSi
      FinMientras

      Si (no encontrado) entonces
            acabar = cierto;
            Cerrada = falso;
      FinSi

FinMientras
Fin
```

La función *Ady* devuelve el punto de la imagen adyacente en la dirección indicada. Y la función Etiqueta devuelve el valor de la etiqueta de un punto de la imagen determinado.

Una vez el algoritmo ha acabado obtendremos una secuencia de coordenadas de los píxeles del contorno, el número de píxeles que conforman el contorno y si es cerrado o no. En el caso de no ser cerrado no se procedería a continuar con más comprobaciones sobre los píxeles con esa etiqueta. En el caso de serlo se pasaría a la extracción de vértices.

3.2.1.2.2 Extracción de vértices

El objetivo de esta etapa de la detección de marcadores candidatos es averiguar si el objeto detectado es cuadrado, es decir, tiene cuatro vértices y se guarda cierta relación entre los píxeles que componen las aristas. Hasta el momento se sabe que el contorno del objeto es cerrado y además contamos con la secuencia de coordenadas de los píxeles del contorno.

$$\mathbf{B = P_1 \cdot P_2 \cdot \ldots \cdot P_3 \equiv}$$
$$\mathbf{B = (x_1, y_1)\ , (x_2, y_2) \ldots (x_N, y_N)}$$

Para determinar el número de vértices que tiene el contorno se sigue una estrategia basada en distancias Euclídeas. Supongamos que tenemos una imagen con un objeto cuyo contorno es como el que muestra la Figura 17.

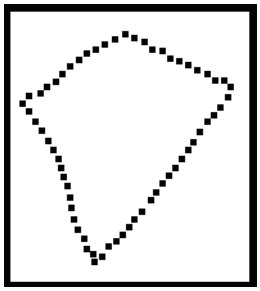


Figura 17. Píxeles del contorno de un marcador candidato

Inicialmente se calculan las distancias entre todos los píxeles del contorno dos a dos. Los dos píxeles más alejados entre ellos serán dos vértices definitivos del conjunto de vértices final. Los dos vértices restantes se buscan entre los píxeles intercalados entre los dos vértices definitivos ya localizados. Esta búsqueda viene guiada por las distancias entre cada píxel y la "cuerda" trazada entre los dos vértices ya localizados, como indica la siguiente figura:

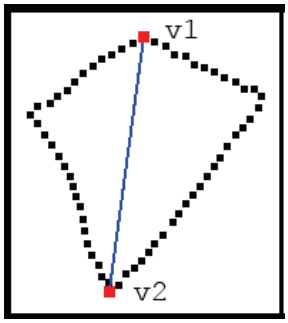


Figura 18. Vértices definitivos y segmentos a procesar.

Para averiguar si alguno de los píxeles del segmento que se está analizando es considerado vértice, es necesario establecer un valor umbral. Si la distancia entre un píxel y la "cuerda" no supera el valor umbral no se considerará píxel, sino, el píxel más alejado pasará a formar parte del conjunto de vértices definitivos y se procederá a hacer el mismo análisis de los dos segmentos de contorno que el vértice encontrado separa, como muestra la Figura 19 (a).

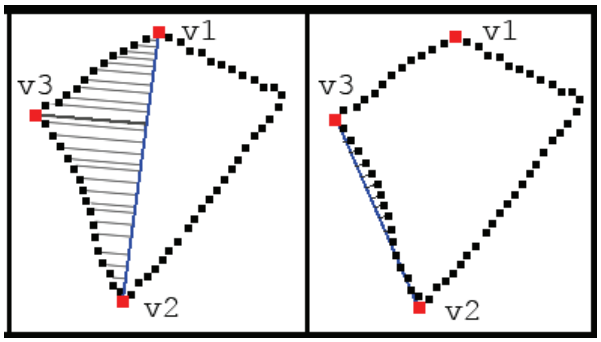


Figura 19. Vértices encontrados.

(a) **(b)**

Este proceso continúa hasta que o bien no se encuentran más vértices como en la Figura 19 (b) (todos las distancias analizadas son inferiores al valor umbral) o ya no quedan más píxeles de contorno para analizar. Si el número final de vértices obtenido por el algoritmo difiere de cuatro el objeto se descartará como marcador, en caso satisfactorio, el marcador se añadirá al conjunto de marcadores candidatos.

3.2.1.3 Identificación del marcador

Una vez se hayan detectado marcadores candidatos en la etapa anterior, será momento de determinar si efectivamente estos se tratan de marcadores válidos en el sistema. Para cumplir con el cometido de identificar el marcador se debe proceder al análisis de la región interna de cada marcador candidato. Independientemente de cómo esté codificada la información en el interior del marcador, un paso imprescindible es el de normalizar la región que el marcador encierra de tal forma que se pueda recuperar la imagen interna deshaciendo la distorsión causada por la captura de la imagen en una vista en perspectiva. Para abordar esta normalización se utilizan métodos de geometría proyectiva que nos permitirán muestrear la región interna obteniendo la imagen interna libre de distorsión. Será entonces cuando de acuerdo a la librería de marcadores de nuestro sistema, y de acuerdo a qué tipo de información transporte el marcador, se proceda a determinar si el marcador candidato es efectivamente un marcador, devolviendo finalmente el identificador del marcador correspondiente en caso afirmativo.

En los siguientes puntos se procede a desarrollar cada uno de los elementos que intervienen en la etapa de identificación de marcadores y cómo los

procesos implicados llevan a cabo su función. En particular se presenta el diseño de la librería de marcadores que el sistema soporta y cómo se codifican los identificadores de los mismos, de acuerdo a las técnicas de codificación digital que se han contemplado. Para finalizar se presenta cómo se lleva a cabo la normalización de la región interna del marcador y la obtención del identificador codificado allí.

3.2.1.3.1 La Librería de Marcadores

Los sistemas de marcadores han utilizado tradicionalmente ciertos patrones para diferenciar un marcador del resto dentro del sistema. En la actualidad, los patrones más comunes son los basados en imagen, popularizados gracias al extendido uso de ARToolkit [Artw3][Kat99].

Los patrones basados en imágenes tienen la ventaja de ser fácilmente interpretables por los humanos, ya que permiten asociar símbolos e imágenes relevantes para nosotros y por tanto hacen las aplicaciones más amigables. Todos los patrones deben ser conocidos por el sistema de detección, y deben ser de una u otra forma registrados previamente a su uso, en lo que se conoce como librería de marcadores. Sin embargo, desde el punto de vista técnico, el uso de imágenes como patrones determina en gran medida la forma en la cual la librería de marcadores los gestiona, y cómo la identificación del patrón se lleva a cabo. En particular, en ARToolkit la librería de marcadores no es más que un repositorio de imágenes de referencia que representan a los distintos patrones de imagen, almacenados todos ellos con las diferentes rotaciones y diversas condiciones de iluminación. Además, al tratarse de imágenes, y el no haber ningún fundamento matemático adicional subyacente que justifique la naturaleza ni forma del patrón a priori, la forma de averiguar si una imagen identificativa de un marcador se corresponde a alguno de los patrones registrados en la librería de marcadores se efectúa mediante la técnica de *correlación de imágenes*, evaluando todas las imágenes contenidas en la librería. Ello supone una importante sobrecarga en el sistema a medida que el número de marcadores registrados se incrementa, así como una tasa de error por confusión bastante elevada si los patrones no son suficientemente diferentes, y ante condiciones de iluminación diferentes de las contenidas en la librería.

Por todo ello, y tomando en consideración los estudios realizados en [Fia04], así como la estrategia allí adoptada, hemos decidido enfocar el subsistema de identificación desde el punto de vista de sistemas de comunicación digital. Un sistema de comunicación digital tiene la ventaja y la particularidad de utilizar ciertas técnicas de codificación que permitan, por una parte, verificar que los mensajes que el emisor envía llegan al receptor totalmente íntegros y sin defectos, y por otra parte, en caso de corrupción del mensaje ser capaz de corregir los errores en la medida de lo posible con el objetivo de evitar retransmisiones que colapsen el sistema de

comunicaciones. Bajo este enfoque, el emisor sería el propio marcador, el código del marcador se consideraría como el mensaje a transmitir, y el receptor sería el subsistema de detección. En cuanto al ruido, que eventualmente pueda haber en el canal y que consecuentemente pueda afectar a la transmisión del mensaje, se estaría considerando como tal los posibles defectos en la impresión de los marcadores, la luz en la captura, la presencia de otros objetos en la escena, la formación digital de la imagen en el sensor de la propia cámara, la perspectiva consecuencia del punto de vista del usuario, etc. Por lo tanto la librería de marcadores en nuestro sistema no consta de un repositorio en el sentido clásico de imágenes identificativas, sino que consta de una serie de códigos digitales y algoritmos que trabajan con ellos para la codificación y decodificación de identificadores de marcadores. A continuación se presenta el diseño del código y el proceso de codificación, que como resultado da soporte a nuestra librería de marcadores compuesta por un total de 512 marcadores. Ver apéndice A.

3.2.1.3.2 Diseño del marcador y codificación

El diseño físico del marcador que nuestra librería de RA basada en marcadores soporta consta de un borde cuadrado negro de anchura variable y de una serie de 36 símbolos digitales (blanco o negro) organizados en una rejilla de 6x6 ocupando la región interna al borde en su totalidad (véase Figura 20).

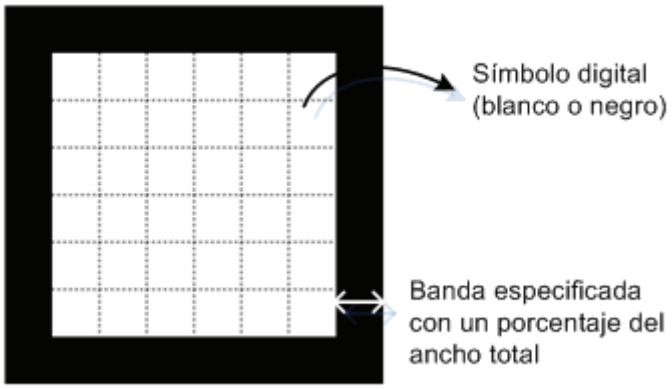


Figura 20. Diseño físico del marcador

El borde se utiliza en las etapas anteriores para localizar marcadores candidatos, así como para la extracción de los cuatro vértices que caracterizan la posición del marcador en la imagen capturada por la cámara. Los 36 símbolos digitales se utilizan exclusivamente para la identificación inequívoca del marcador por parte del sistema. De todos los posibles marcadores que podríamos tener en la librería sólo serán válidos los generados mediante el esquema de codificación mostrado en la Figura 21. Como muestra la figura el procedimiento para generar un marcador consiste en codificar un identificador de 9 bits mediante un código detector de redundancia cíclica (CRC) y el nuevo código resultante a su vez por otro

código convolucional corrector de errores hacia delante (FEC), resultando finalmente en una serie de 36 bits que se dispone directamente en el interior del marcador en el área correspondiente a los símbolos digitales. De esta descripción se deduce que nuestra librería estará limitada a 29=512 marcadores distintos, y que por lo tanto el resto de bits son redundantes con el objetivo de reducir los falsos positivos y la confusión en la identificación.

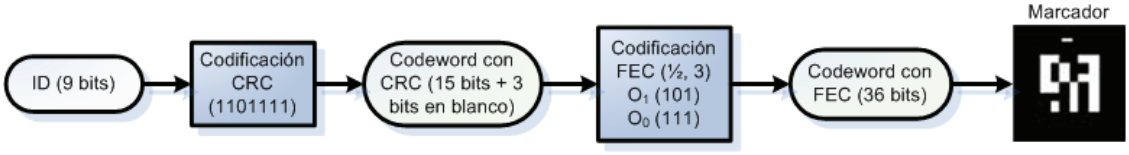


Figura 21. Proceso de codificación del identificador de un marcador

En el ámbito de las comunicaciones digitales es común el uso de dos estrategias básicas para el tratamiento de errores en la comunicación. Por una parte el uso de *códigos de detección de errores* persigue el objetivo de permitir deducir al receptor que un error ocurrió mediante la inclusión de cierta información redundante al mensaje a enviar. Mientras que por otra parte los *códigos de corrección de errores* capacitan al receptor la deducción de lo que el mensaje debería decir por la inclusión de la suficiente información redundante al mensaje a enviar. Cuando hablamos de error, dado que un mensaje es un una serie de *m* bits, nos estamos refiriendo a que en el receptor recibe una serie de *m* bits distinta a la enviada por el emisor. De manera general se define una *palabra código* (*codeword*) como una unidad de *n* bits que incluye los *m* bits del mensaje y los *r* bits redundantes para propósitos de detección y/o corrección de errores en el receptor.

El código utilizado en nuestra librería combina ambas estrategias de tratamiento de errores. La idea es que el FEC obtenga siempre un mensaje como resultado aún cuando hubo un error en la extracción de la palabra código desde el marcador, y en caso de que la corrección no sea correcta el código CRC nos avisará de tal situación.

Códigos de Redundancia Cíclica

Los códigos de redundancia cíclica (*Cyclic Redundancy Code o CRC*) se fundamentan en el tratamiento de cadenas de bits como representaciones de polinomios cuyos coeficientes son sólo 1 o 0. Teóricamente se describe una cadena de k bits como un polinomio con *k* términos yendo de x^{k-1} hasta x^0. La aritmética detrás de la teoría de CRC es la llamada modulo 2, que en esencia contempla la realización de sumas y restas sin acarreos. Esta aritmética tiene la ventaja de ser fácilmente implementable ya que el operador suma y resta se corresponden al operador lógico XOR, la multiplicación a la AND, y la división se puede implementar eficientemente mediante los operadores anteriores y los de desplazamientos de bits.

La idea subyacente a los códigos CRC es como sigue [Tan03]. El emisor y el receptor comparten un polinomio generador *G(x)*, convenientemente consensuado, y la propiedad que se desea verificar en el receptor con el propósito de valorar si el mensaje ha sido recibido correctamente es que el resto de la división polinómica del polinomio representando, el mensaje recibido, entre el polinomio generador debe ser cero. Para ello cuando el emisor quiere enviar un mensaje fuente *M(x)* de *m* bits, añade al final del mensaje una serie de *r* bits de redundancia que hagan cumplir la propiedad a verificar y consecuentemente el mensaje que finalmente se envía al receptor es un mensaje de *n* bits donde $n=m+r$. El Listado 4 muestra el algoritmo de codificación de un mensaje mediante un CRC.

Listado 4. Codificación CRC

```
Entradas
   Polinomio generador G(x)
   Polinomio del mensaje M(x)

Salidas
   Polinomio del mensaje codificado F(x)

Inicio
r = Grado (G(x));

// calculamos los r bits de redundancia
R(x) = resto (x^r·M(x) / G(x));
T(x) = x^r·M(x) + R(x)

Fin
```

Una vez el receptor recibe el mensaje *F(x)* de *n* bits, sólo queda realizar la división entre el polinomio *G(x)*, y verificar que el resto *R(x)* es 0 decidiendo entonces que no ha sucedido ningún error en la recepción del mensaje y procediendo a obtener el mensaje *M(x)* original. El Listado 5 muestra el algoritmo de decodificación de CRC. La justificación de este procedimiento es la siguiente: Se supone que se recibe *F'(x)*, siendo $F'(x) \neq F(x)$ si ha habido errores en la recepción. En tal caso $F'(x)=F(x)+E(x)$, donde *E(x)* es un polinomio representando los errores que han sucedido. Entonces se deduce la siguiente coimplicación que establece que solamente se obtendrá un resto distinto de 0 en el caso de que se produzcan errores en la recepción.

$R(x)=\text{Resto}((T(x)+E(x))/G(x)) \neq 0$ *si y sólo si* $\text{Resto}(E(x)/G(x)) \neq 0$

Listado 5. Decodificación CRC

```
Entradas
   Polinomio del mensaje codificado F(x)
   Polinomio generador G(x)

Salidas
   Polinomio del mensaje decodificado M(x)

Inicio
```

```
   R(x)= resto(T(x)/G(x));
   Si R(x) = 0 entonces
      M(x) = x^-r·T(x)
   Sino
      ERROR
   FinSi

Fin
```

Dado este planteamiento, se definen las siguientes propiedades de detección. Se dice que el código será capaz de detectar *errores simples* de la forma $E(x)=x^i$ si $G(x)$ contiene 2 o más *términos*. Similarmente se dice que será capaz de detectar *errores dobles* de la forma $E(x)=x^i + x^j$ con $i>j$ si $G(x)$ no es divisible entre x, y $G(x)$ no divide x^k+1 para cualquier k hasta el máximo valor de $i-j$. También se dice que será capaz de detectar un número impar de errores (*errores impares*) si $G(x)$ tiene como factor a $(x+1)$. Finalmente respecto a los *errores de ráfaga* de longitud k, $E(x)=x^{j+k}+\ldots+x^j$, esto es, una serie consecutiva de k bits completamente cambiada, se dice que para $k\leq r$ siempre se detectarían las ráfagas de k bits, mientras que para ráfagas de mayor longitud sólo se puede determinar la probabilidad con la cual éstas serían detectadas cuando sucedan[3] como $1-1/2^r$.

Teniendo en cuenta el funcionamiento teórico de los CRC, y que al final podríamos a lo sumo emplear hasta un número de 6 bits para CRC, se diseñó el polinomio generador $G(x)$ que cumpliese con las propiedades detectoras de los CRC y con las restricciones de espacio como $G(x) = x^6+x^5+x^3+x^2+x+1$, o lo que es lo mismo *1101111*.

Con la aplicación del CRC a nuestro identificador del marcador de 9 bits resultando en una palabra código CRC de 15 bits sería suficiente para una detectar una amplia variedad de errores que pudieran presentarse. Sin embargo, hay que tener en cuenta que la distorsión resultante de la perspectiva del punto de vista de la cámara introducirá con toda seguridad errores en el momento de extraer los símbolos digitales codificados en el marcador. Por tanto, si solamente se provee de un código de detección de errores sólo seremos capaces de descartar marcadores candidatos que no hayamos sido capaces de extraer correctamente, o bien descartar aquellos candidatos que efectivamente no son marcadores y por tanto los símbolos digitales contenidos en su interior difícilmente podrán pasar como un código válido. Considerando la naturaleza y los requisitos de un sistema de RA, en el que se persigue en la medida de lo posible reducir el fallo en el seguimiento de los marcadores para poder hacer de la superposición de la información una experiencia amigable y apetecible para el usuario, no sería muy conveniente emplear únicamente una estrategia basada puramente en la detección de errores. Hay que tener en cuenta que una estrategia de detección es útil cuando podemos hacer retransmisiones y obviamente éste

[3] La probabilidad con la que una mensaje recibido incorrectamente debido a un error de ráfaga de longitud mayor que r+1 sea aceptado como válido sin que el sistema se percate se calcula como $1/2^r$

no es el caso ya que en nuestro contexto se podría considerar que estamos ante un canal de comunicación de tipo simplex. De ahí que sería deseable el incorporar alguna técnica de corrección de errores que bajo ciertas condiciones adversas sea capaz de determinar el código del marcador que realmente identifica al marcador candidato. Para ello utilizamos una codificación adicional tras aplicar CRC basada en códigos de corrección de errores hacia delante (*Forward Error Correcting o FEC*).

Códigos de Corrección de Errores hacia delante

Un código de corrección de errores hacia delante, de aquí en adelante código FEC, es uno de los tipos de código correctores más populares utilizados en sistemas de comunicaciones digitales. Un código FEC se basa en los códigos convolucionales [Swe91][Glo03], en los cuales la palabra de código obtenida no sólo depende de los símbolos presentes a la entrada del codificador sino también de los anteriores que hayan sido procesados previamente. Los códigos convolucionales se especifican normalmente con tres parámetros (*n, k, K*), donde *k* es el número de bits de entrada en cada paso de codificación, *K* es el número de registros de desplazamiento de tamaño *k*, y *n* es el número de bits de salida establecidos como combinación lineal de los $k \cdot K$ bits presentes en el estado del codificador. Es habitual expresar el codificador como (k/n, K) donde k/n es conocido como el ratio o razón de código.

El funcionamiento de este tipo de codificadores es como sigue. Al codificador se le van introduciendo mensajes *m* de *k* bits, los *n* bits de salida se establecen entonces como una combinación lineal de los $k \cdot K$ bits de información residente en el codificador. Estas combinaciones lineales se especifican habitualmente mediante *n* polinomios. Los términos no-nulos de cada polinomio indican que los bits correspondientes en el codificador están conectados a un sumador sin acarreo que resultará en una de las salidas. Una vez generada la salida de *n* bits se introducirán los próximos *k* bits de entrada realizando un desplazamiento de *k* bits en los *K* registros. Obviamente los *K-1* mensajes de tamaño *k* anteriores permanecen en el estado del codificador y por esa razón se conocen a estos códigos como recurrentes. La Figura 22 muestra un esquema genérico de un codificador convolucional.

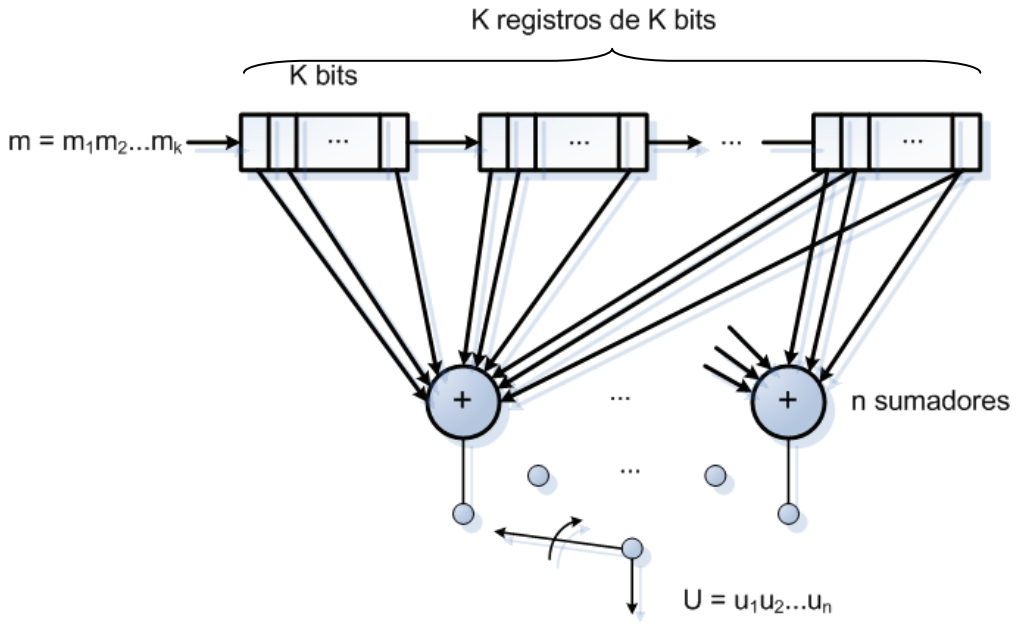


Figura 22. Esquema general de un codificador convolucional

Un concepto importante a la hora de valorar la calidad y la capacidad correctora es la de *distancia mínima de código*. Se define la *distancia de Hamming* entre dos *palabras de código* como el número de bits en que difieren. En particular, la mínima distancia de Hamming entre cualquier palabra código y la palabra nula, esto es, aquella con todos sus bits a 0s, se denomina distancia mínima de código. A esta distancia, en el contexto de la codificación convolucional se le suele denominar distancia libre mínima o simplemente distancia libre d_f. Se demuestra que la máxima capacidad de corrección de errores, t, se puede evaluar a partir de la distancia del código y se ajusta a la expresión $t = Floor[(d_f - 1)/2]$.

Esto significa que un decodificador podría ser capaz de corregir t errores si estos se presentaran, aunque dependiendo de las palabras código que se estén decodificando se podría llegar a corregir un número mayor de errores, si las mismas están suficientemente espaciadas (esto es así porque la distancia de Hamming entre ciertas palabras de código podría ser mucho mayor). Existen en la literatura estudios sobre códigos convolucionales óptimos [Ode76] usualmente para ratios $1/n$, y con n pequeño, para los cuales se ha demostrado que poseen muy buenas características a nivel de corrección de errores. En ese sentido dado que ya hemos utilizados 15 bits de los 36 que tenemos para codificar el código identificador original, hemos decidido emplear un codificador de ratio 1/2 de entre los ya etiquetados como óptimos por los estudios. En particular, hemos escogido el codificador más sencillo de los óptimos, (1/2, 3) con polinomios de conexión 101 y 111.

La Figura 23 muestra la estructura del codificador utilizado. Antes de poder utilizarlo se añaden 3 bits a los 15 bits que se obtenían de la codificación del identificador mediante el CRC para propósitos de vaciado del estado del

codificador. Esta secuencia de 18 bits se procesa en el codificador resultando en una palabra código de 36 bits.

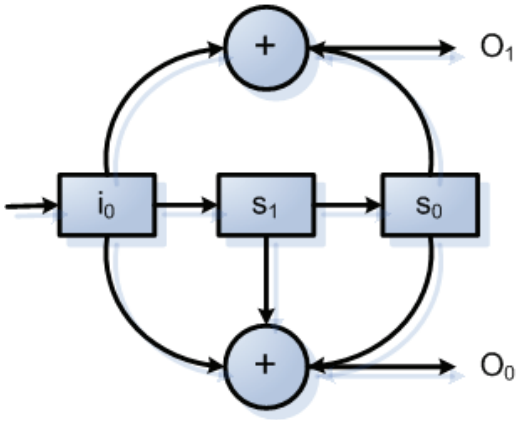


Figura 23. Estructura del codificador convolucional (1/2, 3) con vectores 101 y111

El comportamiento del codificador puede ser representado mediante la Tabla 1, y ésta a su vez puede ser representada mediante un autómata como el de la Figura 24. La particularidad de este autómata es que sus transiciones están etiquetadas con el símbolo excitador de la transición y los símbolos resultantes en las salidas del codificador.

Tabla 1. Comportamiento del codificador convolucional

Input Bit	Input State		Output Bits			
I_0	**S_1**	**S_0**	**V_1**	**V_0**	**O_1**	**O_0**
0	0	0	0	0	0	0
1	0	0	1	1	1	0
0	1	0	0	1	0	1
1	1	0	1	0	1	1
0	0	1	1	1	0	0
1	0	1	0	0	1	0
0	1	1	1	0	0	1
1	1	1	0	1	1	1

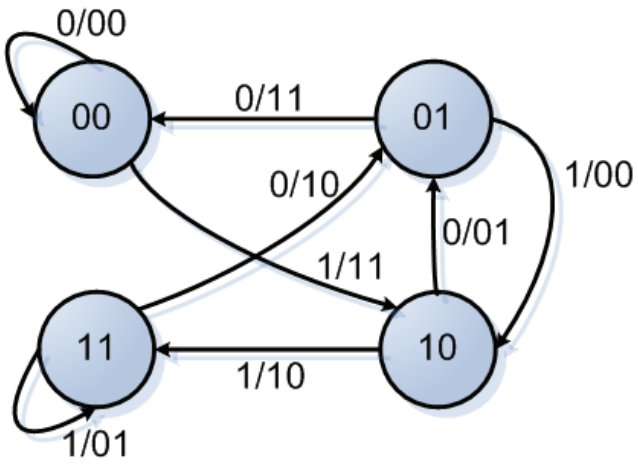


Figura 24. Autómata correspondiente al codificador convolucional empleado

Por lo tanto, la palabra de 18 bits de entrada será entonces codificada de acuerdo al código convolucional anterior obteniendo una palabra código de 36 bits. Esta palabra código es la que definitivamente se representa en los 36 símbolos digitales del marcador rellenando por filas de grupos de 6 bits.

La decodificación de un código FEC se suele realizar mediante decodificadores basados en Viterbi, que proveen una aproximación de máxima verosimilitud a la decodificación ante la presencia de errores en las palabras código recibidas. De este proceso se hablará en el punto 3.2.1.3.5, aunque se pueden encontrar más detalles en [Lan99].

En el punto 3.2.4 se introducirá la aplicación implementada para la generación e impresión de marcadores de acuerdo a las características de los codificadores utilizados para ser utilizados tanto en las aplicaciones de RA de destino como en la etapa de calibración de la cámara.

3.2.1.3.3 Extracción de la palabra código

Hemos visto en los puntos anteriores todo lo concerniente a la librería de marcadores y al diseño del código que se emplea para la codificación del identificador del marcador en su interior. Cuando el sistema está en producción, es decir, cuando se está ejecutando y detectando marcadores candidatos, una operación crítica es la extracción de la palabra código de la región interna del marcador y su posterior decodificación para obtener el identificador. El hecho de descansar sobre métodos digitales de codificación para la identificación de los marcadores no sólo nos permite ser robustos frente otras técnicas clásicas de reconocimiento de formas sino también nos permite sustituir la utilización de las mismas para determinar si un determinado objeto es efectivamente un marcador. El proceso de extraer la palabra de código del marcador candidato y decodificarla se muestra en la Figura 25.

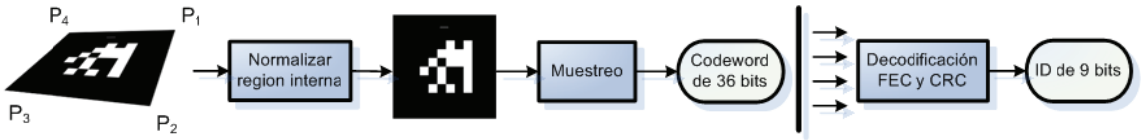


Figura 25. Extracción de la palabra código y su decodificación

Al inicio del proceso se tiene una imagen etiquetada y la lista de marcadores candidatos localizados en la misma, cada uno con 4 vértices que son considerados como sus 4 esquinas. Será entonces cuando mediante un proceso conocido como normalización se intente deshacer la distorsión sufrida por el marcador en la imagen. El objetivo de esta normalización no es otra que ser capaces de muestrear los 36 símbolos digitales que se suponen en el interior del marcador, para finalmente llevar a cado la decodificación FEC y la decodificación CRC resultando en el identificador de 9 bits del marcador.

3.2.1.3.4 Normalización y muestreo de la región interna

Fijémonos en la Figura 26. En ella se muestra un marcador capturado *(a)* y el mismo marcador libre de distorsiones *(b),* adicionalmente hemos representado la estructura en *(c)* y *(d)*, haciendo especial hincapié en la forma de los bordes y la posición de los vértices. Si quisiéramos obtener la relación entre *(c)* y *(d)* desde un punto de vista geométrico estaríamos ante un caso de transformación proyectiva 2D. Este tipo de transformaciones tratan sobre la proyección de una plano 2D a otro plano, y consecuentemente la transformación de los puntos contenidos en el primer plano al ser proyectados de la misma forma en el segundo. La relación entre ambos planos que nos permita establecer la correspondencia entre cualesquiera de los puntos 2D viene determinado por una matriz H de dimensión 3x3 llamada homografía y la ecuación proyectiva $x_i' = H \cdot x_i$. La obtención de la homografía H a partir de los vértices que configuran el marcador de tal forma que nos permita muestrear el interior de su región interna se reduce a la utilización del popular algoritmo en el ámbito de la geometría proyectiva para tal propósito conocido como *Direct Linear Transformation* (*DLT*). En el punto 3.2.2 se explica en detalle los fundamentos matemáticos de este tipo de transformaciones así como el algoritmo para obtener la homografía.

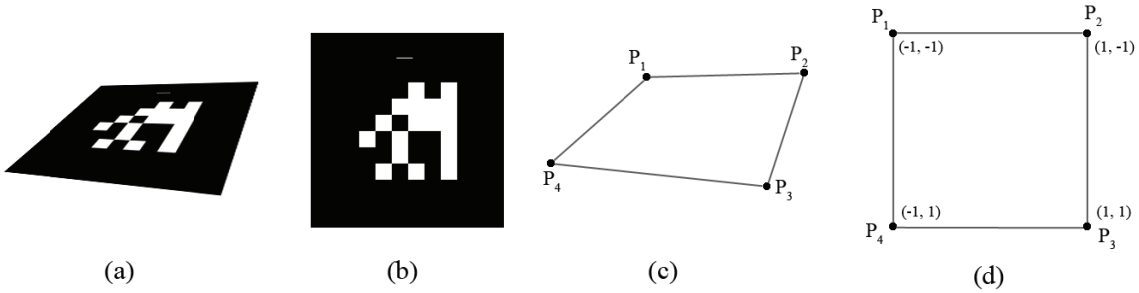


Figura 26. Proceso de normalización

Una vez obtenida la homografía H que relaciona los cuatro puntos del marcador candidato $\{x_i'\}$ con el modelo de marcador sin distorsión $\{x_i\}$, lo único que nos queda es muestrear la región interna por medio de la ecuación $x_i'=H \cdot x_i$ evaluando los puntos x_i en el modelo correspondientes a la región interna donde están ubicados los símbolos digitales para obtener la posición correspondiente en la imagen. Esto resultará en una palabra código de 36 bits que deberá ser decodificada. Evidentemente si no se pudiera obtener la homografía, o bien si los puntos resultantes del muestreo no fueran consistentes, el marcador candidato sería descartado ya que significaría que sencillamente se trataba de una región cerrada con cuatro vértices diferenciados resultante del etiquetado pero sin corresponderse realmente a una marcador.

3.2.1.3.5 Proceso de decodificación

La decodificación de una palabra código se lleva a cabo mediante el algoritmo presentado en el Listado 6. El algoritmo recibe como entrada la palabra código y se limita a llevar a cabo una decodificación FEC, que obtendrá un palabra de 18 bits sobre la que posteriormente se realizará la decodificación CRC de acuerdo a los polinomios de codificación que fueron especificados, devolviendo el código identificativo de 9 bits o un error, y además el grado de similitud de la palabra código evaluada con respecto a la palabra código más cercana que obtuvo la decodificación FEC.

Hay que tener en cuenta que una vez disponemos de los 36 bits extraídos procedentes del interior del marcador candidato, será necesario reordenarlos para obtener cuatro posibles palabras código a evaluar. Esto es así debido a que la serie de 36 bits en el proceso de codificación fueron dispuestos en 6 grupos de 6 bits en el momento de establecer el valor de los 36 símbolos digitales en el marcador, y por tanto da lugar a 4 posibles lecturas diferentes de los bits. Por tanto, para conseguir el código identificador del marcador se decodifica cada una de las cuatro palabras código posibles mediante el algoritmo presentado en el Listado 7, y el código identificativo se corresponderá a aquél que haya obtenido mayor grado de similitud de los cuatro.

Listado 6. Decodificación de la palabra código extraída

```
Entrada
      codeword (36 bits)
Salida
      Id (9 bits)
      grado_similitud

Inicio
    [crcEncoded, grado_similitud] = FECDecode(codeword);
    Id = CRC6Decode(crcEncoded);
Fin
```

Decodificación FEC

De forma general el proceso de decodificación de un código convolucional se plantea como un problema de optimización en el que se pretende obtener la secuencia de símbolos que permita maximizar la probabilidad de obtener la cadena recibida *R* habiendo enviado realmente el mensaje *M*, o lo que es lo mismo, obtener la secuencia de símbolos que genere la cadena codificada de mayor verosimilitud a la recibida. Matemáticamente se corresponde a plantear la optimización según la expresión:

$$arg\max_{M} P(R \mid M) = arg\max_{M} \prod_{k=0}^{K-1} P(R_k|M_k)$$

Una forma de resolver eficientemente este problema es mediante la aplicación del algoritmo de Viterbi [Vit67][For73], que se ha generalizado para resolver problemas de optimización en modelos ocultos de Markov (HMM). La idea del algoritmo es generar un grafo multietapa a partir del grafo que modela el modelo de Markov. Sobre el *trellis*, o dicho de otra forma, la rejilla en la que se anotan los resultados intermedios de cálculo que representa el grafo multietapa, se procede a calcular las probabilidades de ir obteniendo la subsecuencia de *R* que se esté evaluando en cada etapa de forma incremental. Finalmente, se recorre el grafo multietapa hacia atrás desde el estado final en la última etapa hasta el estado inicial en la primera etapa, pasando por todas etapas siguiendo el camino de construcción que obtuvo la máxima probabilidad y recuperando así la secuencia de símbolos que se supone generadora de la cadena evaluada R.

Paradójicamente, en la bibliografía se sigue la formalización anterior pero sin explicitar la obtención de las probabilidades condicionales ni proveyendo explícitamente un grafo de Markov en los ejemplos sobre codificadores. En cualquier caso es obvio que se puede calcular la probabilidad de una transición en función del número de transiciones salientes de un estado y la diferencia entre los símbolos de la salida y los símbolos siendo evaluados. No obstante, nosotros hemos preferido replantear el problema desde el punto de vista de una métrica Hamming obtenida como la similitud entre los símbolos a evaluar y los símbolos esperados. Para ello definimos formalmente los grafos como los de la Figura 24 como $G=(V, A)$. Σ es el conjunto de símbolos de excitación de las transiciones, es decir, el conjunto de símbolos de entrada, $\Sigma=\{s_1, s_2, \ldots\}$. Θ es el conjunto de símbolos de salida producidos por las transiciones, $\Theta=\{\sigma_1, \sigma_2, \ldots\}$. V es un conjunto finito de estados $V=\{e_1, e_2, \ldots\}$, y A es un conjunto finito de transiciones $A=\{t_1, t_2, \ldots\}$. Una transición t_i se define como una 4-tupla de la forma $(e_o, s_u, e_d, \sigma_v)$ indicando que si estando en el estado e_o y sucede el evento s_u entonces se transita al estado e_d generando en la salida el símbolo σ_v. La nueva formalización tiene en cuenta la estructura del *trellis* utilizado en Viterbi de forma que su implementación sea prácticamente directa a partir de la definición de la siguiente recurrencia y teniendo como secuencia a decodificar a $\hat{\sigma} = \widehat{\sigma_1} \cdot \widehat{\sigma_2} \cdot \ldots \cdot \widehat{\sigma_j}$:

$$S(e,j) = \begin{cases} 0, si\ e = e_0\ y\ j = 0 \\ -\infty, si\ e \neq e_0\ y\ j = 0 \\ \max\limits_{(e',s,e,\sigma)\in A} S(e', j-1) + \ similitud\ (\sigma, \widehat{\sigma_j}), en\ otro\ caso \end{cases}$$

El grafo multietapa constaría de J+1 etapas donde J sería la longitud de la secuencia a decodificar. Cada etapa *j* contiene una replicación de los estados *V* de *G*, y las transiciones *A* unirían una etapa con la siguiente. Los símbolos de Θ son combinaciones de bits, y consecuentemente la función similitud se define de tal forma que devuelve la cantidad de coincidencias de bits entre sus dos argumentos. En nuestro caso, Σ*={0,1}*, Θ*={00, 01, 10, 11}*, *G=({e00, e01, e10, e11}, {(e00, 0, e00, 00), (e00, 1, e10, 11), (e01, 0, e00, 11), (e01, 1, e10, 00), (e10, 0, e01, 01), (e10, 1, e11, 10), (e11, 0, e01, 10), (e11, 1, e11, 01)})*, y la representación de una etapa sería como se presenta en la Figura 27, formando un grafo multietapa con 18+1 etapas, siendo la invocación inicial necesaria para resolver el problema *S(e00, J)*.

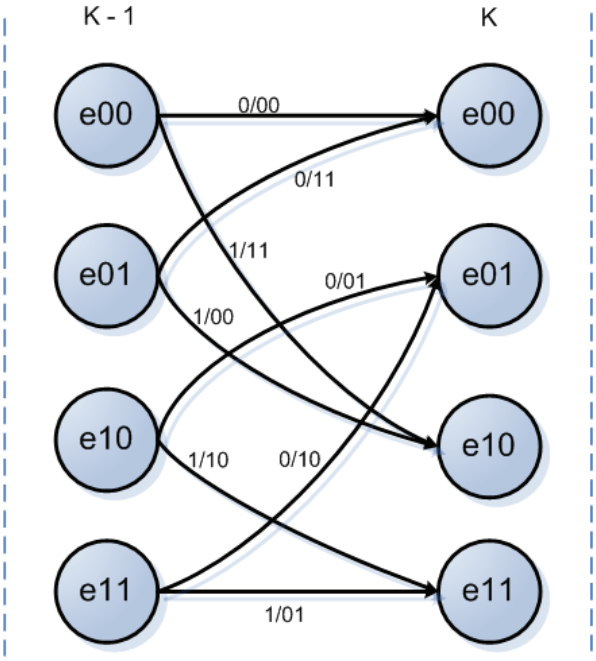


Figura 27. Representación de la etapa en el grafo multietapa del decodificador

Tras haber visto el funcionamiento de cada decodificador y los fundamentos de los códigos CRC y FEC se podría pensar que en lugar de utilizar una aproximación combinada para la codificación se podría utilizar tan sólo una de las técnicas utilizadas, en particular la basada exclusivamente en un código FEC. Sin embargo, la justificación para utilizar una combinada es que el decodificador FEC siempre devuelve un resultado ya que está llevando a cabo una decodificación siguiendo un criterio de verosimilitud, pero eso no significa que siempre acierte a devolver la palabra de 18 bits que realmente originó la palabra código si se produjeron errores en la extracción. Por tanto el hecho de haber codificado con un código CRC el código

identificativo permitirá detectar cuando este problema se presenta y así estaremos haciendo uso de un mecanismo adicional de protección.

3.2.2 Geometría de visión monocular

Esta sección tiene como objetivo establecer los conceptos básicos que serán necesarios a lo largo de los puntos que le suceden. Comenzamos definiendo el sistema de coordenadas Cartesiano, la definición de punto 2D y 3D, continuaremos con coordenadas homogéneas y la definición de proyección 3D a 2D para acabar introduciendo el modelo de cámara utilizado en el proyecto partiendo del modelo de cámara *pinhole*. Estos conocimientos serán de utilidad en los sucesivos puntos cuando se introduzca la estimación de pose y orientación, así como el proceso de calibración de la cámara.

Aplicados a la geometría, física o ingeniería, un *sistema de coordenadas* se define como un sistema en el que se asigna un conjunto de números –normalmente pertenecientes al dominio de los reales- a cada uno de los puntos de un espacio *n*-dimensional. Aunque podemos emplear cualquier sistema de coordenadas para cálculos numéricos en el espacio, el sistema de coordenadas escogido representa únicamente un marco de referencia para dichos cálculos, ya que se considera que el *espacio* en sí mismo existe independientemente de ese marco.

Refiriéndonos a un espacio de tres dimensiones, el uso del sistema de coordenadas Cartesiano permite determinar unívocamente cada punto en el espacio a través de tres números, comúnmente denominados la *coordenada x, y, z* de ese punto. Por lo que un punto 3D P queda definido como:

$$P = (x_p, y_p, z_p)$$

Para definir las coordenadas se especifican, por un lado, tres ejes compuestos por líneas perpendiculares dos a dos, que poseen una dirección asignada. Por otro lado, se debe proporcionar la longitud que representa una unidad de medición.

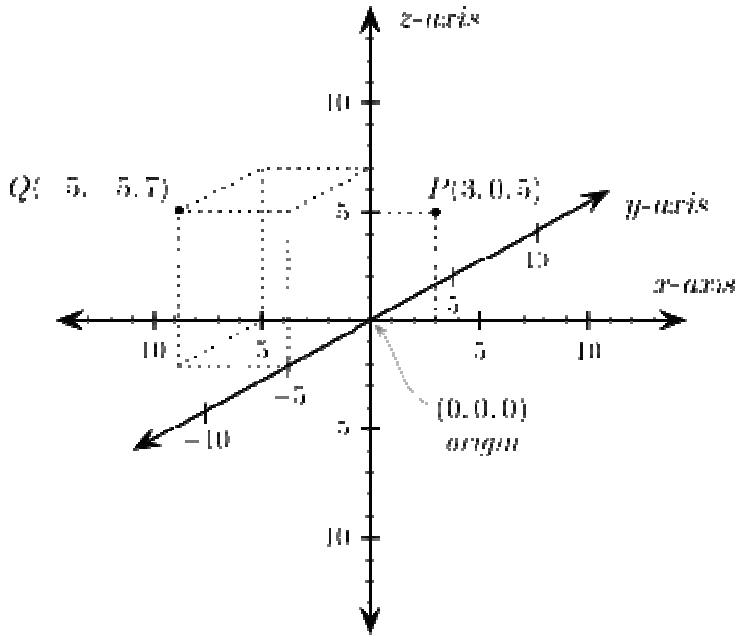


Figura 28. Sistema de coordenadas Cartesiano de 3 dimensiones

En un espacio bidimensional, el plano, las coordenadas Cartesianas están formadas por dos ejes *x* e *y*, normalmente denominados abscisas y ordenadas, que se cortan en el origen 0 cuyas coordenadas son, obviamente, (0,0).

La posición de un punto P en el plano será:

$P = (x_p, y_p)$

Por convención, el origen de un sistema de coordenadas en coordenadas Cartesianas es el punto (0,0,…,0), que puede asignarse a cualquier punto de un espacio Euclídeo.

El sistema de coordenadas Cartesiano es el más comúnmente empleado, no obstante, existen otras posibles representaciones que no son infrecuentes en los cálculos relacionados con la informática gráfica como el sistema de coordenadas polar.

3.2.2.1 Coordenadas homogéneas

Para introducir el concepto de coordenadas homogéneas es necesario explicar primero el porqué de su uso: las transformaciones. Las transformaciones son herramientas fundamentales en la generación de gráficos tridimensionales. Se emplean para mover objetos en el espacio, así como para construir una vista bidimensional del entorno 3D que pueda transferirse a un dispositivo de visualización. El conjunto de puntos tridimensionales pertenecientes a un objeto puede transformarse en otro conjunto a través de una transformación lineal. Ambos conjuntos

permanecen en el mismo sistema de coordenadas. En gráficos por computador se emplea la notación matricial para describir las transformaciones, y normalmente se emplea como convención la representación de los puntos como vectores columna.

Así pues, una transformación compuesta de un punto **P** que da lugar a un nuevo punto **P'**, en su forma general, tendría la forma:

$$\mathbf{P' = RP + t} \qquad \textbf{(1)}$$

Donde **R** es una matriz invertible de 3x3 que contiene la información de ángulos y factores de escala y, **t** es un vector 3D de traslación que contiene los términos de traslación asociados al punto fijo y al centro de rotación. . La realización de dos operaciones distintas en la ecuación (1) da lugar a un desarrollo relativamente engorroso en la concatenación de transformaciones:

$$\mathbf{P'' = R_2\,P' + t_2 = R_2\,(R_1\,P + t_1) + t_2 = (R_2\,R_1)\,P + R_2\,t_1 + t_2} \qquad \textbf{(2)}$$

Obligándonos a llevar la cuenta de las componentes matriciales $\mathbf{M_nM_{n-1}}$, así como la componente de traslación $\mathbf{M_nt_{n-1}} + \mathbf{t_n}$ en cada una de las fases cuando se concatenan *n* transformaciones.

Buscamos una solución más eficiente que permita combinar las transformaciones para obtener directamente las coordenadas finales a partir de las iniciales.

En un espacio bidimensional, dado un vector $\vec{P}_{2D} = \langle Px, Py \rangle$, según lo visto anteriormente, la obtención de su homólogo en coordenadas homogéneas, $\tilde{\vec{P}}_{2D}$, pasa por añadir una componente adicional w = 1, $\tilde{\vec{P}}_{2D} = \langle Px, Py, 1 \rangle$. Generalizando para cualquier valor de w, es posible establecer un homomorfismo entre ambos, que define el punto en dos dimensiones, $\vec{P}_{2D}$, como el punto en que la línea que conecta $\tilde{\vec{P}}_{2D}$ con el origen de coordenadas interseca el hiperplano de 3 dimensiones descrito por w=1.

$$\vec{P}_{2D} = h(\tilde{\vec{P}}_{2D}) = h(\langle P_x, P_y, w \rangle) = \langle \frac{P_x}{w}, \frac{P_y}{w} \rangle \qquad \textbf{(3)}$$

Representado gráficamente:

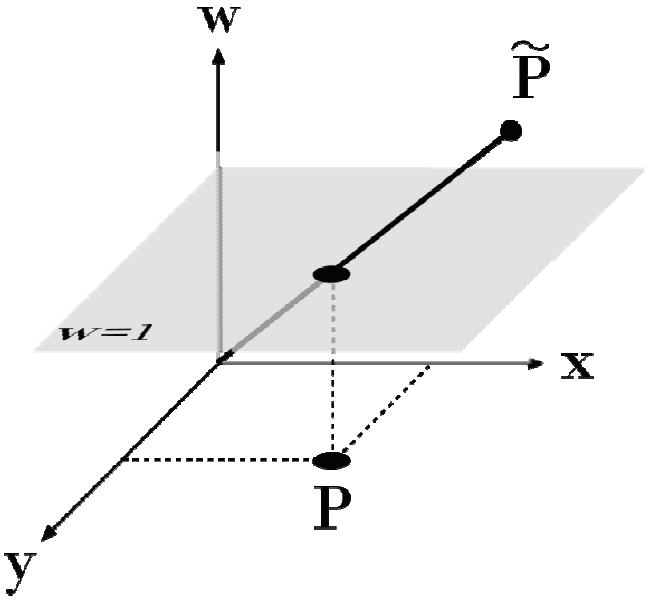


Figura 29. P como proyección al espacio x,y del punto en el espacio x,y,w

Así pues, en el caso de un punto de 3 dimensiones, su equivalente se describe en un subespacio de 4 dimensiones donde se sitúa la coordenada w, y el hiperplano $w = 1$.

El uso de coordenadas homogéneas permite tratar todas las transformaciones geométricas como una multiplicación de matrices de la siguiente manera:

$$\mathbf{P' = RP + t}$$

$$P = [R \quad | \quad t]\tilde{P}$$

$$[R \quad | \quad t] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.2.2.2 Transformación proyectiva y homografía 2D

Una transformación proyectiva transforma un plano en otro plano equivalente en el que se conservan todas las propiedades invariantes a las proyectividades. Estas transformaciones modelan la distorsión geométrica que se introduce sobre un plano cuando se toma una imagen del mismo con una cámara de perspectiva. Bajo una cámara de perspectiva, algunas propiedades geométricas se conservan, tales como colinealidad (una línea recta se proyecta en una recta), mientras otras propiedades no, por ejemplo paralelismo, en general las líneas paralelas no se presentan como tales en la imagen.

Una homografía 2D es el resultado de calcular la transformación proyectiva que convierte un conjunto de puntos $\mathbf{x}_i$ en $\mathbf{x'}_i$, su correspondiente conjunto de puntos también en el plano. En la práctica, los puntos de $\mathbf{x}_i$ y $\mathbf{x'}_i$ son puntos de dos imágenes o de la misma, cada imagen considerada como un plano proyectivo en el plano.

La transformación proyectiva se define como sigue. Una transformación proyectiva entre planos es una transformación lineal sobre 3-vectores homogéneos, representada por una matriz 3x3 **H** de tal forma que, **x'= Hx**.

$$h = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix} \quad H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{4}$$

Es importante resaltar que dado el carácter homogéneo de los vectores, la matriz **H** puede multiplicarse por una constante sin que la transformación se modifique. Por tanto la matriz **H** también es de tipo homogéneo y esta definida salvo una constante de proporcionalidad. Como consecuencia la matriz **H** tan solo posee 8 elementos independientes, ya que uno de ellos lo fija la constante de proporcionalidad.

A la hora de calcular una homografía 2D **H** es necesario saber cuántas correspondencias entre puntos $\mathbf{x}_i \leftrightarrow \mathbf{x'}_i$ son necesarias. Por una parte, y como ya se ha comentado, la matriz **H** tan solo posee 8 grados de libertad por lo que el número de grados de libertad de una transformación proyectiva 2D es 8. Por otro lado, el número de grados de libertad de un punto 2D es 2, que corresponden a las componentes *x* e *y*. Dado que en una correspondencia entre dos puntos, los dos grados de libertad de $\mathbf{x}_i$ de la primera imagen se corresponden con los dos grados de libertad del punto mapeado en la segunda imagen $\mathbf{Hx'}_i$ es necesario especificar como mínimo cuatro correspondencias entre puntos para poder definir completamente la matriz **H,** y ser capaz así de computar la homografía que mapea los puntos 2D en ambos planos.

El método propuesto, conocido como *Direct Linear Transformation* (DLT) ([Har03], p. 90), consiste en, dadas $n \geq 4$ correspondencias de puntos 2D a 2D $\{x_i \leftrightarrow x'_i\}$, determinar la homografía 2D, matriz **H**, tal que $x_i = \mathbf{H}x'_i$. Esta ecuación puede escribirse de la siguiente manera siendo *s* un factor de escala:

$$s\cdot\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix}$$

Operando obtenemos las siguientes ecuaciones:

$$s\,x_i = h_1\,x'_i + h_2\,y'_i + h_3$$
$$s\,y_i = h_4\,x'_i + h_5\,y'_i + h_6$$
$$s = h_7\,x'_i + h_8\,y'_i + h_9$$

Resolviendo por sustitución el sistema de ecuaciones anterior considerando $h_9 = 1$ por ser la componente homogénea de la matriz H, y escribiéndolo de forma matricial obtenemos la siguiente ecuación:

$$\begin{pmatrix} x'_i & y'_i & 1 & 0 & 0 & 0 & -x'_i\,x_i & -x'_i\,y'_i \\ 0 & 0 & 0 & x'_i & y'_i & 1 & -x'_i\,y_i & -y'_i\,y_i \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \rightarrow A_i\mathbf{h} = \mathbf{b}_i$$

Apilando las ecuaciones para cada correspondencia de puntos dada se genera una ecuación matricial de forma A**h = b,** donde A tiene 8 columnas y $2n$ filas, y **b** es un vector con dimensión $2n$. El sistema de ecuaciones resultante puede resolverse mediante técnicas bien conocidas de resolución de ecuaciones lineales, como eliminación Gaussiana. En el caso de disponer de más de 4 puntos o correspondencias, el sistema resultante es sobredeterminado, i.e. tenemos un mayor número de ecuaciones que incógnitas. En ese caso, el sistema puede resolverse desde una perspectiva optimizante basada en técnicas de *mínimos cuadrados*, para lo cual se ha aplicado un método basado en la descomposición de valores singulares (SVD). Este método consiste en factorizar la matriz A de manera que SVD(A) $\rightarrow$ A = UDV^T donde U y V son matrices ortogonales y D es una matriz diagonal con entradas no negativas. Gracias a esta factorización en matrices es posible resolver el sistema sobredeterminado A**h** = **b** de la siguiente manera. A raíz del vector dado **b** y la matriz U resultante de la factorización calculamos el vector **b'** como **b'** = U^T**b.** Entonces, dividiendo cada entrada de **b'** por la i-ésima entrada de la matriz diagonal D obtenida al factorizar, obtenemos un vector **y**, es decir, y_i **= b'**/d_i. Finalmente el vector de incógnitas **h** queda resuelto por la expresión **h = Vy.** Como puede verse la matriz V ha sido obtenida por factorización SVD de la matriz A y el vector **y** como se ha explicado anteriormente ([Har03] p.588). Adicionalmente, dado que normalmente las correspondencias provienen de sistemas de coordenadas muy distintos y por lo tanto los valores de los puntos difieren en varios órdenes de magnitud, se realiza cierta normalización de datos de acuerdo a lo expuesto en [Har95] para mejorar los resultados numéricos.

3.2.2.3 Modelo de cámara

Una cámara actúa como un elemento que mapea el mundo 3D (los objetos del espacio) en una imagen 2D. A este mapeo se le denomina proyección 3D a 2D y se define como: Dado un conjunto de puntos $\mathbf{X}_i$ en el espacio 3D, y un conjunto de puntos correspondientes $\mathbf{x}_i$ en una imagen, encontrar la proyección 3D a 2D que haga coincidir $\mathbf{X}_i$ a $\mathbf{x}_i$. Este tipo de proyección de 3D a 2D es el tipo de proyección que produce una cámara perspectiva.

El mapeo de la cámara se representa como una matriz, y en el caso de mapeo de puntos, es una matriz **M** de 3 x 4 que transforma las coordenadas homogéneas de un punto del espacio 3D a las coordenadas homogéneas de ese mismo punto en el plano imagen. Esta matriz tiene normalmente 11 grados de libertad y permite extraer propiedades de la cámara tales como el centro y la distancia focal. En concreto, el ratio de aspecto o la distancia focal, que son características de las cámaras, conforman una matriz **K** de 3 x 3 que se obtiene aplicando una descomposición a la citada matriz **P**. Esta matriz **P** define el *modelo* de cámara.

Para introducir el modelo de cámara utilizado en el proyecto comenzaremos analizando el modelo más simple, el modelo *pinhole* para progresivamente ir generalizándolo hasta el utilizado.

El centro de proyección en el modelo *pinhole* es el origen de coordenadas del sistema Euclídeo, y consideramos el plano focal Z = *f*. En este modelo, un punto en el espacio con coordenadas $\mathbf{X} = (X, Y, Z)^T$ es mapeado a un punto en el plano focal donde la línea que une el punto **X** con el centro de proyección interseca con este plano, como muestra la Figura 30.

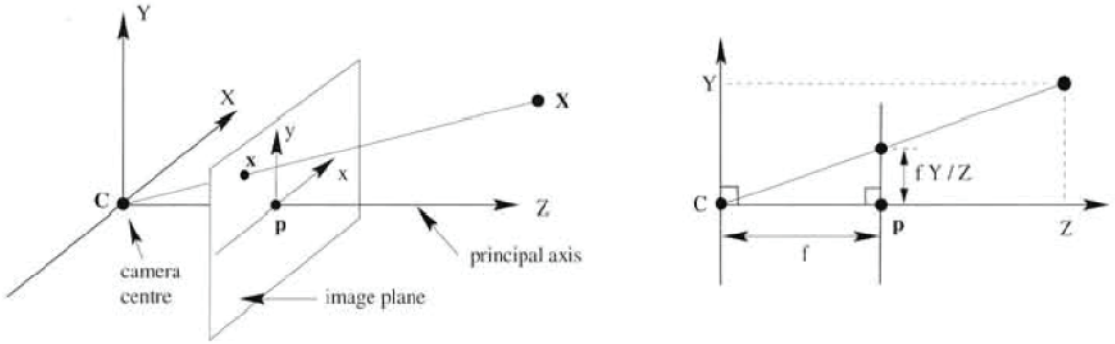


Figura 30. Geometría del modelo de cámara *pinhole*.

C es el centro de proyección de la cámara también conocido como *centro óptico*. La línea perpendicular que va desde el centro óptico hasta el plano de la imagen se llamar *eje principal* de la cámara, y el punto **p** donde el eje principal corta al plano imagen se llama *punto principal*. El centro de la cámara está situado en el origen de coordenadas.

El punto mapeado al plano imagen $\mathbf{x} = \mathbf{PX}$ puede reescribirse como:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Por lo que la matriz **P** de la cámara para el modelo *pinhole* es una matriz homogénea 3 x 4 como la siguiente:

$$\mathbf{P} = diag(f, f, 1)\begin{bmatrix} \mathbf{I} & |0 \end{bmatrix}$$

Por ahora estamos asumiendo que el origen de coordenadas del plano imagen está en el punto principal **p**. En la práctica esto no tiene porqué ser así, puede haber un desplazamiento del punto principal (p_x, p_y) con respecto al origen de coordenadas del plano imagen. En este caso se podría escribir el mapeo del punto teniendo en cuenta el desplazamiento del punto principal de la siguiente manera:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad \textbf{(5)}$$

Si:

$$\mathrm{K} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

podríamos escribir (5) de la siguiente forma:

$$\mathbf{x} = \mathrm{K}[\mathrm{I}\,|\,0]\mathbf{X}_{cam}$$

Donde K es la *matriz de calibración de cámara*, I es una matriz identidad de 3 x 3 y 0 es un vector columna de 0s que representa la traslación. X_{cam} se utiliza para enfatizar que el punto **X** está en el sistema de coordenadas de la cámara, que en este caso coincide con el sistema de coordenadas del mundo dado que, como ya se ha comentado, el centro óptico está en el origen de coordenadas Euclídeo y el eje principal coincide con el eje Z.

En general, los puntos en el espacio pueden estar expresados en diferentes sistemas de coordenadas Euclídeos. Por tanto, dos sistemas de coordenadas Euclídeos como son el sistema de coordenadas de la cámara y el sistema de coordenadas del mundo, están relacionados por una rotación **R** y una traslación **t**.

Por lo tanto se puede generalizar y escribir que:

$$\mathbf{x} = \mathrm{K}[\mathrm{R} \mid \mathrm{t}]\mathbf{X}$$

siendo

$$\mathrm{P} = \mathrm{K}[\mathbf{R} \quad | \quad \mathbf{t}] \tag{6}$$

Este es el mapeo general dado por una cámara *pinhole.* Este modelo tiene 9 grados de libertad: 3 para K (los elementos *f, p_x, p_y*), 3 para **R** y 3 para **t**. Los parámetros contenidos en K se llaman *parámetros intrínsecos* de la cámara. Los parámetros de **R** y **t** que relacionan la orientación y posición de la cámara con el sistema de coordenadas del mundo se llaman *parámetros extrínsecos*.

El modelo *pinhole* asume que las coordenadas de la imagen son coordenadas Euclídeas teniendo el mismo factor de escala en ambos ejes. En el caso de cámaras CCD existe la posibilidad adicional de que los píxeles no sean cuadrados. De esa manera si las coordenadas de la imagen son medidas en píxeles, se introduce un factor de escala desigual en cada eje. En concreto, si el número de píxeles por unidad de distancia en coordenadas de la imagen es m_x y m_y para los ejes x e y respectivamente, la transformación de coordenadas del mundo a coordenadas de píxel se obtiene multiplicando la matriz K por la izquierda por un factor extra (m_x, m_y, 1). Por lo tanto la forma genérica de la matriz de calibración de una cámara CCD es:

$$\mathrm{K} = \begin{bmatrix} \alpha_x & 0 & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}$$

Donde $\alpha_x = fm_x$ y $\alpha_y = fm_y$ representan la distancia focal de la cámara en dimensiones de píxel. De la misma manera el $\tilde{\mathbf{x}}_0 = (x_0, y_0)$ representa el punto principal en dimensiones de píxel con coordenadas $x_0 = m_x p_x$ e $y_0 = m_y p_y$. Además, dos parámetros más, k_1 y k_2, son añadidos con el objetivo de modelar la distorsión radial de algunas cámaras.

Esta distorsión se define según un modelo radial definido por las expresiones:

$$\mathbf{x_d} + \mathbf{d_x} = \mathbf{x_u}$$
$$\mathbf{y_d} + \mathbf{d_y} = \mathbf{y_u}$$

con

$$\mathbf{d_x} = \mathbf{x_d}\,(k_1\mathbf{r}^2 + k_2\mathbf{r}^4)$$
$$\mathbf{d_y} = \mathbf{y_d}\,(k_1\mathbf{r}^2 + k_2\mathbf{r}^4)$$

y

$$\mathbf{r} = \sqrt{(x_d^2 + y_d^2)}$$

donde (x_d, y_d) son las coordenadas del píxel distorsionado y (x_u, y_u) del píxel sin distorsionar como puede verse en la Figura 31.

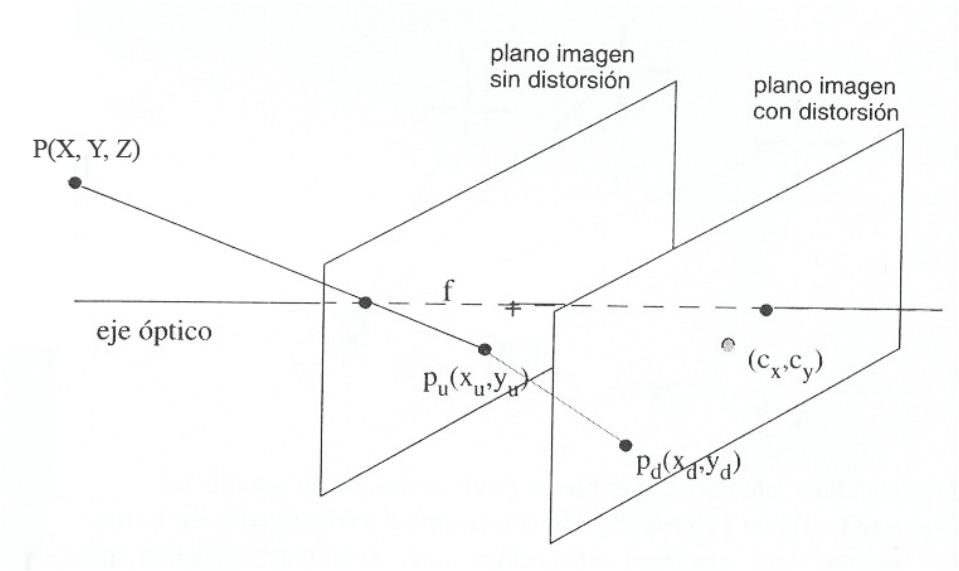


Figura 31. Modelo de cámara con distorsión radial.

Este es el modelo de cámara utilizado en este proyecto. Dado un modelo de cámara es posible efectuar un proceso de calibración de cámara que proporcionará la matriz de parámetros intrínsecos para esa cámara. Este proceso solo se tendrá que efectuar una vez y se explicará en el punto 3.2.4.

3.2.3 Estimación de pose y orientación

La estimación de pose y orientación consiste en reconstruir la posición de la cámara, i.e. el punto de vista del observador, a raíz de la imagen capturada por la misma cámara de un patrón conocido. Dada una situación como la que muestra la Figura 32 donde tenemos una cámara y un patrón que es proyectado en el plano imagen de la cámara, la estimación de la pose y la orientación determina la posición de la cámara con respecto al sistema de coordenadas del patrón.

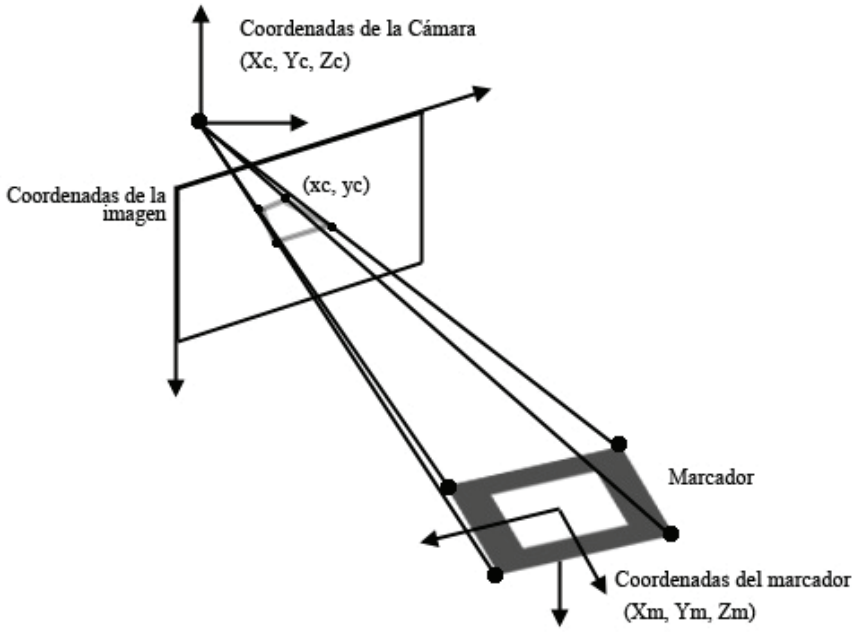


Figura 32. La relación entre coordenadas del modelo y coordenadas de la cámara.

Este proceso es importante a la hora de mostrar un objeto 3D superpuesto en la pantalla de la aplicación. Para que un objeto 3D se muestre de manera que parezca que está siendo visto desde la cámara modelada, es necesario conocer exactamente la posición de la misma desde el sistema de coordenadas de referencia, el del patrón. Esta relación entre los dos sistemas de coordenadas viene dado por una rotación **R** y una traslación **t**, los parámetros extrínsecos.

Teniendo un conjunto de puntos X_i conocidos del patrón y un conjunto de puntos correspondientes x_i pertenecientes a la imagen tomada de ese patrón, la transformación proyectiva viene dada, como ya se ha explicado en la sección 3.2.2.3, por la ecuación $\mathbf{x_i = H X_i}$ donde

$$\mathbf{H} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix}$$

Si reescribimos la ecuación $\mathbf{x_i = H X_i}$ observando (6) obtendremos:

$$\hat{\mathbf{x}}_i = K[\mathbf{R} \quad | \quad \mathbf{t}]\mathbf{X}_i$$

Donde K es la matriz de parámetros intrínsecos conocida puesto que es obtenida al calibrar la cámara. El paso de calibración de cámara es un paso previo necesario a la reconstrucción de la pose y orientación. Por lo tanto los únicos elementos no conocidos son la rotación **R** = (r_1^T, r_2^T, r_3^T**)** y la translación **t,** los parámetros extrínsecos**.**

El método de obtención de los parámetros extrínsecos que se explica en este documento es el propuesto por Zhang [Zha99] como sigue:

Algoritmo

0. Obtener **H** homografía entre el modelo y la imagen, basándonos en el método de Zhang aprovechando el conocimiento de la cámara.

1. Sabemos que $\lambda = 1/\left\|\mathrm{K}^{-1}\mathrm{h}_1\right\| = 1/\left\|\mathrm{K}^{-1}\mathrm{h}_2\right\|$

2.
$$\mathrm{r}_1 = \lambda \mathrm{K}^{-1}\mathrm{h}_1$$

$$\mathrm{r}_2 = \lambda A^{-1}\mathrm{h}_2$$
$$\mathrm{r}_3 = \mathrm{r}_1 \times \mathrm{r}_2$$
$$t = \lambda \mathrm{K}^{-1}\mathrm{h}_3$$

3. Obtenemos la matriz de rotación y el vector de traslación (solución analítica) como sigue

$$\left[\mathbf{R} \;\middle|\; \mathbf{t}\right] \equiv \left[\mathrm{r}_1^{\,T} \quad \mathrm{r}_2^{\,T} \quad \mathrm{r}_3^{\,T} \;\middle|\; \mathbf{t}\right]$$

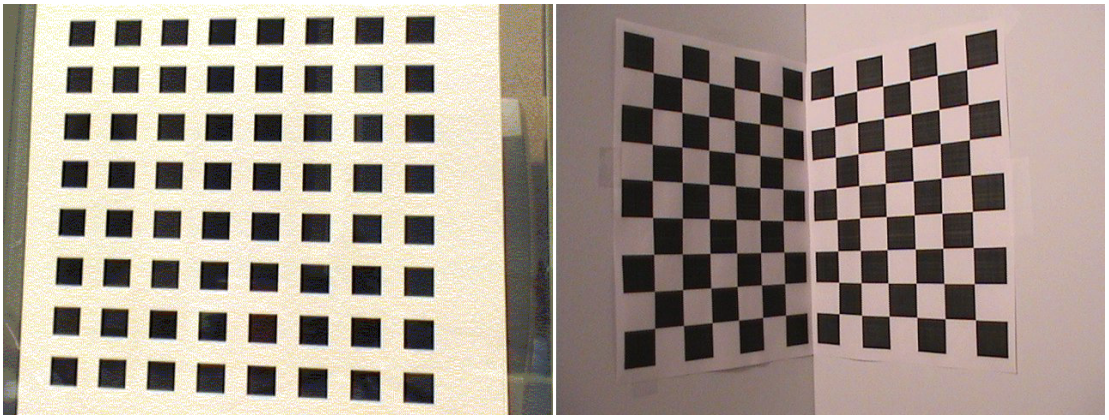
4. Refinamiento de los parámetros extrínsecos. Este refinamiento es un problema de minimización de la distancia entre los puntos conocidos en la imagen x_i y la proyección de cada punto X_i en la imagen, $\hat{x}_i$.

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|\mathrm{x}_{ij} - \hat{\mathrm{x}}\left(\mathrm{K}, \mathbf{R}_i, \mathbf{t}_i, \mathrm{X}_j\right)\right\|^2$$

Esta minimización es un problema de minimización no lineal que puede ser resuelto por el algoritmo de Levenberg-Marquardt, véase el apéndice B.

3.2.4 Calibración de la cámara

Se entiende como calibración de la cámara, el proceso de determinar las características ópticas y geométricas internas de la cámara, es decir, los parámetros intrínsecos. Para poder estimar estos parámetros intrínsecos es necesario obtener un conjunto de puntos de los cuales sepamos las correspondencias entre el plano imagen y el mundo. El objetivo de este proceso es alimentar los métodos de optimización que estiman los parámetros como el de Levenberg-Marquardt habiendo establecido un modelo de cámara. En el presente caso, el modelo de cámara que se utiliza en la calibración ha sido explicado en la sección 3.2.2.3, por lo que es necesario obtener los puntos y sus correspondencias de alguna manera para poder realizar la calibración.



(a) (b)

Figura 33. Patrones de calibración

Existen varias formas de obtener estos puntos y sus correspondencias. Por ejemplo, Zhang [Zha99], utiliza una matriz de cuadrados negros sobre fondo blanco para este objetivo (ver Figura 33 (a)), en cambio otras propuestas pueden emplear configuraciones más sofisticadas como en el caso del mostrado en la Figura 33 (b), utilizando dos planos ortogonales con el mismo patrón impreso en ambos. Para este proyecto hemos utilizado un único patrón plano donde ha sido impreso un array de marcadores de la librería. La creación de este patrón se ha realizado a través de una aplicación (Figura 34) a través de la cual es posible configurar la rejilla de calibración y ordenar los marcadores y sus identificadores dentro de la misma.

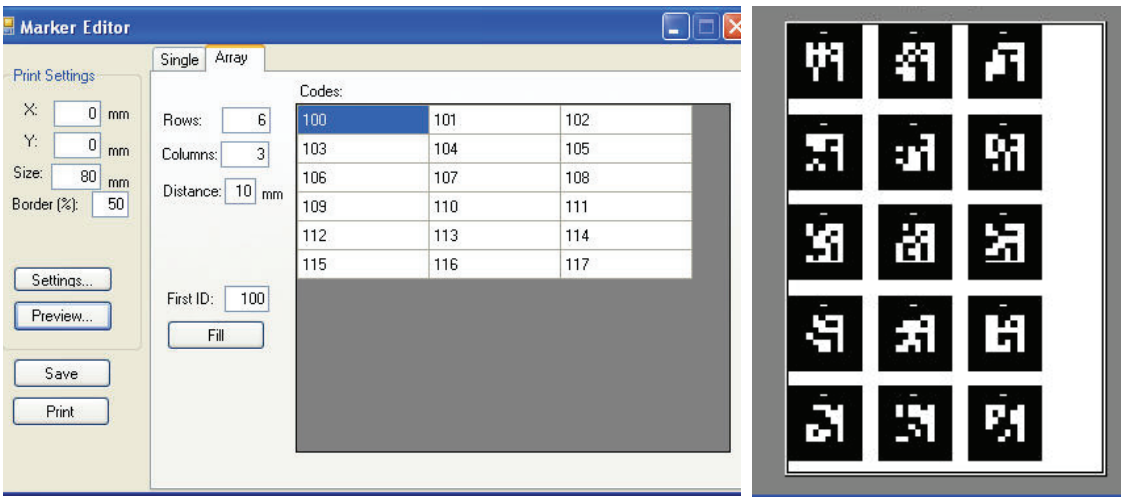


Figura 34. Aplicación para crear la rejilla de calibración.

El motivo de utilizar este tipo de rejilla para la calibración es el de aprovechar los algoritmos de detección de marcadores que ya están implementados en el sistema, para obtener las correspondencias entre los puntos de la imagen y el modelo de rejilla, el patrón de calibración. Además también aprovecha la implementación de algoritmos de identificación de marcadores para poder establecer la correspondencia entre los marcadores

del patrón de calibración y los marcadores de la imagen de manera inequívoca.

Para poder establecer esa correspondencia y para posteriores cálculos también es necesario conocer cuál es la posición de los vértices de los marcadores en la rejilla de calibración. En el momento de crear el patrón de calibración se especifican ciertos parámetros como el número de marcadores, cuál es su tamaño o cuál es la separación entre marcadores (véase la Figura 35). Con esta información es posible calcular la posición exacta de los puntos clave, es decir, las esquinas de los marcadores, en el modelo de calibración.

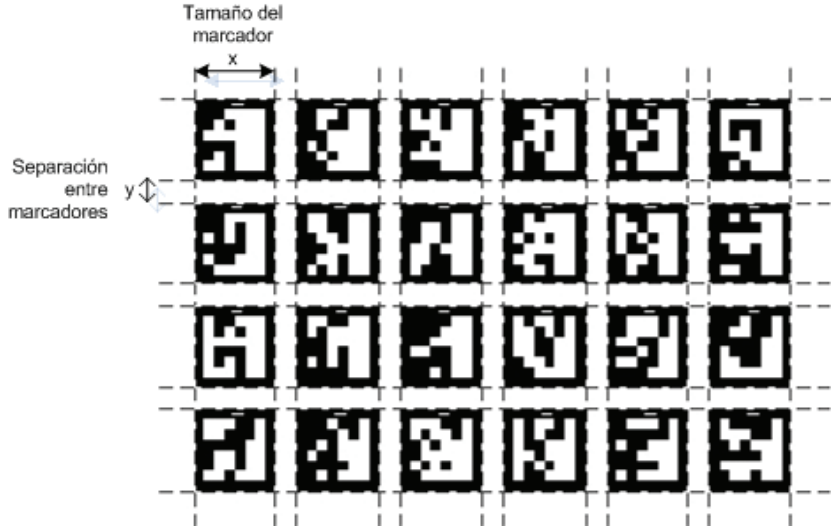


Figura 35. Patrón de calibración.

Una vez se dispone de las correspondencias de puntos entre el plano imagen y el mundo y conociendo el modelo de cámara, la estimación de los parámetros consiste en minimizar mediante Levenberg-Marquardt la siguiente función objetivo:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|\mathrm{x}_{\mathrm{ij}} - \hat{\mathrm{x}}(\mathrm{K}, k_1, k_2, \mathbf{R}_i, \mathbf{t}_i, \mathrm{x}_{\mathrm{j}})\right\|^2 \qquad (7)$$

La técnica utilizada es en esencia la misma que la utilizada por Zhang en su propuesta con la diferencia de utilizar un patrón de marcadores en lugar de un patrón basado sólo en cuadrados completamente negros. Ésta requiere, como ya se ha explicado, que la cámara capte un patrón plano desde varias orientaciones diferentes. Para tomar las diferentes imágenes es posible mover tanto la cámara como el patrón a mano. La Figura 36 muestra algunas de las imágenes captadas para la calibración de la cámara.

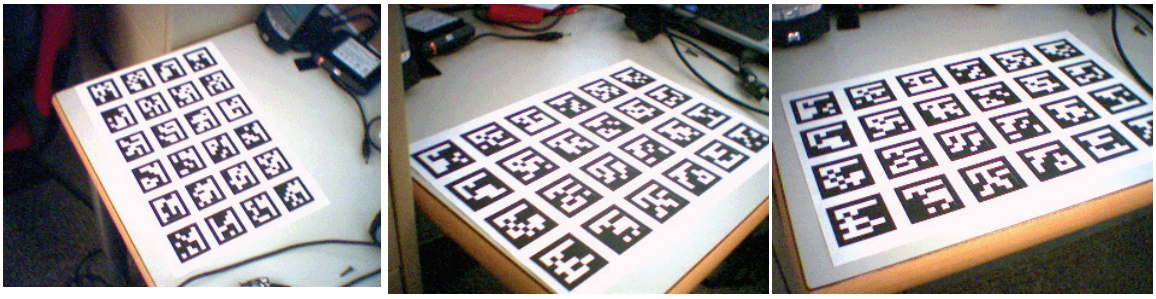


Figura 36. Imágenes tomadas para la calibración de la cámara.

A modo de resumen el proceso de calibración es el siguiente:

1. Imprimir la rejilla de marcadores preparada para la calibración y pegarla en una superficie plana.

2. Tomar varias imágenes de la plantilla de calibración desde diferentes orientaciones moviendo la cámara o la plantilla.

3. Detectar los puntos clave en las imágenes haciendo uso de los algoritmos de detección e identificación de marcadores

4. Estimar los parámetros intrínsecos

5. Refinar los parámetros minimizando la expresión (7)

3.3 Implementación

El hecho de pretender que el subsistema de seguimiento pudiera ser utilizado en diferentes tipos de aplicaciones de realidad aumentada impone la necesidad de mantener el núcleo del subsistema lo más portable posible. Tengamos en cuenta que las restricciones impuestas por las características de los equipos de destino eran en principio muy variadas aunque siempre exigiendo ciertos requisitos mínimos para no comprometer la viabilidad sobre PCs y PDAs. Para la implementación se plantean dos posibilidades en cuanto a lenguajes de programación se refiere. Se puede utilizar lenguajes basados en máquina de virtual como Java o los llamados gestionados como C#.Net, o bien un lenguaje nativo como C/C++. Los lenguajes gestionados tienen la ventaja de que están pensados para la fácil y rápida distribución en plataformas de diferente naturaleza. Sin embargo, tienen el inconveniente de que en el caso de aplicaciones con requisitos de rendimiento y prestaciones elevadas, como es nuestro caso, supone un problema debido principalmente a la gestión automática de la memoria y a la ejecución sobre una máquina virtual. Por ello, tras realizar algunas pruebas de rendimiento se decidió utilizar el lenguaje de programación C/C++. Además de proporcionarnos mayor rendimiento, en la práctica nos permite ser algo más portables ya que no tenemos dependencias de librerías

externas específicas y ello nos permite movernos hacia otras plataformas y sistemas operativos.

En la Figura 37 se muestran las clases y los métodos más relevantes del subsistema en un diagrama de clases. La clase *AppClient* representa la clase cliente del sistema. Ésta hace uso de los métodos y clases del sistema de detección de marcadores para implementar el sistema completo de Realidad Aumentada. Esa parte podría ser ampliada o reescrita por terceras partes según la utilidad de la aplicación final. Por lo tanto, el objetivo del sistema completo depende de qué uso le dé a la clase cliente que use el sistema de detección basado en marcadores. En nuestro caso, ya hemos realizado aplicaciones de de Realidad Aumentada sobre PCs y PDAs, y quedaría utilizarlo para nuestro propósito final de seguimiento en una *superficie interactiva*.

Otro elemento importante es la clase *CVideo*. Esta clase da soporte a la utilización de la cámara desde la aplicación. En realidad, un hardware como la cámara es muy dependiente del equipo, y ello conduce irremediablemente a que ante la utilización de nuevas cámaras se tenga que reprogramar esta clase para dar soporte al nuevo dispositivo.

El resto del diagrama de la Figura 37 representa las clases y relaciones entre clases del sistema de detección basado en marcadores. Por un lado encontramos *CMarkerDetector*. Esta clase es la encargada de analizar las imágenes que *AppClient* le proporciona en busca de la presencia de marcadores. Para las tareas de detección e identificación utiliza, por un lado la clase *CMarkerLibrary* y por otro lado *CMath*. *CMarkerLibrary* proporciona datos del modelo de marcador necesarios en cálculos de homografías y métodos de corrección de errores a la hora de identificar los identificadores del interior de los marcadores encontrados. Por otro lado la clase *CMath* proporciona métodos estáticos para el cálculo de operaciones matemáticas complejas como la descomposición LUD, SVD o métodos como Levenberg-Marquard. Finalmente *CMarkerLibrary* crea tantas instancias de *CMarker* como marcadores ha detectado en la imagen almacenando en estas instancias la información relevante obtenida para cada marcador.

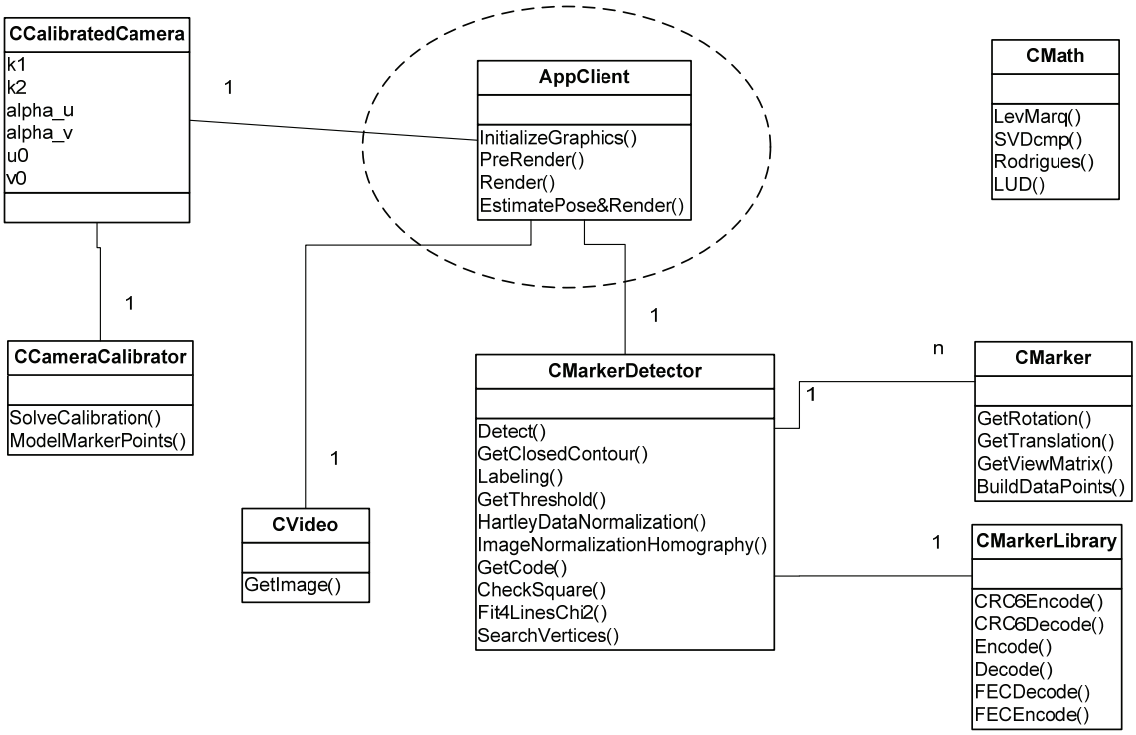


Figura 37. Diagrama de clases del sistema de marcadores

Finalmente encontramos *CCalibratedCamera* y *CCameraCalibrator*. La función de estas clases es proporcionar un método de calibración de la cámara utilizada. Una vez la cámara ha sido calibrada, los parámetros intrínsecos y otras características de la cámara son almacenadas en *CCalibratedCamera*.

A modo de resumen, un sistema que utilice el subsistema de marcadores puede dividirse en las siguientes partes. Por un lado la parte que gestiona la captura de imágenes y manipulación de la cámara, otra parte que se encarga de la detección e identificación de los marcadores y otra parte que sirve para la necesaria calibración de la cámara. Toda esta funcionalidad es utilizada por la clase *AppClient* para que el sistema funcione con una finalidad concreta.

3.4 Conclusiones

Este capítulo ha presentado el subsistema de seguimiento de marcadores que permite el etiquetado de objetos tangibles y el seguimiento de los mismos sobre la superficie interactiva.

Dado que este problema es habitual en aplicaciones de Realidad Aumentada, el desarrollo de este subsistema se ha enfocado al desarrollo de

un sistema de visión por computador de Realidad Aumentada basado en la detección de marcadores. De esta forma, además de cubrir el seguimiento de objetos tangibles en las superficies interactivas, se dispone de un sistema de seguimiento general que ha sido utilizado satisfactoriamente en aplicaciones de *RA* de diversa naturaleza para evaluar su correcto funcionamiento, tanto en dispositivos móviles como en configuraciones basadas en gafas de visualización.

El presente capítulo ha mostrado con cierto detalle las fases de procesamiento y los principales algoritmos implicados tanto en la detección como en la reconstrucción 3D requerida para hacer efectivo el seguimiento coherente que nos permita visualizar información acordemente.

Evidentemente aún queda su utilización en la plataforma de destino que supone la superficie interactiva, lo que hace probable que algún algoritmo deba ser sintonizado y/o mejorado. Otro asunto pendiente es el de extender la API del subsistema o incluso la creación de una capa de abstracción sobre ella para soportar fácilmente su uso para propósitos de *interacción*.

CAPÍTULO IV

PLATAFORMA SEMÁNTICA PARA LA GESTIÓN DE EVENTOS

En este capítulo se presenta la infraestructura software que en última instancia facilite la creación y animación de ecosistemas reactivos que se pretende para nuestro objetivo final de soportar determinadas actividades de aprendizaje creativo sobre *superficies interactivas*. Esta infraestructura software ha sido abordada desde una perspectiva general de los sistemas de distribución de eventos que permita ser flexible para la definición de los tipos y la forma de los eventos. En el ámbito de aplicación en el que estamos es necesario pensar en sistemas de contenidos (*content-based*) en los que la descripción de eventos y las suscripciones son mucho más ricas que en los basados en categorías (*topic-based*). Además, el subsistema ha sido considerado desde una perspectiva semántica, lo que significa que el proceso para determinar qué eventos cumplen con qué suscripciones (*matching process*) no se basa exclusivamente en aspectos sintácticos sino en información ontológica que eventualmente se define para caracterizar el tipo y la forma de los eventos. En particular, la implementación se basa en el lenguaje de ontologías para la web, OWL, que está siendo propuesto por el W3C (*World Wide Web Consortium*) para la Web Semántica. Esta decisión nos permite aprovechar los razonadores existentes que trabajan sobre la lógica subyacente a OWL para implementar el proceso de evaluación de correspondencias entre eventos y suscripciones. Por tanto la infraestructura soporta la creación de ontologías de interés que definen los tipos de eventos, así como también las relaciones semánticas entre conceptos descritos en diferentes ontologías. El envío de eventos de acuerdo a alguna ontología registrada en la infraestructura deriva en un proceso de razonamiento que determina qué suscripciones deben recibir determinados eventos, para finalmente ser remitidos a los suscriptores.

El capítulo tiene la siguiente estructura. En primer lugar se presenta un modelo básico de *ecosistema reactivo*, haciendo énfasis en el concepto de entidad, la definición de eventos y el comportamiento que las entidades deben exhibir en respuesta a la ocurrencia de los eventos que será tomado

como base para el desarrollo de la plataforma final sobre superficies interactivas. A continuación se introducen los conceptos relativos a los sistemas de distribución de eventos que nos permitirán definir el nuestro. También se introduce el uso de ontologías en la *Web Semántica* y la lógica subyacente, *Lógicas Descriptivas*. Entonces se presentará la plataforma semántica de eventos que se ha desarrollado e implementado, para finalmente presentar el subsistema de simulación que basado en la plataforma semántica se pretende para facilitar la simulación del ecosistema en tiempo de juego.

4.1 Ecosistemas Reactivos

En este primer punto del capítulo, y antes de entrar en detalle en la plataforma de eventos, se presenta nuestra primera definición de ecosistema reactivo que en cierta medida motive el uso del subsistema de eventos semántico, y que permita en nuestro trabajo futuro trabajar sobre él, extendiéndolo para poder soportar futura funcionalidad que pudiera surgir durante el desarrollo efectivo de la infraestructura para juegos creativos sobre superficies interactivas. Hay que recordar que, como se dijo en el capítulo 2, el objetivo último es dar soporte a la creación como una actividad más dentro del *juego*, y por lo tanto se deberá dar soporte tanto a la autoría de ecosistemas reactivos como a su posterior simulación.

4.1.1 Entidades reactivas básicas

En muchos asuntos concernientes a la educación escolar y la instrucción, el profesor es el responsable de idear escenarios de aprendizaje consistentes que encuentren los conceptos que deban ser transmitidos y las habilidades a ser desarrolladas. Tras ese primer paso de concepción, el profesor tiene que presentar a los estudiantes el entorno básico en el que quiere que la acción acontezca y las tareas que deben abordar a través de la definición de algunos actores que actúan en dicho escenario. Esta perspectiva se centra en las teorías de aprendizaje presentadas en el capítulo 2 que defienden el aprendizaje por medio de la acción, de la interacción con artefactos, objetos, entidades y/o el entorno. Por tanto debemos acercarnos al concepto de ecosistema, en el que residen entidades interactivas, basadas en un modelo simple de relaciones acción-reacción y causa-efecto, que permitan exhibir comportamiento, que interaccionen entre ellas y respondan a las acciones de los participantes.

Un entorno reactivo, también conocido en nuestro contexto como ecosistema, es el lugar donde varias entidades habitan, interactuando entre ellas, y llevando a cabo sus acciones con el propósito de alcanzar sus objetivos. El profesor les dice a los alumnos qué tareas tienen que acometer y cada uno de

ellos es responsable del diseño de una entidad específica en el ecosistema. Estas entidades son especiales, en el sentido de que son ciudadanos de primer orden porque representan, hasta cierto punto, la personificación del rol que los estudiantes deben asumir, explorar, y además estas entidades les permitirán manipular el entorno a través de sus acciones. Las entidades ordinarias, o al menos las más modestas, tales como objetos y elementos del escenario, o aquellas que no están bajo la responsabilidad de estudiantes son también importantes ya que son necesarias para construir escenarios razonables y porque también pueden presentar comportamiento reactivo. La Figura 38 muestra una parte de nuestro modelo de entidades reactivas como un diagrama de clases UML, para la autoría de entidades reactivas que cumplan estos requisitos así como algunos de sus atributos relevantes.

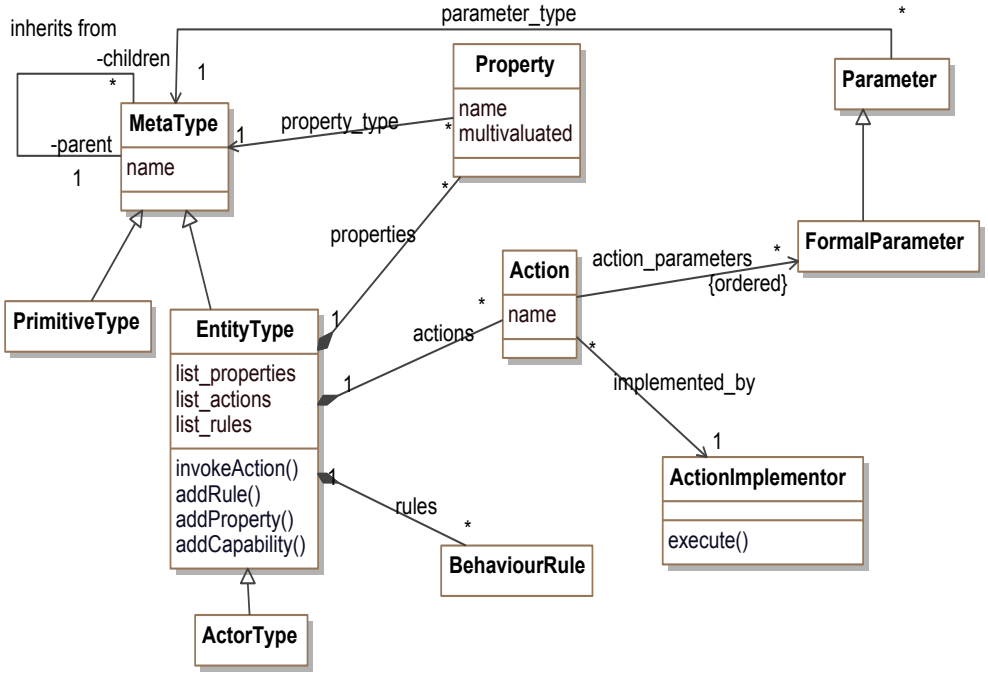


Figura 38. Diagrama UML de clases para la definición de tipos de entidad

La clase *EntityType* soporta la definición de entidades reactivas y es la clase más importante de esta parte. Cada *EntityType* consiste de un conjunto de propiedades, un conjunto de acciones, y por último, un conjunto de reglas reactivas que expresan el comportamiento necesario para definir reacciones que las entidades deben exhibir. Para diferenciar definiciones de *entidades-ordinarias* de las de *entidades-actores* se ha especializado el concepto en la clase *ActorType*. En esencia las *entidades–ordinarias* y las *entidades-actores* a nivel de modelo estático son idénticas en su definición en estos momentos. La diferencia radica en lo conceptual ya que se pretende que las entidades que responden a acciones de usuario sean las *entidades-actores*. Además se intuye la necesidad que surgirá de explosionar los tipos de entidad dando lugar a una amplia y profunda jerarquía, desde el momento en el que se

determinen precisamente las posibilidades del motor de simulación que deba animar todo el ecosistema.

Las propiedades (*Property)* son como las cualidades o atributos que las entidades tienen. Tienen un nombre (*name*) y un tipo (*type*), que puede ser, o bien un tipo de entidad, o bien un tipo de datos primitivo. Por esta razón, y dado que estamos modelando tipos, también se provee de una clase *PrimitiveType* para soportar tipos básicos predefinidos, y la clase *MetaType* que generaliza ambas categorías de tipos.

Las acciones (*Action*) pueden entenderse como el comportamiento activo que las entidades pueden exhibir. Por tanto, una acción es como una operación que la entidad puede realizar. La clase *Action* tiene un nombre (*name*), y un conjunto ordenado de parámetros. Además, un parámetro se caracteriza por un tipo de los que se están definiendo. Más a allá de la definición de la acción, se necesita proveer de un mecanismo que permita la especificación de lo que realmente la acción hace, y para este propósito se podría adoptar una amplia variedad de lenguajes de especificación, bien gráficos, de scripting, o sencillamente un librería de terceros que aporte una implementación a la acción. Este requisito está abstraído en el modelo por medio de la clase *ActionImplementor*. Las acciones pueden ser invocadas de tres formas diferentes: desde otra acción, desde una regla, o incluso por lo participantes a través de la manipulación de las entidades que estén bajo su responsabilidad en el momento de jugar.

Las reglas (*BehaviourRule*) cubren el comportamiento reactivo que las entidades pueden exhibir. Actualmente, estamos considerando el uso de reglas similares a las reglas ECA (*Event-Condition-Action*). Una *BehaviourRule* se compone de un antecedente y de un consecuente. El antecedente es una condición general que incluye la descripción de un evento y ciertas condiciones sobre sus parámetros. El consecuente es una descripción de la invocación a una acción.

Adoptando el uso de un entorno reactivo y la introducción de la fase de diseño como una importante actividad además de sólo jugar, algunos de los principios de Gee se adoptan casi de forma inmediata [Gee05]. Ese trabajo presenta algunos principios considerados como buenos para lo videojuegos que quieran ser entendidos para entornos de aprendizaje. Algunos de ellos se presentan de forma inherente en nuestra aproximación basada en ecosistemas reactivos, especialmente aquellos relacionados a los categorizados como "Aprendices Motivados" (*Empowered learners)* y "Comprensión" (*Understanding)*, mientras que la ocurrencia de otros principios dependen de cómo los profesores diseñen las actividades, los escenarios, y las tareas a realizar. El profesor, mediante la asignación de responsabilidades a los alumnos de crear entidades reactivas, definición de sus propiedades, acciones, y reglas, les fuerza a estudiar los contenidos curriculares relacionados al escenario planteado, y a mantener una

participación activa. Además, de esta manera, se da el principio de "Co-diseño" (*Co-Design)* dado que los alumnos están también diseñando la interactividad y las reacciones que guiarán la evolución del ecosistema. De forma similar, el principio de "Identidad" (*Identity*) se cubre doblemente, ya que los alumnos tienen que aprender el rol de la entidad a través de su diseño, pero también por medio del juego posterior. Las entidades pueden verse como componentes sencillas que sólo pueden ser completamente entendidas como parte de un sistemas complejo donde todo cobra significado, tal y como el principio de "Pensamiento como sistema" (*System thinking*) describe. En nuestro caso, el sistema complejo es el ecosistema, y esto es, a la vez, correlacionado a alguna realidad física que el profesor ha decidido implicar en el escenario. En este sentido, el escenario preparado es un modelo simplificado de la realidad, y cuando se juega, las consecuencias se simulan con un bajo riesgo soportando de esta forma los principios de "Peceras" (*Fish tanks)* y "Cajas de arena" (*Sandboxes*).

4.1.2 Eventos y reacciones

Hasta el momento no se ha hablado en detalle de las reglas y los eventos. Se procede pues a describir la parte del modelo relacionado con los eventos y las reglas de comportamiento (véase la Figura 39). Los eventos son una parte esencial de todo sistema reactivo porque comunican los cambios relevantes en el estado de las entidades. En particular, los eventos son lanzados por las acciones cuando éstas se ejecutan. En nuestro modelo, la clase *EventType* permite la definición de tipos de eventos, teniendo un nombre (*name*) y un conjunto ordenado de parámetros.

Como se dijo anteriormente, una regla de comportamiento se compone de un antecedente y de un consecuente. Específicamente el antecedente es lo que hemos llamado *patrón de eventos*, *EventPattern*. Un patrón de eventos es una construcción flexible utilizada para especificar ocurrencias genéricas de de eventos. Un patrón de eventos tiene un tipo de evento y un conjunto ordenado de argumentos. El tipo de evento limita los posibles argumentos reales que cumplan con su definición de parámetros, i.e. el tipo de cada argumento y su posición válida en la lista de argumentos. Un argumento o parámetro real es o bien un parámetro instanciado o una variable. Un parámetro instanciado tiene una referencia a una entidad o a un valor de un tipo primitivo. Una variable es como un comodín que toma valores en el momento de darse la correspondencia entre una ocurrencia de un evento y un patrón de evento. Además, se puede expresar las condiciones sobre los argumentos.

Por otra parte, el consecuente es un *patrón de acción*. Un patrón de acción es similar a un patrón de evento pero referido a una acción que debe ser tomada en caso de que el antecedente sea satisfecho por una ocurrencia de un evento. Las variables implicadas en el consecuente tienen que aparecer

en el patrón de evento del antecedente tal que los valores tomados por las variables en el antecedente se propagan a las variables correspondientes del consecuente.

Personas diferentes perciben la realidad de forma diferente, y consecuentemente la gente piensa y aprende también de forma diferente. Los individuos, y consecuentemente las entidades, tenemos un punto de vista subjetivo, y como resultado, internalizamos los conceptos observados en términos de otros que hemos experimentado previamente. Esto nos lleva a cada uno de nosotros a construir nuestra propia *malla de conocimientos* de acuerdo a nuestras propias experiencias y pensamientos, tal y como el principio de Gee que habla sobre el "Significado como imagen de la acción" (*Meaning as action image).* Esto significa que se necesita dar soporte a la definición subjetiva de eventos ocurriendo en el ecosistema y la posibilidad para definirlos en términos de otros eventos para los que los participantes ya conocen su significado exacto. También se está considerando esto en nuestro modelo al permitir la definición de ciertas relaciones de equivalencia entre tipos de evento. La clase *EntityType* tiene un conjunto de correspondencias entre los tipos de evento. La clase *EventTypeMapping* establece una relación de subsunción entre dos tipos de eventos, y la clase *ParameterMapping* refina la conversión entre sus parámetros. El establecimiento de este tipo de relaciones es muy importante porque de esta forma los participantes podrán personalizar la percepción de sus entidades, permitiéndoles indicar que se si sucede un determinado evento expresado según el vocabulario u ontología de una entidad entonces es como si estuviera sucediendo un evento conocido en la ontología de otra entidad. Por ejemplo, dos entidades diferentes podrían entender de distinta manera lo que es un "evento positivo" para sus respectivos objetivos e intereses, y podrían expresar dicha percepción subjetiva por medio de la definición de sus relaciones de correspondencia entre términos de otros eventos definidos internamente. Para poder soportar fácilmente esta característica en tiempo de ejecución, se considera un subsistema de eventos semántico genérico que se presenta en el resto del capítulo.

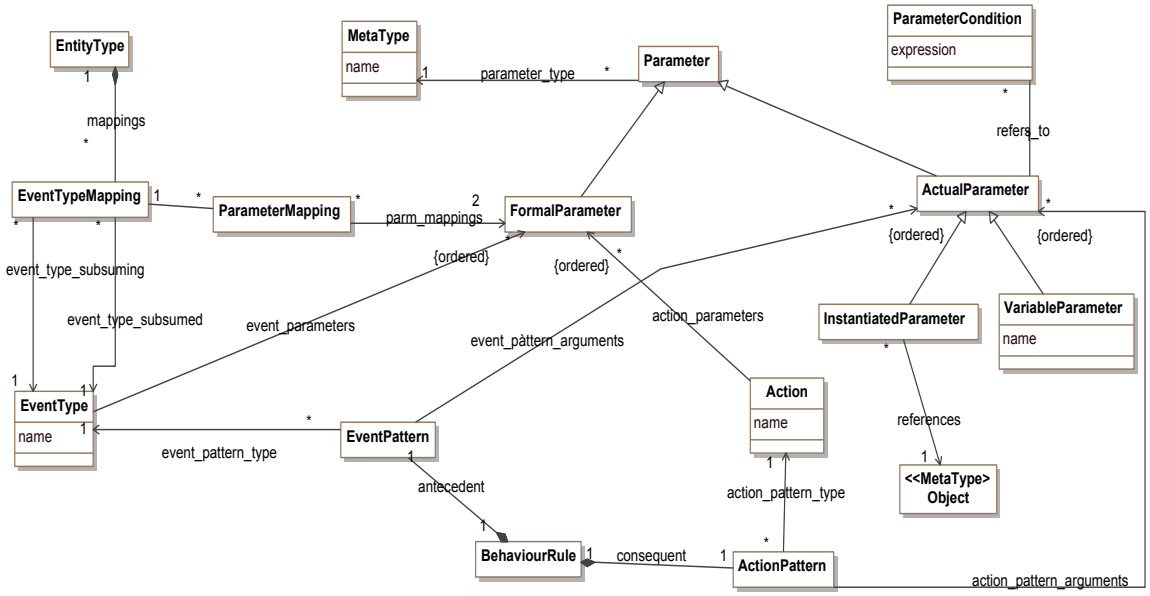


Figura 39. Diagrama de clases UML para eventos y reglas de comportamiento.

4.2 Sistemas de Publicación/Suscripción

Cuando se está interesado en la distribución de determinadas piezas de información de acuerdo a ciertos criterios específicos de filtrado y con cierta estructura se suele abordar tal distribución desde la perspectiva de los sistemas de publicación/suscripción de eventos.

Los sistemas de publicación/suscripción de eventos se caracterizan por su funcionamiento dinámico y desacoplado dado que en general siguen esquemas de entrega de eventos asíncronos con miras a ofrecer una mayor escalabilidad. Informalmente, un sistema de publicación/suscripción es un paradigma que provee de una forma de interacción débilmente acoplada entre productores y consumidores de información [Eug03][Liu03]. Los productores, conocidos comúnmente como publicadores (*publishers*), publican ciertas piezas de información llamados eventos (*events*) en el sistema de publicación/suscripción. Los consumidores, conocidos como suscriptores (*subscriber*), son capaces de expresar sus intereses sobre los eventos. Las peticiones que expresan esos intereses son conocidas como suscripciones (*subscription*). En cualquier momento que un publicador publica un evento, el evento es notificado a los suscriptores que tienen alguna suscripción que se ajusta al evento. Por tanto, el modelo de publicación/suscripción ofrece en esencia un servicio para las aplicaciones y/o componentes de las mismas que lo utilizan para el almacenamiento, la gestión de suscripciones, y la entrega asíncrona de los eventos publicados sin necesidad de que los publicadores y los suscriptores se conozcan.

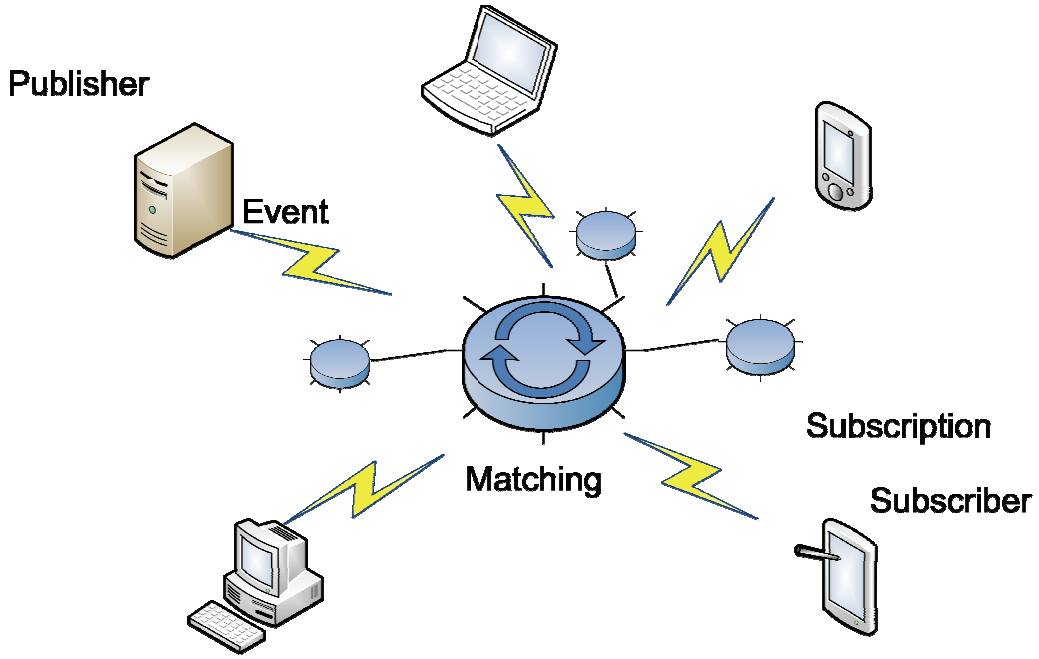


Figura 40. Descripción general de sistemas de publicación/suscripción

La Figura 40 plasma de forma generalista un sistema de publicación/suscripción de eventos. La infraestructura estaría compuesta por uno o más nodos de computación que ejercen ciertas funciones a modo de servidor, manteniendo y gestionando las suscripciones. El funcionamiento general sería como sigue. Los suscriptores envían sus intereses de recibir información en forma de suscripciones de acuerdo a algún lenguaje de suscripción soportado por alguno de los servidores de la infraestructura. Los publicadores envían eventos hacía la infraestructura según el modelo de eventos subyacente. El servidor que recibe el evento lo evalúa confrontándolo contra las suscripciones y entonces distribuye el evento a los suscriptores correspondientes. Este proceso puede ser muy complejo en función de la topología de la infraestructura, el modelo de eventos y los lenguajes de suscripción. De este modo, lo que caracteriza a los sistemas de publicación/suscripción es 1) cómo los suscriptores expresan sus suscripciones, 2) cómo los eventos se cotejan contra las suscripciones, y finalmente 3) cómo los eventos son efectivamente encaminados a los suscriptores.

Con respecto a cómo los suscriptores expresan sus suscripciones, los sistemas de publicación/suscripción se categorizan típicamente en dos amplios grupos: los basados en categorías (*topic/subject-based*) [Row01][Zhu01][Bal07], y los basados en contenido (*content-based*) [Car00][Pie02][Zhu07]. El esquema basado en categorías es la aproximación más sencilla y simple, y consiste esencialmente en anotar los eventos con una cadena o una etiqueta que represente una categoría o tema (*topic/subject*) disponible en el conjunto de categorías permitidas en el sistema. En esta aproximación los suscriptores tienen que registrarse en

aquellas categorías en las que estén interesados, y por tanto las suscripciones consisten en una lista de categorías de interés. Tiene la ventaja de que la información a transmitir puede ser todo lo compleja que se quiera ya que el sistema de publicación sólo requiere trabajar con las etiquetas que acompañan a dicha información, pero generalmente una lista de etiquetas no suele ser bastante expresivo para la gran mayoría de las aplicaciones ya que podría no definir de forma precisa el evento de interés.

Por el contrario, un esquema basado en contenido provee métodos de suscripción más expresivos y complejos basados en el contenido real de los eventos. Por tanto, la información que se transmita va a ser analizada en cierta manera para determinar a quién debe ser entregada, y por tanto la complejidad de la información que se quiera soportar, que es en sí misma la que determina la forma del evento, afecta a cómo deben ser expresadas las suscripciones. Generalmente en este esquema se busca un compromiso entre la facilidad de uso y la expresividad, tomando los eventos usualmente como un conjunto de *pares propiedad-valor*, y las suscripciones consistentes en una conjunción de tripletas de la forma <*propiedad, operador, valor*>.

Respecto al proceso de cómo las suscripciones se cotejan frente a los eventos que llegan (*matching process*), éste viene determinado en gran medida por el tipo de suscripciones que se soportan. En el caso de sistemas de publicación/suscripción basados en categorías el proceso sencillamente consiste en inspeccionar las etiquetas de las suscripciones y las etiquetas que categorizan a los eventos. Sin embargo, en los sistemas basados en contenido el proceso puede llegar a ser extremadamente complejo dependiendo de la expresividad dotada al lenguaje de suscripciones ya que consiste en evaluar el contenido del propio evento. En un modelo basado en contenidos como el mostrado anteriormente basado en eventos como pares propiedad-valor y suscripciones como tripletas en las que hay implicados ciertos operadores, el *proceso de matching* consistiría en averiguar qué suscripciones cumplen con cada uno de los eventos de acuerdo a sus especificaciones. El asunto se complica más todavía cuando los operadores consideradores son también complejos, como es el caso de operadores temporales. La mayoría de los sistemas se basan en un proceso de comparación puramente sintáctico. Esto significa que la forma de los eventos suele ser completamente cerrada, y en el caso de que exista la posibilidad de definir las plantillas de los eventos éstos deben ser totalmente conocidos por todos los suscriptores, así como el significado de cada propiedad universalmente establecida. En este caso el tratamiento de la heterogeneidad de sistemas se convierte en una tarea muy complicada teniendo en cuenta que en potencia cualquiera de las aplicaciones y/o sistemas que hacen uso del servicio de publicación/suscripción podrían ser de naturaleza muy diversa y por lo tanto deberían conocer todo el vocabulario del resto de los participantes en el sistema de distribución de eventos. Una alternativa a las aproximaciones sintácticas para tratar con la

heterogeneidad es el dotar de información semántica al evento o bien expresar semánticamente el evento en sí mismo en algún formalismo adecuado. Esto permite aumentar, aún más si cabe, la potencia de los lenguajes de suscripción y las posibilidades de los sistemas de publicación/suscripción para tratar con la heterogeneidad aunque ello supone que el proceso de comprobación de suscripciones a eventos se vuelve mucho más complejo y la escalabilidad se compromete en gran medida. Cuando el proceso de evaluar las correspondencias entre eventos y suscripciones ya no sólo supone la evaluación de ciertos valores para ciertas propiedades sino que implican también relaciones semánticas que deben ser inspeccionadas, el proceso se vuelve más duro computacionalmente hablando.

En cuanto a cómo los eventos son encaminados a los suscriptores (en función de la topología), existen varias aproximaciones típicas. La más básica es la completamente centralizada en la que un simple "intermediario" (*broker*) se encarga de mantener todas las suscripciones, recibir todos los eventos, realizar el *proceso de matching* y finalmente entregar todos los eventos a los suscriptores correspondientes. Esta infraestructura no es en sí misma distribuida, y por lo tanto sufre en general de sobrecarga en el procesamiento de las correspondencias de eventos a suscripciones, y en la entrega de los eventos propiamente dicha. Otra aproximación es la distribuida basada en varios intermediarios. En esta aproximación algunos nodos asumen un rol administrativo de la misma forma que lo eran los intermediarios en una aproximación centralizada, pero esta vez se requiere cierta comunicación entre ellos para compartir información sobre encaminamiento. El hecho de utilizar varios intermediarios permite escalar a un mayor número de suscripciones y eventos, evitando en gran medida sobrecargas como las comentadas. La ventaja es que se disponen de una serie de nodos que se reparten la carga, formando parte de la infraestructura de manera permanente y fiable, y que están dedicados a tareas específicas. A cambio, el reparto de trabajo requiere del intercambio de mensajes entre los intermediarios tanto sobre suscriptores como de eventos para que el sistema de publicación/suscripción funcione como se espera.

La otra aproximación distribuida bastante popular es la basada en topologías *peer-to-peer*, en las que en principio, todos los participantes en el sistema, ya sea como suscriptor o como publicador, forman parte de la infraestructura de distribución y asumen ciertas responsabilidades para realizar tareas de evaluación de eventos y también de distribución. Este tipo de infraestructuras permiten minimizar la sobrecarga y escalar a un mayor número de eventos y suscripciones. Pero a cambio sufren de algunos problemas que deben ser abordados. Por citar algunos, la infraestructura de distribución ya no pertenece a una entidad que provee el servicio, pudiendo plantearse problemas de privacidad. Otros problemas más evidentes son la

volatilidad de los nodos, su baja fiabilidad, sus diferentes capacidades computacionales, y la mayor cantidad de información de control y de mantenimiento requerida para la gestión de la infraestructura.

En general, la investigación en el uso de sistemas de publicación/suscripción de eventos persigue en la medida de lo posible la máxima distribución de la infraestructura con miras a explorar la escalabilidad y el rendimiento de las infraestructuras en cada caso. La distribución de la infraestructura es una tarea de diseño compleja ya que las posibilidades dependen mucho de las características del modelo de eventos, e.g. contenidos vs. categorías, y el *proceso de matching* que viene impuesto en cierta medida por los requisitos. Por ejemplo, los sistemas basados en categorías son los más sencillos, y ello ha llevado a ser los primeros en ser totalmente distribuidos en redes *peer-to-peer*. Sin embargo, los basados en contenidos son bastante más complejos, y ello requiere el diseño de complejos algoritmos distribuidos para el mantenimiento de la infraestructura aún cuando hablamos de una basada en varios intermediarios. La eficiencia y el fundamento de la distribución de la carga dependen de cómo se pueda distribuir el proceso de evaluar las correspondencias entre eventos y suscripciones también. Un ejemplo claro de ello es el caso de los sistemas que tienen un modelo semántico. En ese caso se dispone de una lógica subyacente pero todavía no se ha extendido el desarrollo y uso de algoritmos de razonamiento que fácilmente trabajen de forma distribuida, a diferencia de las aproximaciones sintácticas, lo que hace que la distribución de tales sistemas sea aún más desafiante.

4.3 Ontologías en la Web Semántica

La *Web Semántica* se definió como la evolución de la Web tal y como la conocíamos a principios del 2000[4] en la cual, en lugar de consistir únicamente de documentos que los humanos pudieran leer, se incluirían datos e información para que las computadoras pudieran manipular [Ber01][Sha06]. La Web tradicional, expone a la humanidad una cantidad inmensa de información en forma de documentos y páginas Web. Los humanos somos capaces de leer dos páginas Web, posiblemente en dos lenguas distintas, con estructuras completamente distintas, y ser capaces de relacionarlas en base al contexto y a la información que allí se maneja (aún no habiendo un enlace que apunte de una a otra). Metafóricamente se dice que la información o las páginas serían puntos, y que los humanos seríamos capaces de conectar dichos puntos. Dado que la Web fue concebida para el consumo por parte de las personas, la información está en general poco estructurada, expresada en lenguaje natural, utilizando representaciones gráficas, etc. Por ello, a las máquinas (al software) le resulta muy difícil

[4] Todavía está vigente esa perspectiva porque la evolución hacia el concepto de Web Semántica está siendo lento aunque sí que estamos evolucionado hacia estadios intermedios.

realizar "esa conexión de puntos" como hacen los humanos, entre otras cosas, como el ignorar información irrelevante como pueden ser los anuncios, etc. El objetivo principal será, pues, definir "algo" sobre la infraestructura de la Web actual para que la información sea procesable por las máquinas. Se podría decir que la Web Semántica es una "gran iniciativa" que pretende facilitar el intercambio y la integración de información por parte de agentes software por medio de dar significado a los recursos. El objetivo último es llegar a la *web de datos* (*"Web of Data" vs. "Web of Documents"*) en la que la información sea procesable por las máquinas automáticamente, para lo cual se están definiendo formatos comunes para la integración y combinación de datos desde diferentes fuentes, y lenguajes específicos para registrar cómo los datos se relacionan con el mundo real. Esta visión está muy relacionada con otros campos más específicos como el de los sistemas de agentes en el que se tratan formatos comunes para facilitar la comunicación e interoperabilidad entre agentes, y representación del conocimiento (modelos semánticos), y cómo inferir sobre ellos (obtención y utilización de información tácita). Evidentemente esta vez todo ello se aborda sobre una infraestructura global y abierta, y supone realmente un gran banco de pruebas muy interesante para ideas anteriormente desarrolladas. No obstante, es indispensable pensar desde un primer momento en lenguajes necesarios para abordar todas las tareas que potencialmente los agentes necesitarían realizar.

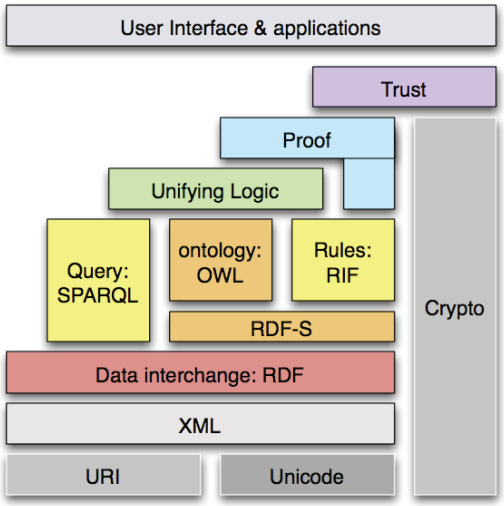


Figura 41. Pila de lenguajes y tecnologías en la Web Semántica

La Figura 41 muestra la pila de lenguajes y tecnologías que con el paso de los años se ha ido definiendo y proveyendo especificaciones e implementaciones [Bra07](*slide 23*). La Web Semántica discute principalmente aspectos relativos a 1) los formatos comunes para la integración y combinación de datos desde diferentes fuentes; y 2) lenguajes para registrar cómo los datos se relacionan a los objetos del mundo real.

Para que la información sea realmente procesable por las máquinas (*machine-readable*) se necesita *a)* alguna forma que nos permita enlazar datos con objetos del mundo real y además que lo haga de forma no ambigua. Para ello se utilizan URIs; *b)* un mecanismo de intercambio de información "en crudo", que permita la creación de formatos serializables y resuelva el problema de la codificación. Para este propósito se utiliza UNICODE, y XML; *c)* un modelo de datos común sencillo, que de acceso, conecte, y describa recursos. En este sentido se dispone de RDF (*Resource Description Framework*); *d)* también se necesita el acceso a esos datos descritos con RDF, acceso que se realiza a través del lenguaje de consulta SPARQL; *e)* definición de vocabularios comunes que permitan el entendimiento por parte de las máquinas. Esta capacidad se proporciona a través de RDFS y OWL; *f)* lógica que nos permita razonar sobre los vocabularios y los datos que sigan esas definiciones. Este proceso debe estar soportado por razonadores que trabajen principalmente sobre bases de conocimiento en OWL, así como también mediante la utilización de reglas escritas en RIF (*Rule Interchange Format*) que permitan actuar a los diferentes razonadores basados en reglas ya existentes.

En el área de Representación del Conocimiento (*Knowledge Representation, KR*), una *ontología* es el término utilizado para referirse a un compromiso mutuo sobre algún dominio de interés que puede ser utilizado como un marco de trabajo unificador para la resolución de problemas tales como la pobre comunicación entre organizaciones, interoperabilidad, reutilización y compartición del dominio [Usc96]. Se dice que una *conceptualización* es una "vista" del mundo concebida como un conjunto de conceptos, sus definiciones, y sus interrelaciones. Al expresar explícitamente una conceptualización en un lenguaje específico se obtiene lo que se conoce como *ontología del dominio*. Una *Base de Conocimiento* (*Knowledge Base, KB*) consiste de cierto conocimiento terminológico y cierto conocimiento sobre objetos entendidos como instancias. El lenguaje utilizado para expresar una ontología y sus instancias relacionadas puede ser completamente informal, e.g. lenguaje natural, o formal, e.g. usando un formalismo lógico con una semántica bien definida. El uso de formalismos es de especial importancia porque así se permite la inferencia automática sobre la base de conocimiento por medio de un razonador lógico.

Este es el caso de OWL-DL, un sublenguaje del lenguaje estándar recomendado por el W3C [OWLw3] para representar semánticamente contenidos web. Éste es fundamental porque el objetivo final de la Web Semántica es entender la Web como una web de datos en la cual diversos agentes software sean capaces de analizar automáticamente la información insertada en las páginas web y de ofrecer servicios inteligentes.

OWL-DL se fundamenta en *Description Logics* (DL) [Baa03], que es un fragmento decidible de la Lógica de Primer Orden. Por tanto, en principio, con OWL-DL se podría definir cualquier expresión que se ajuste a DL. La

expresividad de DL permite la definición de conceptos, roles, y ciertas restricciones. Los conceptos son clases que son interpretadas como conjuntos de objetos. Los roles representan relaciones y se interpretan como relaciones binarias entre objetos. Los conceptos y roles corresponden a las clases y propiedades en OWL respectivamente. Algunas de las construcciones más destacables disponibles en OWL son las que nos permiten expresar subsunción y equivalencia. Por medio de la subsunción se pueden construir jerarquías de conceptos de forma explícita mientras que con el uso de equivalencias entre clases se pueden definir clases equivalentes. Similarmente existen construcciones centradas en propiedades con las cuales se pueden definir jerarquías de propiedades y propiedades equivalentes. Estas relaciones de equivalencia son de gran utilidad ya que de esta forma los razonadores sobre DL pueden automáticamente integrar diversas ontologías en una sola de tal forma que todos los conceptos puedan ser utilizados juntos indistintamente. Esto será útil para nuestro propósito tal y como se discutirá en 4.5.

4.3.1 Lógica de Descripciones y OWL-DL

Cuando se habla de *Lógicas Descriptivas* se refiere realmente a una familia de formalismos para la representación de conocimiento. Como ya se adelantó, estos formalismos permiten la definición de ontologías, que permiten la definición del conocimiento de un dominio de aplicación (el "mundo"), primero por medio de la definición de conceptos relevantes en el dominio (su *terminología*), y entonces usando estos conceptos para especificar las propiedades de los objetos y los individuos que existen en el dominio (la descripción del mundo). Las Lógicas Descriptivas tienen su origen en las conocidas *redes de herencia estructuradas* [Bra77][Bra78]que fueron introducidas para superar la ambigüedad en las primeras *redes semánticas* y *frames*. En esencia, en el desarrollo de los lenguajes descriptivos ha imperado las ideas de que 1) los bloques de construcción sintáctica básica son conceptos atómicos (predicados unarios), roles atómicos (predicados binarios), y los individuos (constantes); 2) el poder expresivo del lenguaje está restringido en el sentido de utilizar un conjunto reducido de constructores para la creación de conceptos complejos y roles; 3) el conocimiento implícito sobre conceptos e individuos puede ser inferido automáticamente con la ayuda de procedimiento de inferencia. En particular, las relaciones de subsunción entre los conceptos y las relaciones de instancia entre individuos y conceptos juegan un papel importante ya que así no se exige el establecimiento de todas las relaciones explícitamente.

Un sistema de representación del conocimiento basado en Lógicas Descriptivas provee facilidades para establecer bases de conocimientos, para razonar sobre su contenido y para manipularlas. La Figura 42 muestra la arquitectura básica.

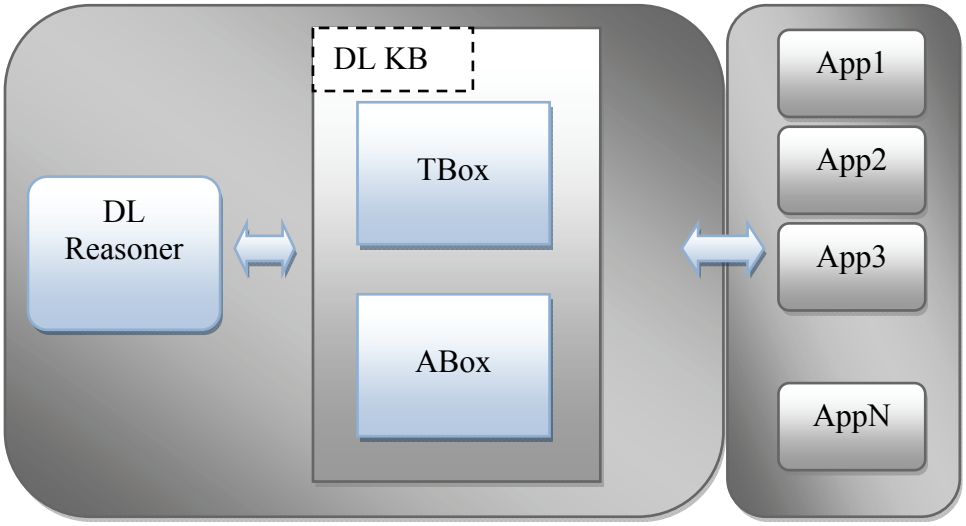


Figura 42. Arquitectura de sistemas de representación del conocimiento basados en Description Logics

Una base de conocimiento comprende esencialmente dos componentes: a saber, *TBox* y *ABox*. Lo que se conoce como *TBox* introduce la *terminología*, es decir, el vocabulario de un dominio de aplicación, mientras que la componente correspondiente a la componente *ABox* contiene aserciones sobre individuos expresados en términos del anterior vocabulario. El vocabulario consiste de conceptos, que denotan conjuntos de individuos, y de roles, que denotan relaciones binarias entre individuos. Además de conceptos y roles atómicos se permite la definición de descripciones complejas de conceptos y roles.

Como muestra la figura y también de acuerdo a las ideas directoras de desarrollo de estos lenguajes, un sistema de DL no sólo almacena terminologías y aserciones, sino que también ofrece servicios que *razonan* sobre ellos. El lenguaje para la construcción de descripciones tiene un significado y las declaraciones en la base de conocimientos se pueden identificar con fórmulas en la Lógica de Primer Orden. La formalización subyacente, posibilita cierta capacidad de inferencia y automatización. En la actualidad hay varios razonadores de DL disponibles en la comunidad lo suficientemente maduros [Sir07][Tsa06]. Los servicios de inferencia comúnmente suministrados por los razonadores DL son la *comprobación de consistencia* (*consistency checking*), *satisfabilidad de conceptos* (*concept satisfiability*), *clasificación* (*classification*), y *compresión* (*realization*). Además también soportan consultas aunque este servicio se construye habitualmente sobre los servicios de inferencia anteriores. De forma alternativa, las consultas pueden expresarse como nuevos conceptos en la ontología. Consecuentemente, los servicios de clasificación y de comprensión pueden ser invocados sobre estos conceptos para razonar sobre las

características de la consultas-concepto en sí mismas, y computando la respuesta de esta forma.

En esencia las Lógicas Descriptivas permiten la definición de conceptos o clases, ya sean simples o complejas por medio de constructores de unión, intersección o complemento, soportando la formación de taxonomías, el establecimiento de equivalencias entre ellas, así como igualdad entre individuos. Las propiedades o roles pueden definirse con ciertas características suponiendo por ejemplo una relación inversa, transitiva, simétrica, funcional o funcional inversa. También se soporta la expresión de restricciones de cardinalidad, y la descripción extensional de conceptos mediante las conocidas clases enumeradas.

OWL es un estándar y recomendación de W3C para la especificación de ontologías para la Web Semántica. El lenguaje OWL pretende facilitar una mayor capacidad de procesamiento de información por parte de las máquinas. El W3C provee de tres variantes de OWL, a saber Lite, DL y Full. Estas tres especificaciones difieren entre ellas según el grado de expresividad, que determinará la complejidad de las inferencias a realizar así como su decidibilidad. En particular, OWL-Lite es el lenguaje más restringido, y por tanto el de menor expresividad. Éste soporta principalmente clases sencillas, clasificación jerárquica, equivalencias de clases y propiedades, igualdad entre individuos, restricciones sencillas de cardinalidad explícita 0 o 1, clases complejas como intersección de otras, y la caracterización de las propiedades como inversa, transitiva, simétrica, funcional, y/o funcional inversa. Debido a su restricción los algoritmos de razonamiento son más sencillos, y tienen un menor coste computacional. Por otra parte, OWL-DL incorpora todas las capacidades de OWL-Lite pero extiende su expresividad soportando además las clases enumeradas que sirven para realizar una descripción extensional de un concepto al indicar sus individuos, clases complejas por medio de la unión y complemento, cardinalidad explícita libre, y clases disjuntas. OWL-DL, al igual que OWL-Lite, se fundamenta en los axiomas de las Lógicas Descriptivas, que supone una característica deseable ya que la completitud computacional y la decidibilidad están demostradas. Finalmente, OWL-Full permite la utilización de las clases como instancias y la extensión del vocabulario predefinido en OWL, y no se tiene garantías de decidibilidad.

En la Tabla 2 y Tabla 3 se muestra de forma sintetizada los constructores de OWL-DL, así como su correspondencia a los axiomas en DL, la semántica asociada, y una descripción informal. La Tabla 2 presenta cómo realizar ciertas descripciones y restricciones, mientras que la Tabla 3 presenta los axiomas terminológicos, que efectivamente construyen la base de conocimiento por medio del uso de las descripciones y restricciones ya presentadas utilizando los operadores de inclusión ($\sqsubseteq$) e igualdad ($\equiv$) que representan la subsunción y la equivalencia respectivamente.

Tabla 2. Descripciones de OWL-DL y Lógicas Descriptivas (I)

Abtract Syntax	DL Syntax	Semantics	Description
Class(A)	A	$A^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}}$	Define un concepto atómico A, que será subsumido por el concepto universal.
Class(owl:Thing)	$\top$	$\top^{\mathfrak{I}} = \Delta^{\mathfrak{I}}$	Representa al concepto universal que subsume a todos los conceptos.
Class(owl:Nothing)	$\bot$	$\bot^{\mathfrak{I}} = \emptyset^{\mathfrak{I}}$	Representa la noción opuesta al concepto universal (conocido como *bottom concept*).
intersectionOf(C1,C2,...)	$C_1 \sqcap C_2$	$(C_1 \sqcap C_2)^{\mathfrak{I}} = {C_1}^{\mathfrak{I}} \cap {C_2}^{\mathfrak{I}}$	Expresa descripciones de conceptos complejos que representen la intersección de conceptos.
unionOf(C1,C2,...)	$C_1 \sqcup C_2$	$(C_1 \sqcup C_2)^{\mathfrak{I}} = {C_1}^{\mathfrak{I}} \cup {C_2}^{\mathfrak{I}}$	Expresa descripciones de conceptos complejos que representen la unión de conceptos.
complementOf(C)	$\neg C$	$(\neg C)^{\mathfrak{I}} = \Delta^{\mathfrak{I}} \setminus C^{\mathfrak{I}}$	Expresa una descripción de un concepto que represente el complemento de otro concepto.
oneOf(o1,o2,...)	$\{o_1\} \sqcup \{o_2\}$	$(\{o_1\} \sqcup \{o_2\})^{\mathfrak{I}} = \{{o_1}^{\mathfrak{I}}, {o_2}^{\mathfrak{I}}\}$	Permite la descripción extensional al indicar los individuos que conforman un concepto.
restriction(R someValuesFrom(C))	$\exists R.C$	$(\exists R.C)^{\mathfrak{I}} = \{x \mid \exists y.\langle x,y\rangle \in R^{\mathfrak{I}} \wedge y \in C^{\mathfrak{I}}\}$	Cuantificación Existencial sobre un rol, exigiendo que alguna de los individuos relacionados con ese rol sea del concepto especificado C.
restriction(R allvaluesFrom(C))	$\forall R.C$	$(\forall R.C)^{\mathfrak{I}} = \{x \mid \forall y.\langle x,y\rangle \in R^{\mathfrak{I}} \rightarrow y \in C^{\mathfrak{I}}\}$	Cuantificación Universal sobre un rol, exigiendo que todos los individuos participantes en el rol debe ser del concepto indicado C.
restriction(R hasValue(o))	$\exists R.\{o\}$	$(\exists R.\{o\})^{\mathfrak{I}} = \{x \mid \langle x, o^{\mathfrak{I}}\rangle \in R^{\mathfrak{I}}\}$	Establece que el valor de un determinado rol R debe ser un individuo específico o.
restriction(R minCardinality(m))	$\geq mR$	$(\geq mR)^{\mathfrak{I}} = \{x \mid \lvert y.\langle x,y\rangle \in R^{\mathfrak{I}}\rvert \geq m\}$	Restricción de cardinalidad mínima asociada al rol.
restriction(R maxCardinality(m))	$\leq mR$	$(\leq mR)^{\mathfrak{I}} = \{x \mid \lvert y.\langle x,y\rangle \in R^{\mathfrak{I}}\rvert \leq m\}$	Restricción de cardinalidad máxima asociada al rol.

restriction(T someValuesFrom(u))	$\exists T.u$	$(\exists T.u)^{\mathfrak{I}} = \{x \mid \exists t.\langle x,t\rangle \in T^{\mathfrak{I}} \wedge t \in u^{D}\}$	Cuantificación Existencial sobre un rol, exigiendo que alguna de los individuos relacionados con ese rol sea del tipo de datos especificado u.
restriction(T allvaluesFrom(u))	$\forall T.u$	$(\forall T.u)^{\mathfrak{I}} = \{x \mid \forall t.\langle x,t\rangle \in T^{\mathfrak{I}} \rightarrow t \in u^{D}\}\}$	Cuantificación Universal sobre un rol, exigiendo que todos los individuos participantes en el rol debe ser del tipo de datos indicado u.
restriction(T hasValue(w))	$\exists T.\{w\}$	$(\exists T.\{w\})^{\mathfrak{I}} = \{x \mid \langle x, w^{D}\rangle \in T^{\mathfrak{I}}\}$	Establece que el valor de un determinado rol R debe ser un valor específico w.
restriction(T minCardinality(m))	$\geq mT$	$(\geq mT)^{\mathfrak{I}} = \{x \mid \lvert t.\langle x,t\rangle \in T^{\mathfrak{I}}\rvert \geq m\}$	Restricción de cardinalidad mínima asociada al rol.
restriction(T maxCardinality(m))	$\leq mT$	$(\leq mT)^{\mathfrak{I}} = \{x \mid \lvert t.\langle x,t\rangle \in T^{\mathfrak{I}}\rvert \leq m\}$	Restricción de cardinalidad máxima asociada al rol.
ObjectProperty(S)	S	$S^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times \Delta^{\mathfrak{I}}$	Expresa la descripción de un rol que implicará a conceptos.
ObjectProperty(S' inverseOf(S))	S^{-}	$(S^{-})^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times \Delta^{\mathfrak{I}}$	Expresa la descripción de un rol como inverso de otro.
DatatypeProperty(T)	T	$T^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times \Delta_{D}$	Expresa la descripción de un rol en el que se implica un tipo de datos.

La convención utilizada en las tablas se define como sigue. Sea C, R_I, R_D, y I los conjuntos de *URIs* que pueden utilizarse para denotar respectivamente clases, propiedades abstractas, propiedades de tipos de datos, e individuos. Una interpretación es una tupla $\mathfrak{I} = (\Delta^{\mathfrak{I}}, \Delta_D, \cdot^{\mathfrak{I}}, \cdot^{D})$ donde el dominio de individuos $\Delta^{\mathfrak{I}}$es un conjunto no vacío de individuos, el dominio de tipos de datos Δ_D es un conjunto no vacío de valores, $\cdot^{\mathfrak{I}}$es una función de interpretación de individuos que hace corresponder:

- cada nombre de individuo a∈ I a un elemento a$\cdot^{\mathfrak{I}} \in \Delta^{\mathfrak{I}}$,
- cada nombre de clase CN∈ C a un subconjunto $CN\cdot^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}}$,
- cada nombre de propiedad abstracta RN∈ R_I a una relación binaria $RN^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times \Delta^{\mathfrak{I}}$,

- cada nombre de propiedad de tipo de datos TN∈ T_D a una relación binaria $TN^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times \Delta_D$,

y $\cdot^D$ es una función de interpretación de tipos de datos.

El símbolo A∈ C denota el uso de una clase, y C, C_1, C_2,… son descripciones de clases. S∈ R_I es el nombre de un rol mientras que R es una descripción de un rol. Por otra parte o, o_1, o_2,… ∈I son individuos, u representa un rango de datos, mientras que T∈ R_D representa un rol de tipo de dato.

Tabla 3. Axiomas OWL-DL y Lógicas Descriptivas (II)

Abtract Syntax	DL Syntax	Semantics	Description
Class(A partial C_1,…,C_n)	$A \sqsubseteq C_1 \sqcap \dots \sqcap C_n$	$A^{\mathfrak{I}} \subseteq {C_1}^{\mathfrak{I}} \cap \dots \cap {C_n}^{\mathfrak{I}}$	Describe el concepto A como aquel que es subsumido por la intersección de un conjunto de conceptos dado.
Class(A complete C_1,…,C_n)	$A \equiv C_1 \sqcap \dots \sqcap C_n$	$A^{\mathfrak{I}} = {C_1}^{\mathfrak{I}} \cap \dots \cap {C_n}^{\mathfrak{I}}$	Describe el concepto A como aquel completamente definido por la intersección de un conjunto de conceptos dado.
EnumeratedClass(A o_1,…,o_n)	$A \equiv \{o_1\} \sqcup \dots \sqcup \{o_n\}$	$A^{\mathfrak{I}} = \{{o_1}^{\mathfrak{I}} \cap \dots \cap {o_n}^{\mathfrak{I}}\}$	Describe el concepto A de forma extensional por enumeración de sus individuos.
SubClassOf(C_1,C_2)	$C_1 \sqsubseteq C_2$	${C_1}^{\mathfrak{I}} \subseteq {C_2}^{\mathfrak{I}}$	Establece una relación de subsunción directa entre las clases.
EquivalentClasses(C_1,…,C_n)	$C_1 \equiv \dots \equiv C_n$	${C_1}^{\mathfrak{I}} = \dots = {C_n}^{\mathfrak{I}}$	Establece una relación de equivalencia entre las clases.
DisjointClasses(C_1,…,C_n)	$C_i \sqsubseteq \neg C_j : (1 \leq i < j \leq n)$	${C_i}^{\mathfrak{I}} \cap {C_j}^{\mathfrak{I}} = \emptyset : (1 \leq i < j \leq n)$	Establece una relación de disyunción entre ciertos conceptos
SubPropertyOf(R_1,…,R_n)	$R_1 \sqsubseteq R_2$	${R_1}^{\mathfrak{I}} \subseteq {R_2}^{\mathfrak{I}}$	Establece una relación de subsunción entre roles.
EquivalentProperties(R_1,…,R_n)	$R_1 \equiv \dots \equiv R_n$	${R_1}^{\mathfrak{I}} = \dots = {R_n}^{\mathfrak{I}}$	Establece una relación de equivalencia entre roles.
ObjectProperty(R super(R_1) … super(R_n)	$R \sqsubseteq R_i$	$R^{\mathfrak{I}} \subseteq {R_i}^{\mathfrak{I}}$	Expresa la descripción del rol,
domain (C_1) … domain(C_k)	$\geq 1R \sqsubseteq C_i$	$R^{\mathfrak{I}} \subseteq {C_i}^{\mathfrak{I}} \times \Delta^{\mathfrak{I}}$	Establece el dominio del role a conceptos específicos
range (C_1) … range(C_h)	$\top \sqsubseteq \forall R.C_i$	$R^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times {C_i}^{\mathfrak{I}}$	Establece el rango del role a conceptos específicos.
[Symmetric]	$R \equiv R^-$	$R^{\mathfrak{I}} = (R^-)^{\mathfrak{I}}$	Establece el rol como simétrico

[Functional]	$Func(R)$	$\{\langle x,y\rangle \mid \lvert\{y.\langle x,y\rangle \in R^{\mathfrak{I}}\}\rvert \leq 1\}$	Establece el rol como funcional, lo que significa que si el rol se utiliza para relacionarse con dos instancias, éstas serán consideradas para ser la misma.
[InverseFunctional]	$Func(R^{-})$	$\{\langle x,y\rangle \mid \lvert\{y.\langle x,y\rangle \in (R^{-})^{\mathfrak{I}}\}\rvert \leq 1\}$	Establece el rol como funcional inverso, lo que significa que si dos instancias mediante un rol se relacionan con una tercera instancia, entonces las dos instancias primeras se consideran la misma.
[Transitive])	$Trans(R)$	$R^{\mathfrak{I}} = (R^{\mathfrak{I}})^{+}$	Establece el rol como transitivo
DatatypeProperty(T super(T1) ... super(Tn)	$T \sqsubseteq T_i$	$T^{\mathfrak{I}} \subseteq T_i^{\mathfrak{I}}$	Expresa la descripción del rol de tipo de datos,
domain (C1) ... domain(Ck)	$\geq 1T \sqsubseteq C_i$	$T^{\mathfrak{I}} \subseteq C_i^{\mathfrak{I}} \times \Delta_D$	Establece el dominio del rol a conceptos específicos
range (d1) ... range(dh)	$\top \sqsubseteq \forall T.d_i$	$T^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}} \times {d_i}^D$	Establece el rango del role a tipos de datos específicos.
[Functional])	$Func(T)$	$\{\forall x \in \Delta^{\mathfrak{I}} \mid \lvert\{t.\langle x,t\rangle \in T^{\mathfrak{I}}\}\rvert \leq 1\}$	Establece el rol como funcional
Individual(o type(C1) ... type(Cn)	$o: C_i, 1 \leq i \leq n$	$o^{\mathfrak{I}} \in C_i^{\mathfrak{I}}, 1 \leq i \leq n$	Aserta un individuo o como instancia del concepto indicado,
value(R1, o1) ... value(Rn, on)	$\langle o, o_i\rangle: R_i, 1 \leq i \leq n$	$\langle o^{\mathfrak{I}}, {o_i}^{\mathfrak{I}}\rangle \in R_i^{\mathfrak{I}}, 1 \leq i \leq n$	tomando valores en los roles que le correspondan
SameIndividual(o1,...,on)	$o_1 = \cdots = o_n$	${o_1}^{\mathfrak{I}} = \cdots = {o_n}^{\mathfrak{I}}$	Expresa la equivalencia entre individuos
DifferentIndividuals(o1,...,on)	$o_i \neq o_j, 1 \leq i < j \leq n$	$o_i^{\mathfrak{I}} \neq o_j^{\mathfrak{I}}, 1 \leq i < j \leq n$	Expresa la diferencia entre individuos

Para ejemplificar la expresión de conocimiento mediante DL y OWL-DL supongamos que queremos expresar una ontología en la que se recojan algunos conceptos sobre relaciones familiares. La Tabla 4 muestra un ejemplo concreto.

Tabla 4. Ejemplo descripción mediante DL y OWL-DL

Lógicas Descriptivas	OWL-DL (sintaxis abstracta)
$Person \sqsubseteq \forall name.String \sqcap \exists name \sqcap (\geq 1 name)$	SubClassOf(Person intersectionOf(restriction(hasName someValuesFrom(xsd:string)) restriction (hasName allValuesFrom(xsd:string))

$Male \sqsubseteq \exists hasGender.\{'Male'\}$	restriction(hasName minCardinality(1)))
$Female \sqsubseteq \exists hasGender.\{'Female'$	SubClassOf(Person owl:Thing)
$Male \sqsubseteq \neg Female$	SubClassOf(Male owl:Thing)
$Man \equiv Person \sqcap \neg Woman$	SubClassOf(Male restriction(hasGender hasValue("Male")))
$Woman \equiv Person \sqcap Female$	SubClassOf(Female restriction(hasGender hasValue("Female")))
$Mother \equiv Woman \sqcap \exists hasChild.Person$	SubClassOf(Female owl:Thing)
	DisjointClasses(Female Male)
$Father \equiv Man \sqcap \exists hasChild.Person$	Class(Man)
$Parent \equiv Mother \sqcup Parent$	EquivalentClasses(Man intersectionOf(Person Male))
$Child \equiv Person \sqcap \exists hasChild^{-}.Person$	Class(Woman)
$Person2 : Woman$	EquivalentClasses(Woman intersectionOf(Person Female))
$\langle Person2, "Nerea" \rangle : hasName$	Class(Father)
$\langle Person2, "Female" \rangle : hasGender$	EquivalentClasses(Father intersectionOf(restriction(hasChild someValuesFrom(Person)) Man))
$Person1 : Man$	SubClassOf(Father owl:Thing)
$\langle Person1, "Jorge" \rangle : hasName$	Class(Mother)
$\langle Person1, "Male" \rangle : hasGender$	EquivalentClasses(Mother intersectionOf(restriction(hasChild someValuesFrom(Person)) Woman))
$\langle Person1, Person2 \rangle : hasChild$	SubClassOf(Mother owl:Thing)
	Class(Parent)
	EquivalentClasses(Parent unionOf(Mother Father))
	Class(Child)
	EquivalentClasses(Child intersectionOf(Person restriction(hasParent someValuesFrom(Person)))
	DisjointClasses(Male Female)
	ObjectProperty(hasChild range(Person))
	ObjectProperty(hasParent inverseOf(hasChild) range(Person))
	DataProperty(hasGender range(xsd:string))
	DataProperty(hasName range(xsd:string))

	Individual(Person1 type(Man) value(hasChild Person2) value(hasName "Jorge"^^xsd:string) value(hasGender "Male"^^xsd:string)) Individual(Person2 type(Woman) value(hasName "Nerea"^^xsd:string) value(hasGender "Female"^^xsd:string))

Básicamente se define el concepto *Person* como aquello que tenga un nombre, y los conceptos *Male* y *Female* como conceptos disjuntos. Sobre estos tres conceptos se define el resto de conceptos de interés. Una vez definidos los conceptos de *Man* y *Woman* que esencialmente son la intersección de los individuos que cumplen con la descripción de *Male* y *Female* respectivamente con *Person*, se puede proceder a la definición de *Father* y *Mother*, como aquellos conceptos que tienen algún hijo. Entonces se define el concepto de *Parent* como la unión de *Mother* y *Father*, y el concepto *Child* se define como las personas que tengan "padres". Se ha definido el rol *hasParent* como inverso de *hasChild*. Esto permite que el razonador mantenga automáticamente las instancias de los roles con tan sólo explicitar las instancias para el rol *hasChild* como es el caso del ejemplo, en el que se han definido dos instancias de *Person*.

Más información introductoria sobre Lógicas Descriptivas puede encontrarse en [Baa03](especialmente Cap. 1 & 2), y [DLw3]. Información más detallada y específica sobre OWL-DL y su relación con DL puede encontrarse en [OWLw3],[Hor04],[Pan07].

4.4 Una Aproximación a la Publicación/Subscripción Semántica

Una vez introducidos los conceptos básicos concernientes a los sistemas de publicación/suscripción de eventos, y a la representación del conocimiento mediante OWL-DL es momento de introducir en este punto la aproximación al sistema de publicación/suscripción semántico que hemos abordado como combinación de ideas de ambos campos.

Como ya se introdujo en el punto 4.1, se quiere soportar la comunicación de eventos entre diferentes entidades. Entidades que pueden haber definido sus tipos de eventos de forma diferente para representar sucesos similares, y que expresen cierto comportamiento como reacción a los eventos que acontezcan. Sobre las definiciones de los tipos de eventos se establecen

ciertas relaciones de equivalencia, que contribuyen a la visión del mundo particular de cada entidad, y por ello se optó por el uso de ontologías como mecanismo natural de integración. Por tanto, la aproximación se centra en el desarrollo de forma genérica de una plataforma que soporte la publicación/suscripción de eventos semánticos que se podrá utilizar para el propósito final de comunicar entidades coexistentes en el ecosistema de las superficies interactivas.

Cuando hablamos de eventos semánticos, excluyendo de este concepto el simple etiquetado de mensajes de acuerdo a ciertas ontologías de forma similar a como se abordaría en los sistemas clásicos de publicación/suscripción basados en categorías, nos estamos refiriendo a definir complejas estructuras de información que definen el contenido del evento y en la que los lenguajes utilizados tienen asociada cierta semántica bien definida que facilita la manipulación e integración de dicha información. El concepto de sistemas de eventos semánticos no es nuevo en absoluto. En [Pet03], se presenta una infraestructura para eventos semánticos pero en el que sólo se utiliza un vocabulario de sinónimos y una jerarquía de conceptos. En [Pet05] y en [Wan04], las ontologías se expresan por medio de taxonomías de clases RDF, y ambos utilizan un proceso de *matching* "ad hoc" basado en diversos aspectos de la Teoría de Grafos ya que se consideran grafos RDF. No obstante, un proceso de razonamiento no está realmente presente en esas aproximaciones. En cambio, en [Usc03] y [Li04] se basan en ontologías DAML+OIL (que puede considerarse como el antecesor inmediato de OWL) y razonamiento sobre DL. Además, en [Las05] se extiende el razonamiento de DL con programas lógicos más generales para superar algunas limitaciones en expresividad para sus aplicaciones. En [Hal07] se propone el uso de OWL en sistemas RSS con el propósito de poder confiar en el uso de un razonador de DL para el proceso de "matching" y obtener un mecanismo de sindicación más expresivo.

La aproximación que se presenta en este capítulo consiste de un sistema de publicación/suscripción de eventos cuya descripción se realiza en el lenguaje para ontologías recomendado como estándar en la Web Semántica, OWL-DL. Esto permitirá, orientando adecuadamente la estructura del servicio de publicación/suscripción, el uso de un razonador en DL para llevar a cabo el proceso de *matching,* que de otra forma se trataría de un proceso *ad hoc*. El publicador debe, o bien definir una ontología representando su propio modelo de eventos y el conocimiento del dominio, o bien adoptar una ontología ya existente que se ajuste a sus necesidades. Las ontologías deben escribirse en OWL-DL y deben ser registradas en el sistema antes de poder ser utilizadas. Los publicadores deben proveer eventos como individuos OWL-DL expresados en términos de su propia ontología de tal forma que éste pasa a ser conocimiento insertado en la base de conocimiento del sistema de eventos.

Más allá de la definición de la ontología, las correspondencias entre conceptos también se suministran tal y como veremos más adelante para permitir al razonador de DL funcionar indistintamente con los eventos de todas las ontologías.

Los suscriptores especifican sus suscripciones como consultas mediante expresiones de clase OWL-DL. Finalmente, el razonador de DL se utiliza para determinar qué suscripciones deben responder a qué eventos al resolver las consultas.

La Figura 43 muestra a grandes rasgos la estructura del *intermediario* desarrollado y finalmente implementado.

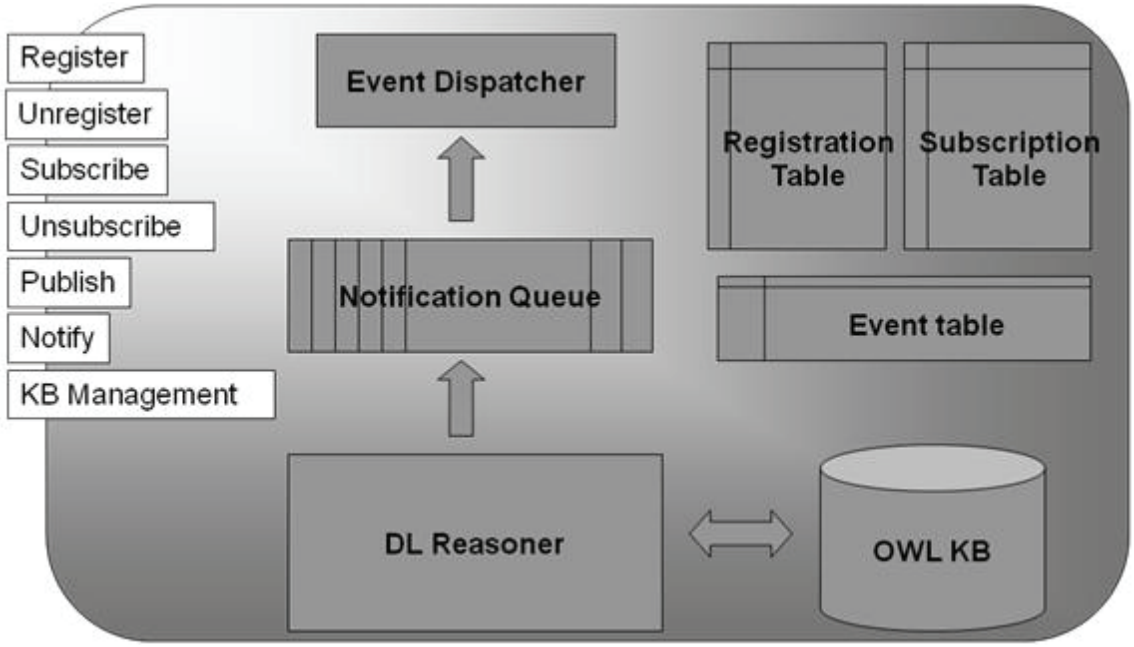


Figura 43. Estructura del broker

Las tablas de búsqueda rápida de "registro", de "suscripciones" y de "eventos" mantienen la información sobre los publicadores y los suscriptores, las suscripciones y la relación de eventos. Una parte importante es la base de conocimiento de OWL, que mantiene las ontologías y los eventos que se van publicando.

El razonador de DL periódicamente trata de resolver las consultas que representan las suscripciones obteniendo los eventos que deben ser notificados, y almacenando esta información en la cola de notificación. Finalmente, el despachador de eventos consume los eventos de la cola de notificación empezando la comunicación efectiva de los mismos hacia los suscriptores correspondientes.

El servicio "*Register*" permite al invocador el obtener acceso al sistema de publicación/suscripción para utilizar el resto de servicios. El servicio

"*Unregister*" indica el fin de la utilización del sistema por parte del invocador liberando los recursos que éste consumía. Los métodos "*Subscribe*" y "*Unsubscribe*" permiten indicar suscripciones expresadas en términos de alguna ontología, e indicar que ya no se desea estar suscrito a alguna suscripción respectivamente. El servicio "*Publish*" permite el ingresar eventos en el sistema, y el servicio "*Notify*" será invocado desde el *intermediario* para entregar los eventos a los suscriptores respectivos. El *intermediario* tiene la interfaz típica de cualquier otra infraestructura de publicación/suscripción, como se ha podido ver, pero la diferencia principal radica en la necesidad de un servicio adicional, al que de forma genérica se le ha dado el nombre de "KB Management", para llevar a cabo tareas sobre las ontologías, como puede ser añadir ontologías, establecer equivalencias, etc.

Para ejemplificar el uso de los lenguajes implicados en el sistema a bajo nivel supongamos que hay al menos dos entes (publicadores y suscriptores) que se dedican al sector de noticias de agencias. Cada uno de ellos podría trabajar con un modelo de eventos ligeramente diferente. Por ejemplo, uno de los publicadores podría definir un titular como un "*Headline"*, mientras que otro podría definir "*Latest news*", publicando el primero un evento de la forma que se muestra a continuación:

Listado 7. Muestra de evento sobre la ontología Ont1

```
<Headline rdf:about="#SampleEvent1">
  <hasAgency>CNN News</hasAgency>
  <hasText rdf:datatype="&xsd;string">
   Elderly residents rescued from fire</hasText>
  <hasPublishing_DateTime rdf:datatype="&xsd;dateTime">
   20080409T13:30:00</hasPublishing_DateTime>
  <hasJournalist>John Smith</hasJournalist>
  <hasAppropiateness_Grade>6<hasAppropiateness_Grade/>
</Headline>
```

Por otra parte el otro publicador podría basarse en otra ontología aunque similar y publicar un evento como el siguiente:

Listado 8. Muestra de evento sobre la ontología Ont2

```
<Latest News rdf:about="#SampleEvent2">
  <hasOrganization>REUTERS</hasOrganization>
  <hasPublishing_DateTime rdf:datatype="&xsd;dateTime">
    2008-04-09T12:00:00</hasPublishing_DateTime>
  <hasPublisher>Michael Jones</hasPublisher>
  <hasHeadline>Floods and drought to rise due to climate
change</hasHeadline>
  <hasAppropiateness_Grade>9<hasAppropiateness_Grade/>
</Latest News>
```

En ambos casos los eventos están representando dos titulares de acuerdo a sus propias ontologías, conteniendo como información relevante la agencia de noticias, el periodista que lo escribió, la fecha de publicación, el texto, y una calificación sobre su idoneidad que es útil para filtrar noticias. Además de las ontologías, también habría que proveer de las correspondencias entre conceptos para permitir al razonador de DL funcionar indistintamente con

los eventos de ambas ontologías, tal y como se realiza para el ejemplo anterior mediante las equivalencias expresadas a continuación[5]:

Listado 9. Equivalencias entre conceptos en Ont1 y Ont2 con respecto a Ont1

```
...
<owl:Class rdf:about="#Headline">
  <owl:equivalentClass rdf:resource="&Ont2;Latest_News"/>
  <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>

<owl:DatatypeProperty rdf:about="&Ont2;hasHeadline">
  <owl:equivalentProperty rdf:resource="#hasText"/>
</owl:DatatypeProperty>
...
```

Por su parte, los suscriptores especifican suscripciones mediante consultas como ya se comentó anteriormente, y el razonador de DL del *intermediario* se encargará de lanzar el proceso de resolución. Este proceso obtendrá los individuos (i.e. eventos) que se clasifican bajo la clase de las expresiones de consulta. Dado que las equivalencias entre los conceptos de las ontologías han sido establecidas, las suscripciones pueden ser expresadas en términos de cualquiera de las ontologías registradas. A continuación se muestra un ejemplo de consulta:

Listado 10. Ejemplo de consulta OWL

```
Ont1:Headline that Ont1:hasAgency value 'REUTERS' and
Ont1:hasAppropiateness Grade some int[>= 7]
```

La suscripción devolverá cada evento que sea publicado por una agencia "REUTERS" y que estén marcadas con una idoneidad mayor o igual a 7, que en este ejemplo se corresponde a la noticia expresada mediante la Ont2 pese a haber expresado la consulta en términos de Ont1.

Como se ha mostrado, las ontologías y los eventos se representan mediante la sintaxis *OWL/XML*, mientras que las suscripciones se escriben mediante la sintaxis de conocida como *Manchester OWL* [Manw3], que es una sintaxis más amigable de OWL muy próxima a la sintaxis de DL.

La implementación del sistema se ha realizado en Java. Para tratar con el proceso de confrontación de suscripciones frente a eventos se ha utilizado el razonador Pellet [Pelw3][Sir07], y para el manejo de ontologías se ha hecho uso de OWL-API [owlapi], y algunas APIs de Protégé [Prow3]. La Figura 44 muestra el diagrama de clases de implementación con las principales clases y sus propiedades y métodos más importantes. El núcleo del *intermediario* está implementado en el paquete *broker.core*. Además el núcleo se puede utilizar entre diferentes procesos por mediación de las clases de los paquetes *broker*, *broker.messaging* y *broker.client*. En ese caso la clase *broker* envuelve el núcleo y gestiona las comunicaciones que los clientes realicen, que suponen los publicadores y los suscriptores. Para los clientes la

[5] El caso del ejemplo las relaciones de equivalencia son triviales y directas pero podría utilizarse expresiones de DL más complejas

funcionalidad se expone convenientemente a través de una clase *proxy*, de forma que la gestión de las comunicaciones sea totalmente transparente. Las comunicaciones se llevan a cambo a través de paquetes TCP/IP que envían mensajes codificados mediante las clases del paquete *broker.messaging* que contiene las clases correspondientes para cada tipo de mensaje que supone cada servicio así como también las clases para mensajes de reconocimiento y resultado de operaciones.

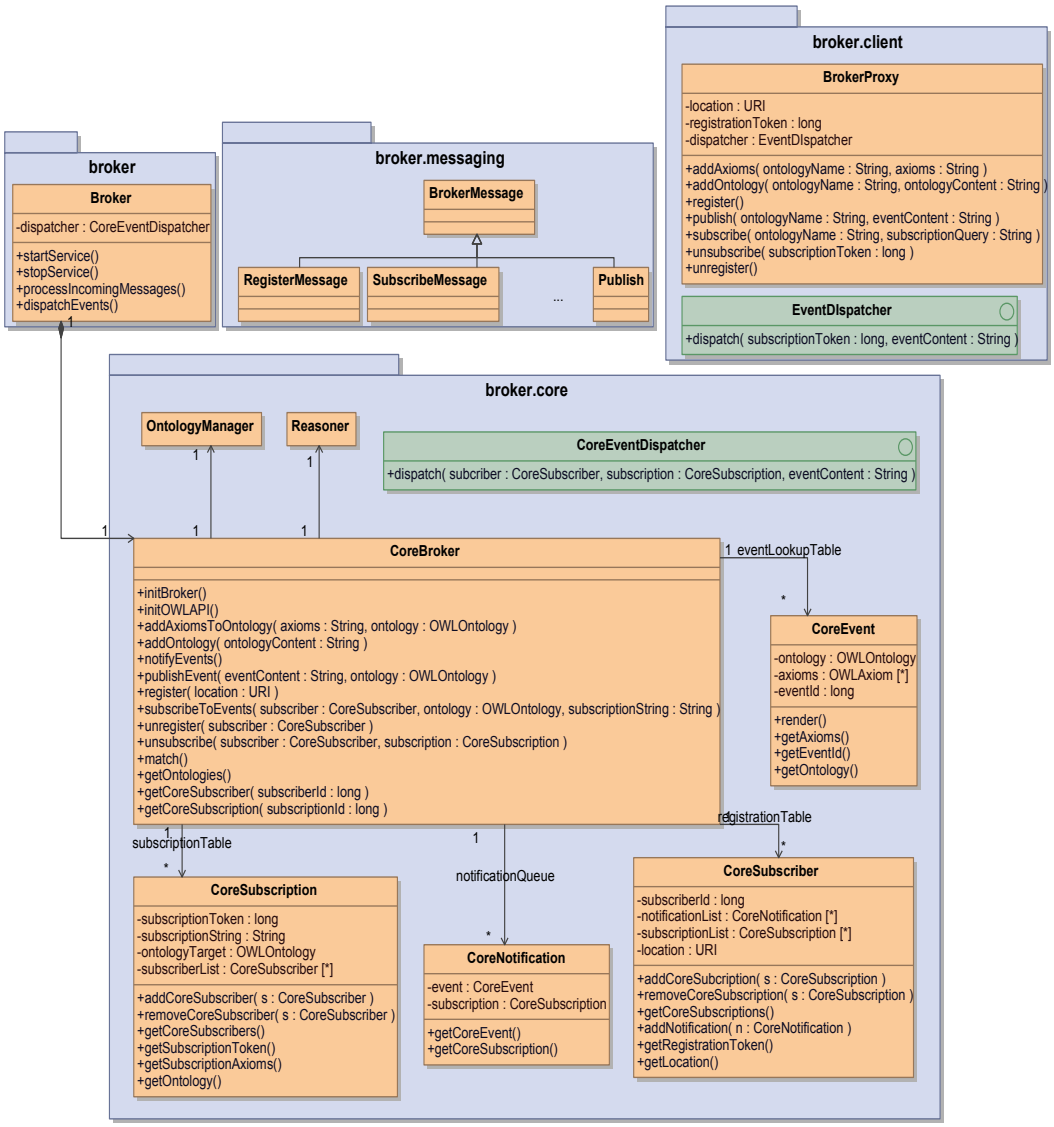


Figura 44. Diagrama de clases de implementación del *intermediario*

Nuestra implementación actual todavía trata el problema de la distribución desde un punto de vista totalmente centralizado basado en un *intermediario*. Tal y como se manifestaba en la introducción a los sistemas de publicación/suscripción de eventos, la distribución de la infraestructura dependía en gran medida de la posibilidad de distribuir el proceso de *matching*, y los datos. Precisamente la Web Semántica por medio del uso de

URIs permite una fácil distribución de los datos pero todavía no se ha madurado lo suficiente en la distribución de los procesos de razonamiento, que en nuestro caso supone el proceso de "*matching*". El primer tipo de distribución está empujando la investigación hacia la investigación en la distribución de los procesos de razonamiento de forma que se aproveche dicha distribución de datos para diseñar algoritmos de razonamiento distribuidos más efectivos y escalables sobre lo que se conoce como ontologías modulares [Ser05][Bor06][Bao06][Wan07]. No obstante, todo apunta a, que en cualquier caso, la evolución utilizando las nuevas aproximaciones distribuidas será en primera instancia partir hacia sistemas basados en varios *intermediarios*.

4.5 Subsistema para la simulación

Con los modelos en el punto 4.1 sólo se pueden definir tipos de eventos, entidades y actores que queremos que estén disponibles en el ecosistema. Sin embargo, se necesita ir más allá porque también es necesario jugar en el ecosistema. En general, necesitamos una forma apropiada para evaluar las ocurrencias de los eventos a los patrones de eventos. En particular, dado que se quiere manejar diferentes puntos de vista, uno por cada actor, se necesita una forma fácil para tratar con equivalencias y relaciones entre tipos de evento. Además, hay que tener en cuenta que con el modelo estamos permitiendo el diseño de actores, tipos de entidad, y tipos de evento, y como consecuencia se necesita de un mecanismo que de forma genérica cubra estos requisitos. Para alcanzar ese propósito vamos a centrarnos en la utilización de diferentes espacios tecnológicos. Se dice que un espacio tecnológico es un contexto de trabajo en el que se dispone de un conjunto de conceptos, una base de conocimiento, habilidades y herramientas para el desempeño de cierta actividad por parte de una comunidad. En el contexto de la computación algunos ejemplos de espacios tecnológicos son los lenguajes de programación concretos, los árboles de sintaxis abstracta, los lenguajes basados en XML, los sistemas de gestión de bases de datos, etc. La idea de utilizar espacios tecnológicos, de tal forma que mediante proyección y mantenimiento de la dualidad de las representaciones en los mismos, se aborde la resolución de ciertos problemas que de otra forma no sería factible y/o muy complicado se basa en las ideas expuestas en [Kur02]. En nuestro caso se pretende descansar sobre un proceso de traducción que resulta en un conjunto de axiomas y aserciones en un espacio ontológico basado en ontologías OWL y razonadores de Lógicas Descriptivas, que en última instancia faciliten al subsistema de eventos genérico el llevar a cabo la evaluación de eventos y de reglas.

Se ha señalado que los eventos y los patrones de eventos son asuntos clave para los ecosistemas reactivos que se están proponiendo. Como un primer

paso para soportarlos, y dado que queremos utilizar el subsistema de eventos presentado en el punto 4.4, se necesitará desarrollar un subsistema de autoría y de simulación que se soporte en diferentes capas de abstracción y funcionalidad como el mostrado en la Figura 45, que está claramente dividido en dos espacios tecnológicos concretos: el del ecosistema y el de OWL.

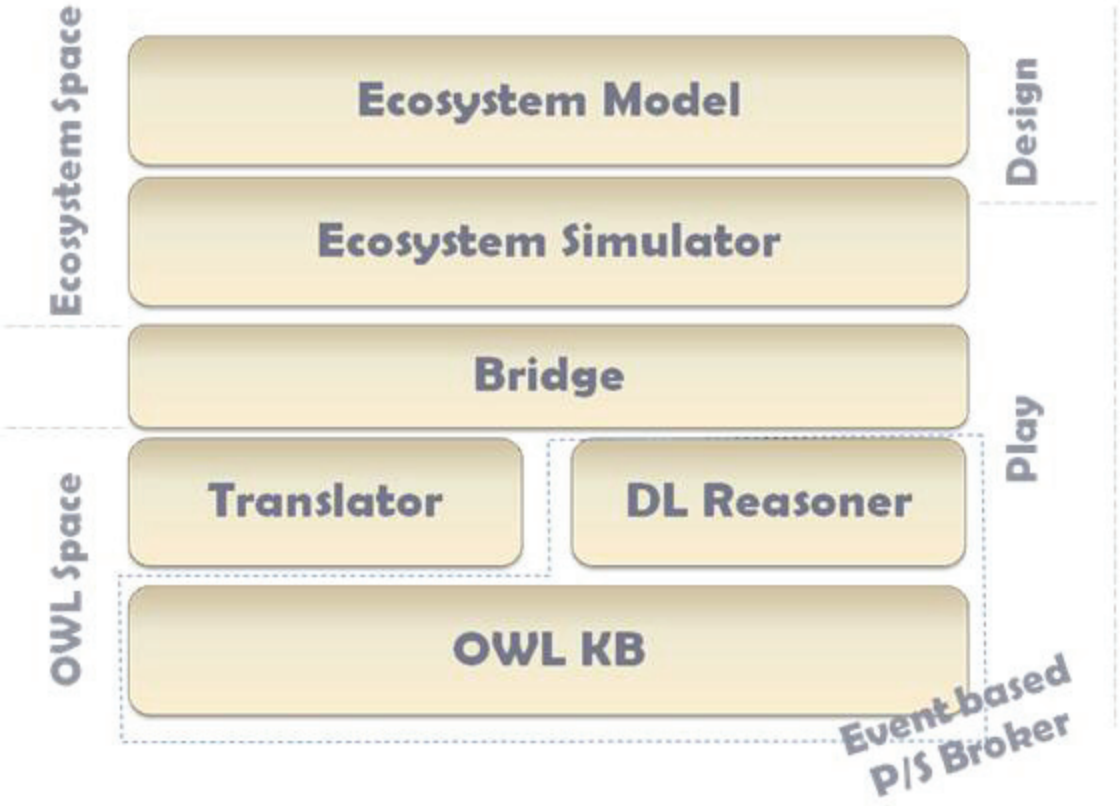


Figura 45. Perspectiva general del subsistema de autoría y simulación.

El espacio del ecosistema consiste principalmente de un modelo de ecosistema y de un simulador de ecosistemas. En tiempo de diseño, el ecosistema se confecciona y algunos parámetros de ajuste del simulador tendrán que establecerse de acuerdo a las características del ecosistema. En tiempo de juego, el simulador es el responsable para traer el modelo a la vida. Por supuesto, el espacio de ecosistema podría ser implementado sin las "cosas" disponibles en el espacio de OWL, pero seguramente se requeriría una implementación *ad hoc* más compleja para la gestión de eventos. A grandes rasgos el espacio de OWL consiste de una base de conocimiento de OWL, que estará compuesta de un conjunto de ontologías dependiendo de las definiciones hechas en el ecosistema, una razonador de DL, para el soporte de la evaluación de eventos, y una componente traductora, que haga efectiva la proyección del modelo del ecosistema en la base de conocimiento. En realidad donde se ubica la base de conocimiento y el razonador de DL es precisamente el lugar del *intermediario* del subsistema de eventos

implementado. El *puente* actúa como un marco de trabajo común entre ambos espacios, aislando y desacoplándolos, y proveyendo de servicios para mantener la dualidad en ambos espacios. Esto nos permitirá expandir fácilmente la funcionalidad del espacio de OWL de acuerdo a nuevos requisitos del sistema en el futuro, e incluso sustituirlo.

Por tanto, nuestro subsistema de simulación descansa sobre un paradigma de proyecciones y dualidad de conceptos en dos espacios diferentes, y lo hace siguiendo el siguiente esquema. Los tipos de entidades y los tipos de eventos son subsumidos por las clases de OWL *_Entity* y *_EventType* respectivamente. Estas clases han sido creadas exclusivamente para ese propósito. Las propiedades (*Property*) de la definición de un tipo de evento y los parámetros formales de los tipos de eventos son subsumidos por propiedades en *_Property*, y *_EventParameter* en OWL respectivamente[6]. Las definiciones de tipos de entidad (*EntityType*) se proyectan a subclases de *_Entity* en OWL. Las definiciones de propiedades (*Property*) se proyectan a subpropiedades de *_Property* en OWL. De esta forma se obtienen dos jerarquías cuyas raíces son *_Entity* y *_Property*. El Listado 11 es un fragmento de la traducción en sintaxis *OWL/XML* para la definición de una *EntityType* cuyo nombre es "EntityType1" y una de sus propiedades es "property1".

Listado 11. Muestra para una definición de tipo de evento

```
… <!-- declared in the eco model ontology -->
<SubClassOf>
  <OWLClass URI="&eco_model;EntityType1"/>
  <OWLClass URI="&eco_core;_Entity"/>
</SubClassOf>
…
<SubObjectPropertyOf>
  <ObjectProperty URI="&eco_model;EntityType1_Property"/>
  <ObjectProperty URI="&eco_core;_EntityProperty"/>
</SubObjectPropertyOf>
<SubObjectPropertyOf>
  <ObjectProperty URI="&eco_model;EntityType1_Property_property1"/>
  <ObjectProperty URI="&eco_model;EntityType1_Property"/>
</SubObjectPropertyOf>
<ObjectPropertyDomain>
  <ObjectProperty URI="&eco_model;EntityType1_Property_property1"/>
  <OWLClass URI="&eco_model;EntityType1"/>
</ObjectPropertyDomain>
<ObjectPropertyRange>
  <ObjectProperty URI="&eco_model;EntityType1_Property_property1"/>
  <OWLClass URI="&eco_model;EntityType2"/>
</ObjectPropertyRange>
```

Respecto a los eventos, las definiciones de tipos de evento (*EventType*) generan una serie de axiomas. En primer lugar se crea una subclase OWL de *_EventType*, y en segundo lugar una subpropiedad de *_EventParameter* en OWL representando una propiedad general para los parámetros formales del tipo de evento en particular. Cada parámetro formal (*FormalParameter*)

[6] El modelo real es más complejo porque en OWL hay propiedades cuyo rango son objetos y propiedades cuyo rango son tipos de datos básicos tal y como mostraron las tablas de axiomas, pero se ha ignorado este detalle en nuestra discusión por claridad.

en una definición de tipo de evento produce una subpropiedad de la propiedad general creada específicamente para subsumir las definiciones de los parámetros del tipo de evento en cuestión. Esta vez obtenemos otras dos jerarquías cuyas raíces son *_EventType* y *_EventParameter*. El Listado 12 presenta una muestra de una definición de tipo de evento para "EventType1" y sus parámetros, "parm1" y "parm2".

Listado 12. Muestra de definición de tipo de evento

```
… <!-- declared in the eco model ontology -->
<SubClassOf>
  <OWLClass URI="&eco_model;EventType1"/>
  <OWLClass URI="&eco_core;_EventType"/>
</SubClassOf>
<SubObjectPropertyOf>
  <ObjectProperty URI="&eco_model;EventType1_Parameter"/>
  <ObjectProperty URI="&eco_core;_EntityEventParameter"/>
</SubObjectPropertyOf>
<SubObjectPropertyOf>
  <ObjectProperty URI="&eco_model;EventType1_Parameter_parm1"/>
  <ObjectProperty URI="&eco_model;EventType1_Parameter"/>
</SubObjectPropertyOf>
<SubObjectPropertyOf>
  <ObjectProperty URI="&eco_model;EventType1_Parameter_parm2"/>
  <ObjectProperty URI="&eco_model;EventType1_Parameter"/>
</SubObjectPropertyOf>
```

En relación a las equivalencias entre tipos de eventos, cada entidad tiene su propia ontología para la expresión de su visión particular del mundo, i.e. sus definiciones de correspondencias del modelo del ecosistema se proyectarán en una ontología particular. A esta ontología se restringirá el alcance de las relaciones de equivalencia y subsunción allí definidas. Cada correspondencia (*EventTypeMapping*) producirá los correspondientes axiomas que indiquen la subsunción entre tipos de eventos en la ontología particular de la entidad que las define. En cuanto a las correspondencias sobre sus parámetros (*ParameterMapping*), se producen axiomas de equivalencia entre propiedades sobre cada par de las propiedades OWL correspondientes que representan los parámetros formales. Si se quisiera expresar que la ocurrencia de uno de los eventos siempre implica el reconocimiento del otro tipo de evento y viceversa, entonces se tendría que establecer dos correspondencias en las que se intercambian los roles. Un ejemplo de este último caso, que se concreta en una equivalencia entre "EventType1" y "EventType2" y uno de sus parámetros se muestra en el Listado 13. En este caso sólo queremos la equivalencia de estos dos tipos de eventos y sencillamente se producen axiomas de equivalencia de clases.

Listado 13. Muestra de correspondencia entre definiciones de eventos

```
<!-- defined in the EntityType1 viewpoint ontology that includes
eco_model and eco_core ontos-->
<EquivalentClasses>
  <OWLClass URI="&eco_model;EventType1"/>
  <OWLClass URI="&eco_model;EventType2"/>
</EquivalentClasses>
<EquivalentObjectProperties>
  <ObjectProperty URI="&eco_model;EventType1_Parameter_parm1"/>
  <ObjectProperty URI="&eco_model;EventType2_Parameter_parm1"/>
</EquivalentObjectProperties>
```

Las instancias concretas de las definiciones de los tipos de entidad junto con los valores de las propiedades concretas corresponden a individuos de OWL, que se insertan en la base de conocimiento. En particular, cada una de estas instancias se declara como un individuo de la clase de OWL representando la definición del tipo de entidad a la que pertenece. Cada vez que el valor de una propiedad cambia en el espacio del ecosistema, el *puente* debería cambiar el valor correspondiente en el espacio de OWL. La ocurrencia de eventos son declarados como individuos de las clases OWL correspondientes que representan su tipo de eventos, y los valores de sus parámetros también se establecen para cada uno de ellos de forma acorde a su definición.

Listado 14. Muestra de instancia de entidad y de una ocurrencia de evento

```
<ClassAssertion>
  <OWLClass URI="&eco_model;EntityType1"/>
  <Individual URI="&eco_model;entity001"/>
</ClassAssertion>
<ObjectPropertyAssertion>
  <ObjectProperty URI="&eco_model;EntityType1_Property_property1"/>
  <Individual URI="&eco_model;entity001"/>
  <Individual URI="&eco_model;entity002"/>
</ObjectPropertyAssertion>
<ClassAssertion>
  <OWLClass URI="&eco_model;EventType1"/>
  <Individual URI="&eco_model;event001"/>
</ClassAssertion>
<ObjectPropertyAssertion>
  <ObjectProperty URI="&eco_model;EventType1_Parameter_parm1"/>
  <Individual URI="&eco_model;event001"/>
  <Individual URI="&eco_model;entity001"/>
</ObjectPropertyAssertion>
<ObjectPropertyAssertion>
<ObjectProperty URI="&eco_model;EventType1_Parameter_parm2"/>
  <Individual URI="&eco_model;event001"/>
  <Individual URI="&eco_model;entity002"/>
</ObjectPropertyAssertion>
```

Finalmente, el antecedente de cada regla de comportamiento, i.e. patrón de eventos, se traduce a una consulta en OWL, y el proceso de evaluar los patrones de eventos y ocurrencias de eventos se lleva a cabo por medio de un proceso de razonamiento, de tal forma que el razonador de DL gestiona toda la información en la base de conocimiento, obtenida como proyección del espacio del ecosistema, y finalmente se computa la instanciación del antecedente. Por ejemplo, si tuviéramos un patrón de evento como `EventType1(?X, entity001, ?Y) / ?Y.EntityType1_Property_property2 >= 45`, donde "EventType1" es una definición de tipo de evento que acepta tres parámetros; "EntityType1_Property_property2" es una propiedad para la definición del tipo de entidad "EntityType1"; ?X y ?Y son variables cuyos tipos se determinan por la definición del parámetro; y EntityType1_Property_property2>=45 es una condición adicional. Este patrón se traduciría a una expresión de DL escrita en sintaxis Manchester OWL como la siguiente:

```
EventType1 that EventType1_Parameter_parm1 value entity001 and
EventType1_Parameter_parm2 some (EntityType1_Property_property2 some
int [>= 45])
```

Una vez el antecedente haya sido instanciado por el razonador, el *puente* puede entonces instanciar los patrones de acción e instar al simulador a realizar su invocación. La Tabla 5 resume las proyecciones al espacio OWL.

Tabla 5. Resumen de proyecciones al espacio OWL

Espacio del Ecosistema	Espacio OWL
Definición de EntityType para "EntityType1"	1. clase OWL "EntityType1" como subclase de "_Entity" 2. propiedad OWL "EntityType1_Property" como subpropiedad de "_EntityProperty"
Definición de propiedad para "property1" en relación con "EntityType1"	Propiedad OWL "EntityType1_Property_property1" como subpropiedad de "EntityType1_Property"
Definición de EventType para "EventType1"	1. clase OWL "EventType1" como subclase de "_EventType" 2. propiedad OWL "EventType1_Parameter" como subpropiedad de "_EntityEventParameter"
FormalParameter relacionado a la definición de EventType	Propiedad OWL "EventType1_Parameter_parm1" como subpropiedad de "EventType1_Parameter"
Definición de EventTypeMapping entre tipos de eventos "EventType1" y "EventType2"	Axioma de subsunción entre "EventType1" y "EventType2"
Definición ParameterMapping relacionada una definición de un tipo de evento	Axioma OWL de equivalencia entre propiedades OWL "EventType1_Parameter_parm1" y "EventType2_Parameter_parm1
Instancia de una definición de EntityType, e.g."entity001" como instancia de "EntityType1"	Individuo OWL "entity001" como instancia de la clase OWL "EntityType1"

Cada entidad puede ser vista como la propietaria de un modelo de eventos, i.e. su modelo de eventos, y sobre ellas existe la ontología común del mundo que contiene conceptos con significado común a todas las entidades. Además de realizar las proyecciones hay que realizarlas en las ontologías adecuadas.

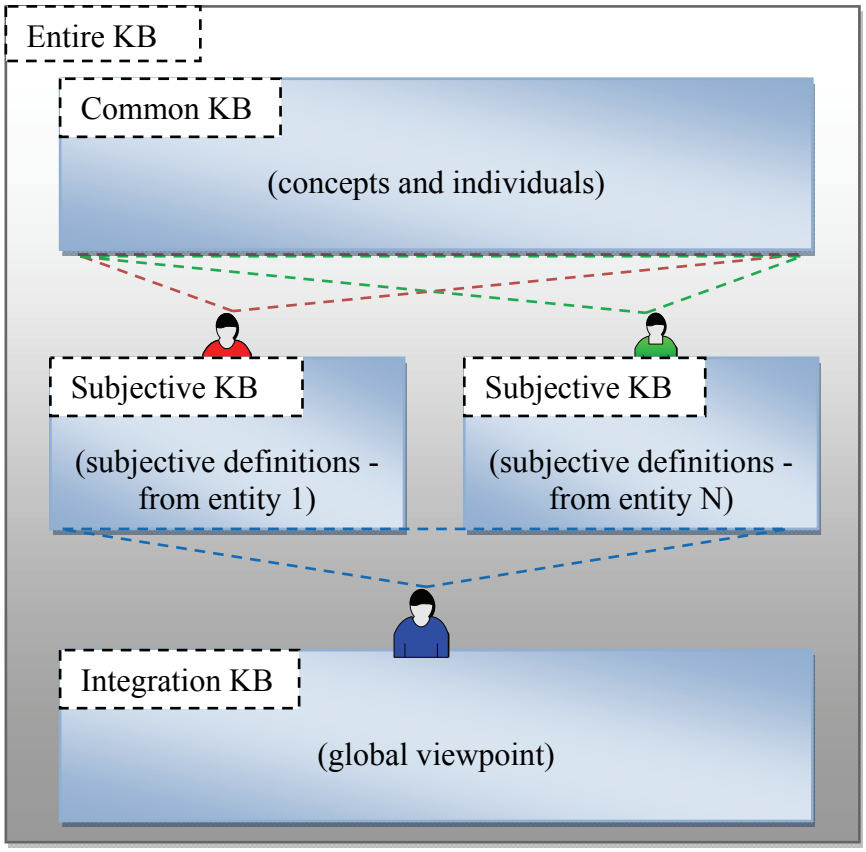


Figura 46. Jerarquía entre ontologías para soportar la subjetividad de las entidades

Consideremos una ontología como una "teoría", sobre la que un proceso de búsqueda-respuesta debería poder trabajar y obtener respuestas estando restringido exclusivamente a dicha teoría. Si cada entidad quiere obtener respuestas que son sólo válidas para ella en el caso de que sus preguntas impliquen percepciones subjetivas, entonces deberá sólo trabajar sobre su teoría y no sobre la del resto, lo que significa que no hay que mezclarlas. La Figura 46 muestra de forma intuitiva la jerarquía de ontologías que debemos establecer para poder responder al requisito de subjetividad en nuestro sistema de publicación/suscripción de eventos. Dispondremos de una o más ontologías que conformarán la base de conocimiento común, en la que estarán definidos todos los conceptos e individuos comunes, en los que no existe una percepción subjetiva. Cada entidad tendrá una teoría o punto de vista, que se materializará en una ontología. Dicha ontología referencia a la común y además contiene las equivalencias y subsunciones entre tipos de eventos para cumplir con la declaración de correspondencias en el modelo del ecosistema, conformando su punto de vista y delimitando el alcance de las definiciones contenidas en la misma. Supongamos un escenario en el que se hayan diseñado paralelamente diversas entidades en las que de una u otra forma expresen necesidades de agua. Por ejemplo, algunas entidades de tipo "planta" podrían lanzar eventos "padece_sequedad"; otras como tipos de "edificio" podrían lanzar el evento "en_llamas"; y otras como los "tanques_de_suministro_urbano" podrían lanzar el evento "bajo_mínimos".

Todas esas definiciones formarían parte de la ontología común. Seguidamente, un tipo de entidad "camión_bomberos" podría ejercer un rol en el escenario que supusiera la extinción de incendios, el regar las plantas, y el proveer de agua a los tanques de suministro para la población para lo cual podría definir un tipo de evento "necesidad_de_agua" y haciendo que este subsuma a los tres tipos eventos anteriormente mencionados. De esta forma se podría escribir reglas de comportamiento en las que el camión de bomberos pudiera actuar ante el acontecimiento de cualquiera de las tres situaciones.

Adicionalmente se podría definir un nivel más en la jerarquía de las ontologías, que correspondería a un nivel integrador o de punto de vista global, ya que desde esa posición se tendría una visión global en la que se aplicarían todas las percepciones definidas por las entidades al mismo tiempo, aunque en principio sólo tendría sentido si se necesitara tener algún proceso que requiriera la obtención de respuestas desde una visión global del ecosistema. No obstante, no se descarta en estos momentos que a medida que se planteen escenarios de uso para su implementación efectiva en superficies interactivas puedan surgir más niveles en la jerarquía, especialmente como integración de pares de entidades, lo cual no sería un problema para el subsistema de eventos, sino que tan sólo supondría modificar la funcionalidad del *puente*.

4.6 Conclusiones

En este capítulo se ha presentado el modelo básico de ecosistema reactivo que deberá soportar el entorno de aprendizaje final, y que deberá ser extendido en el momento se determinen completamente las características del motor de simulación a desarrollar.

Dado que el funcionamiento de los ecosistemas se centra en la comunicación de eventos, en este capítulo se ha presentado la implementación de un subsistema de comunicaciones de eventos basado en un paradigma de publicación/suscripción. Dicho subsistema se caracteriza por estar basado en contenidos y en particular, por el hecho de que los eventos se expresan en términos de ontologías OWL que pueden introducirse en el subsistema. El uso de ontologías permitirá el establecimiento de equivalencias entre tipos de eventos por medio de axiomas de equivalencia y de subsunción de conceptos permitiendo la posibilidad de diseñar las entidades de forma paralela y de poseer un punto de vista particular que podrá ponerse en común al final del diseño.

Aunque por el momento la infraestructura consiste de un único *intermediario*, y en principio para el entorno de aprendizaje previsto no sería necesario recurrir a la distribución de la infraestructura, sería

razonable su estudio en el caso de que se piense en el diseño de entornos compuestos de varias superficies interactivas persiguiendo una buena escalabilidad. Además sería interesante para otras aplicaciones que de forma genérica necesiten un subsistema de publicación/suscripción semántico similar.

Finalmente, dado que se pretende el desarrollo último de herramientas de autoría y de simulación que trabajen sobre el sistema de comunicaciones, se ha presentado cómo debe ser el subsistema de simulación desde el punto de vista del mantenimiento de la dualidad entre dos espacios tecnológicos: el modelo del ecosistema y el de OWL.

CAPÍTULO V

CONCLUSIONES Y TRABAJO FUTURO

5.1 Conclusiones

En la presente tesis de máster se han presentado dos infraestructuras software que permitirán el desarrollo posterior de un entorno de aprendizaje creativo planteado sobre superficies interactivas. Sin embargo, la relevancia de este trabajo no es en sí misma el desarrollo de dichas partes, sino que radica en el concepto que queremos desarrollar presentado en el capítulo 2 que realmente es lo que está suscitando mayor interés en la comunidad.

Hasta la fecha el uso de superficies interactivas en el ámbito de la educación y el aprendizaje se había centrado en el desarrollo de aplicaciones de juegos educativos, que si bien tenían en cuenta las ventajas ofrecidas por la naturaleza de esta tecnología, tales como la interacción social y la mediación física, se limitaban a aplicaciones muy específicas totalmente predeterminadas en las que no se fomentaba la creatividad.

Sin embargo, en la visión mostrada en esta tesis de máster se persigue el suministrar una herramienta a los educadores que les permita mediante el diseño de actividades de creación, diseño, y ejecución de ecosistemas, el promover y estimular la creatividad, la motivación y los conocimientos de los estudiantes, y mejorar los procesos de aprendizaje al proveer de un entorno adecuado que dé cabida a la experimentación, reflexión, actividad, y discusión colaborativa.

5.2 Trabajos futuros

Quedan todavía muchos aspectos que desarrollar en el futuro próximo, tanto en relación a los desarrollos presentados en los capítulos 3 y 4, como

también para conseguir el entorno final presentado en el capítulo 2. A continuación se esbozan los más inmediatos y evidentes:

- Construcción efectiva de la superficie interactiva.
- Abordar los eventuales problemas que pudieran surgir al utilizar el subsistema de marcadores en la superficie interactiva (sintonización).
- Añadir una capa de abstracción sobre el subsistema de marcadores para proveer facilidades para la gestión de la interacción.
- Determinar las capacidades del motor de simulación, y consecuentemente extender el modelo de ecosistema reactivo e implementar el subsistema de simulación que también comprende el puente entre el ecosistema y su representación en OWL.
- Evaluar diferentes motores de medios que puedan ser utilizados en nuestro entorno.
- Analizar, diseñar, e implementar el software final que integre todos los motores desarrollados y/o utilizados que permita la creación, y ejecución efectiva del juego.

5.3 Publicaciones

En el momento de la redacción de ese documento se han producido las siguientes publicaciones directamente relacionadas con la misma.

- Alejandro Catalá, Javier Jaén. "A Semantic Model for Reactive Entities to Support Collaborative Game Design". Proceedings of the ACM Conference on the Future of Game Design and Technology, "ACM FuturePlay 2008", 2008 (ISBN 978-1-60558-218-4). (Indexed ACM Digital Library)

- Alejandro Catalá, Javier Jaén, José A. Mocholí. "A Semantic Publish/Subscribe Approach for U-VR Systems Interoperation", In proc. of the IEEE International Symposium on Ubiquitous Virtual Reality 2008 (ISUVR08), 2008 (ISBN 978-0-7695-3259-2). (Indexed DBLP, IEEE Library)

- Alejandro Catalá, Javier Jaén, José A. Mocholí. "Juegos ubicuos: experiencias de aprendizaje óptimas", capítulo del libro "Videojuegos y aprendizaje", 2008, Editorial Graó (ISBN 978-84-7827-539-7).

- Raquel Acosta, Alejandro Catalá, Jose Miguel Esteve, Jose Antonio Mocholí, Javier Jaén. "eCoology: un sistema para aprender jugando". Revista Novática nº 182, julio-agosto 2006 (ISSN 0211-2124).

5.4 Otros resultados

Además de las publicaciones, que reflejan los resultados del presente trabajo en cuanto a investigación, son también destacables los aspectos concernientes a la transferencia de tecnología, su impacto social y las colaboraciones externas iniciadas:

- El subsistema de reconocimiento de marcadores ha sido incorporado a "MoMo", que es un producto comercial licenciado por la *UPV* lo que demuestra un concreto resultado de transferencia de tecnología a un producto ya existente.

- La tecnología de marcadores desarrollada fue licenciada por la *UPV* a los laboratorios de investigación de *Telefónica I+D*, para formar parte de su catálogo de demostraciones de tecnologías en la sede de sus laboratorios.

- Fruto de ello, el desarrollo del subsistema de reconocimiento de marcadores tuvo un cierto impacto social en forma de artículos de prensa tanto de en periódicos de tirada nacional como en ediciones regionales en toda España, como por ejemplo en *ABC, Levante, Las Provincias, El Correo Gallego, Canarias7* entre otros; reportajes en noticias de televisión como en *Notícies 9* de *Canal 9*, *Tele7*; y reportajes en programas de televisión de divulgación científica como *Campus Universitari* de *Punt 2* y en *UPV TV*. Dicha diseminación ha tenido lugar en el contexto del proyecto de diseminación de resultados científicos concedido a la UPV por el FECYT. Este esfuerzo de divulgación viene avalado por la Comisión Interministerial de Ciencia y Tecnología ya que supone uno de los seis objetivos estratégicos definidos en la nueva Estrategia Nacional de Ciencia y Tecnología.

- Relación con la cooperativa "La Fundició", dedicada a la oferta de servicios educativos y producción cultural, para el uso y experimentación con la plataforma en desarrollo en sus actividades pedagógicas.

- Relación con el grupo de investigación *Music Technology Group* de la *Universidad Pompeu Fabra de Barcelona* por mediación de Sergi Jordà, coordinador del área de *Interactive Music Systems*, para el desarrollo de un proyecto de investigación nacional conjunto.

APÉNDICE A. LIBRERÍA DE MARCADORES

El presente apéndice muestra todos los marcadores existentes en la librería.

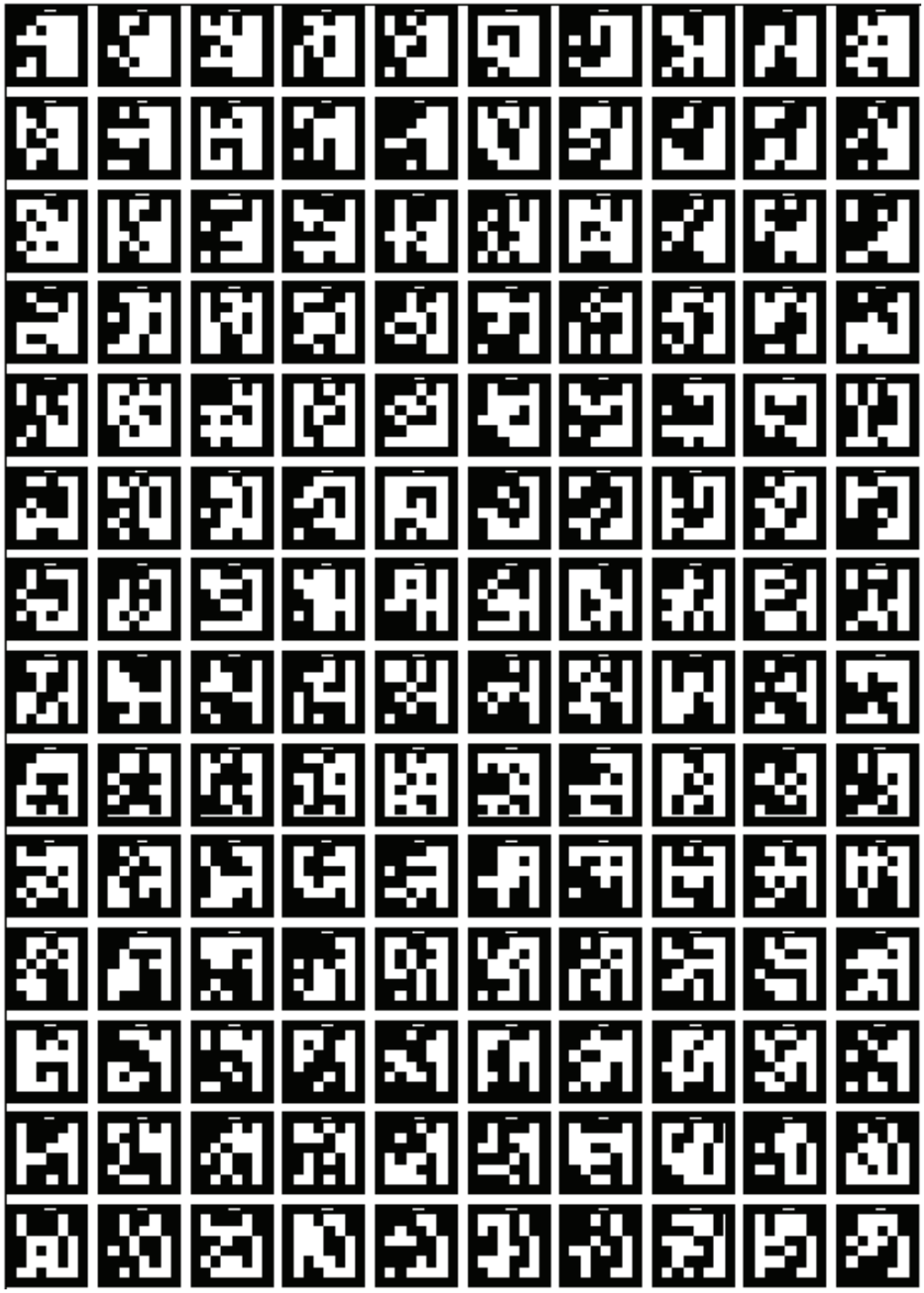


Figura 47. Marcadores desde el ID = 1 hasta el ID = 140

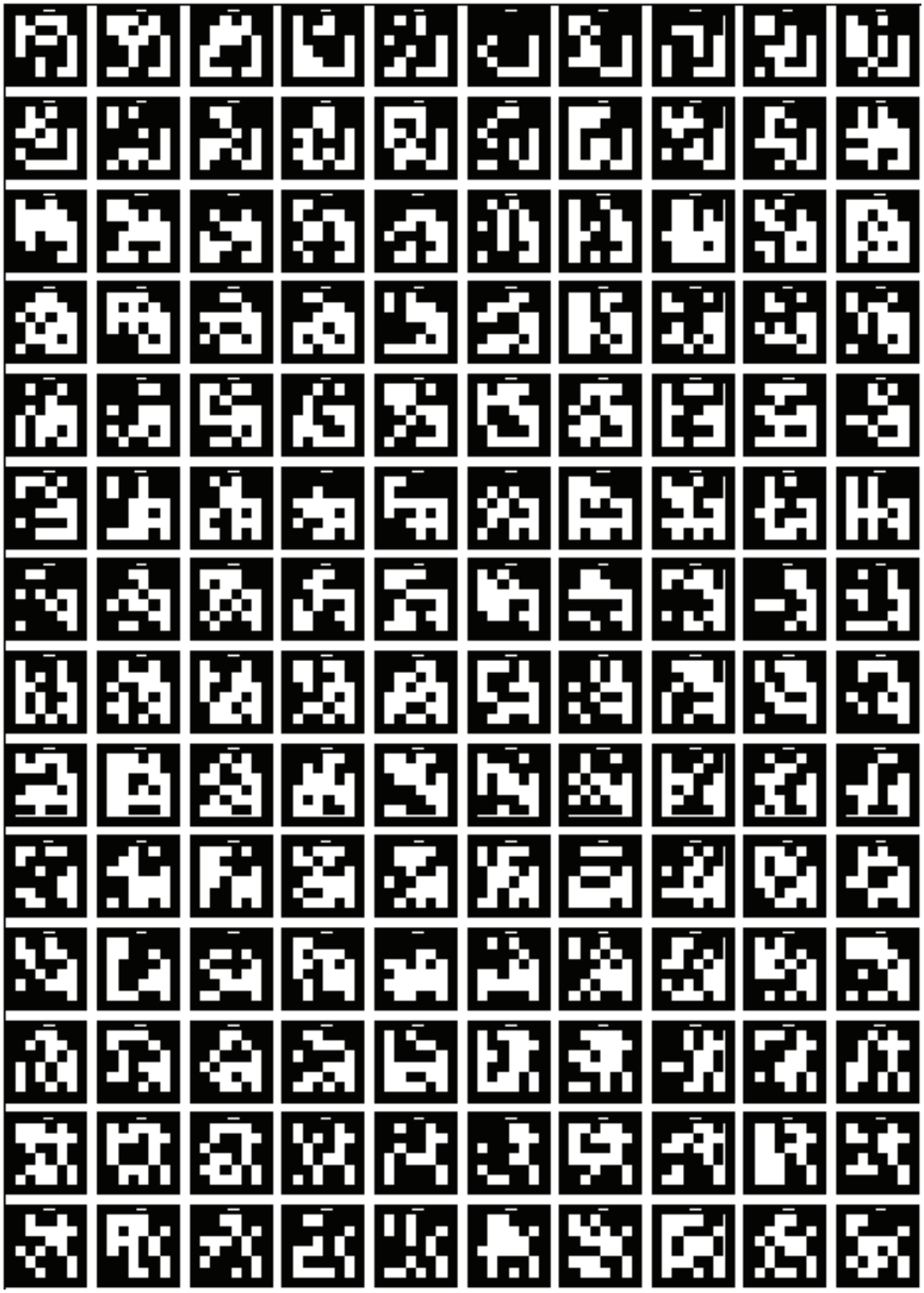


Figura 48. Marcadores desde el ID = 141 hasta el ID = 280

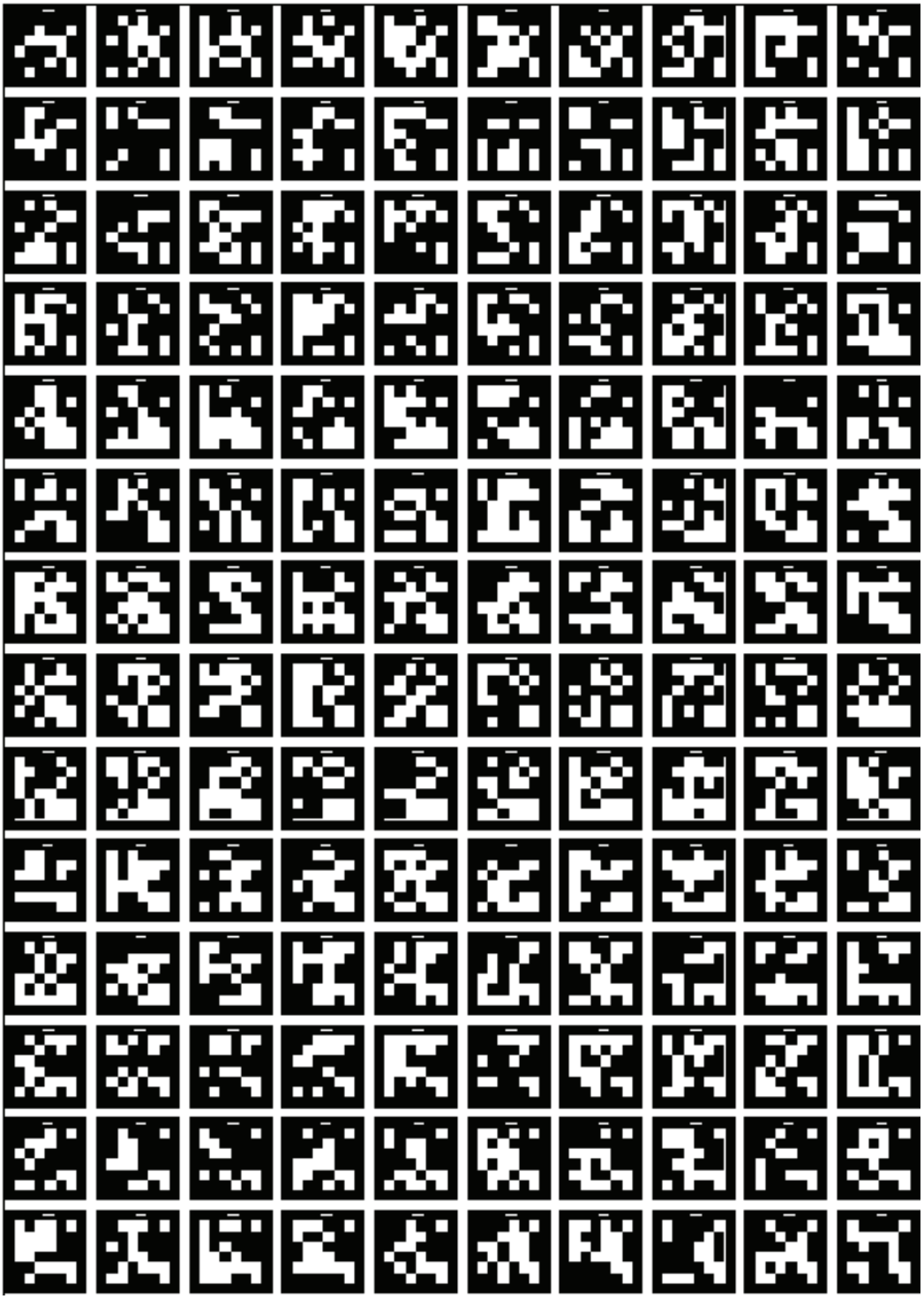


Figura 49. Marcadores desde el ID = 281 hasta el ID = 420

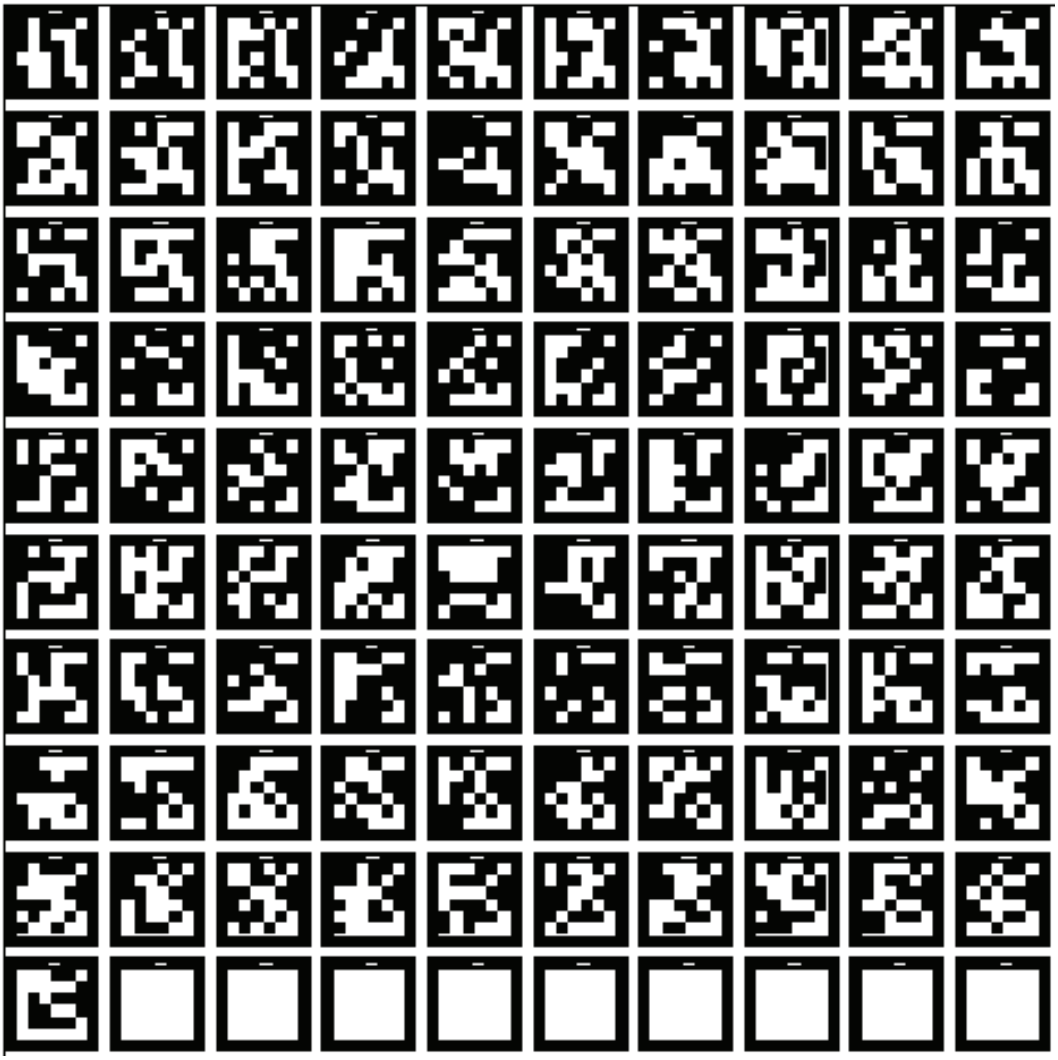


Figura 50. Marcadores desde el ID = 421 hasta el ID = 512

APÉNDICE B. LEVENBERG-MARQUARDT

Levenberg-Marquardt es un método de optimización para la resolución de problemas de mínimos cuadrados no lineales. Dado un vector función f: $R^n \mapsto R^m$ con $m \geq n$ queremos minimizar $\|f(x)\|$, o encontrar de manera equivalente:

$x^* = \text{argmin}_x\{F(x)\}$,

donde

$$F(x) = \tfrac{1}{2} \sum_{i=1}^{m} (f_i(x))^2 = \frac{1}{2}\|f(x)\|^2 = \frac{1}{2} f(x)^T f(x), \text{ y}$$

$$f_i(x) = y_i - M(x, t_i)$$

Donde x en modelo M es la lista de parámetros a estimar, y t_i el argumento correspondiente a y_i.

Este método se basa en el cálculo del *Jacobiano* **J,** es decir de una matriz $\mathbf{J} \in \mathbf{R}^{m \times n}$ donde cada elemento se define como,

$$(\mathbf{J}(x))_{ij} = \frac{\partial f_i}{\partial x_j}(x)$$

Dado que el vector **f** es una función continua, tiene segundas derivadas parciales.

Entrada

La función M(x, t_i) que nos permita calcular $f_i(x)$ para cada observación.

La derivada parcial de M respecto a cada uno de los parámetros a estimar que nos permita calcular el Jacobiano.

Lista de observaciones: {(t_i, y_i)}

```
     Los valores iniciales para los parámetros a estimar x
     Parámetros de critero de parada: μ, ε, k_max

Salida

    Conjunto de parámetros x* de forma que se minimiza F(x)

Inicio

    k = 0;  v = 2;  x = x_0;
    A = J(x)^T J(x);   g = J(x)^T f(x);
    found = ( ||g||_∞ ≤ ε_1);    μ = τ * max{a_ii}  ;
    mientras (not found) and (k < k_max)
        k = k+1;      resolver (A + μ I)h_lm = -g
        si ||h_lm|| ≤ ε_2(||x|| + ε_2)
            found = true;
        sino
            x_new = x + h_lm
            ρ = (F(x) - F(x_new)) / ½ h_lm^T(μh_lm - g)
            si ρ > 0
                x_new = x
                A = J(x)^T J(x); g = J(x)^T f(x);
                found = ( ||g||_∞ ≤ ε_1);
                μ = μ * max{1/3, 1-(2ρ-1)^3}; v = 2;
            sino
                μ = μ * v;   v = 2 * v;
            finsi

         finsi

    finmientras
Fin
```

Uno de los problemas en los que estamos interesados en llevar a cabo una estimación de ciertos parámetros de acuerdo al criterio de mínimos cuadrados es la expresión siguiente:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \left\| \mathrm{x}_{ij} - \hat{\mathrm{x}}(\mathrm{K}, \mathbf{R}_i, \mathbf{t}_i, \mathrm{x}_j) \right\|^2$$

En ella los parámetros a estimar son K, $\mathbf{R}_i$ y $\mathbf{t}_i$ y nos serviría para obtener el modelo de cámara visto en este documento[7]. La función del modelo M(x, t), i.e. $\hat{\mathrm{x}}(\mathrm{K}, \mathbf{R}_i, \mathbf{t}_i, \mathrm{x}_j)$, que recoge estos parámetros es:

[7] Por claridad en este anexo no incluimos la estimación completa en la que contemple la distorsión radial en el modelo de cámara, aunque evidentemente ello sólo supone una extensión a las expresiones que aquí se muestran.

$$M = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} \alpha_u & c & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array} \right] \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix}$$

La tercera columna de la matriz de rotación se anula por lo que **R** toma la forma,

$$\mathbf{R_T} = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \\ r_{31} & r_{32} \end{bmatrix}$$

Calculamos *M* como:

$$M = \begin{bmatrix} (\alpha_u\ r_{11} + c\ r_{21}\ +\ u_0\ r_{31})X\ +(\alpha_u\ r_{12} + c\ r_{22}\ +\ u_0\ r_{32})Y\ +\ (\alpha_u\ t_x + c\ t_y\ +\ u_0\ t_z) \\ (\alpha_v\ r_{21} + v_0\ r_{31})X\ +(\alpha_v\ r_{22} + v_0\ r_{32})Y\ +\ (\alpha_v\ t_y + v_0\ t_z) \\ r_{31}\ X\ +r_{32}\ Y\ +t_z \end{bmatrix}$$

Adicionalmente, necesitamos proveer las derivadas parciales de *M* respecto: α_u, c, u_0, α_v, v_0, t_x, t_y, t_z para construir el *Jacobiano*.

$$\frac{\partial M}{\partial \alpha_u} = \begin{bmatrix} r_{11}X + r_{21}Y\ +\ t_x \\ 0 \\ 0 \end{bmatrix} \qquad \frac{\partial M}{\partial \alpha_v} = \begin{bmatrix} 0 \\ r_{12}X + r_{22}Y\ +\ t_y \\ 0 \end{bmatrix} \qquad \frac{\partial M}{\partial t_y} = \begin{bmatrix} c \\ \alpha_v \\ 0 \end{bmatrix}$$

$$\frac{\partial M}{\partial c} = \begin{bmatrix} r_{21}X + r_{22}Y\ +\ t_y \\ 0 \\ 0 \end{bmatrix} \qquad \frac{\partial M}{\partial v_0} = \begin{bmatrix} 0 \\ r_{31}X + r_{32}Y\ +\ t_z \\ 0 \end{bmatrix} \qquad \frac{\partial M}{\partial t_z} = \begin{bmatrix} u_0 \\ v_0 \\ 1 \end{bmatrix}$$

$$\frac{\partial M}{\partial u_0} = \begin{bmatrix} r_{31}X + r_{32}Y\ +\ t_z \\ 0 \\ 0 \end{bmatrix} \qquad \frac{\partial M}{\partial t_x} = \begin{bmatrix} \alpha_u \\ 0 \\ 0 \end{bmatrix}$$

En vez de calcular las derivadas parciales de las componentes $\mathbf{R_T}$ partiendo de la expresión matricial, utilizamos la fórmula de Rodrigues para expresar estos componentes como un vector de 3 componentes $\vec{w} = (w_1, w_2, w_3)$.

$$\mathrm{Rodr}(\vec{w}) = \mathrm{Rodr}(w_1, w_2, w_3) = I_3 + \sin\left(\sqrt{w^2{}_1 + w^2{}_2 + w^2{}_3}\right)\begin{bmatrix} 0 & \frac{-w_3}{\|w\|} & \frac{w_2}{\|w\|} \\ \frac{w_3}{\|w\|} & 0 & \frac{-w_1}{\|w\|} \\ \frac{-w_2}{\|w\|} & \frac{w_1}{\|w\|} & 0 \end{bmatrix}$$

$$+ \quad (1 - \cos(\sqrt{w^2{}_1 + w^2{}_2 + w^2{}_3}))\begin{bmatrix} 0 & \frac{-w_3}{\|w\|} & \frac{w_2}{\|w\|} \\ \frac{w_3}{\|w\|} & 0 & \frac{-w_1}{\|w\|} \\ \frac{-w_2}{\|w\|} & \frac{w_1}{\|w\|} & 0 \end{bmatrix}^2 = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

Esta fórmula relaciona los componentes de la matriz de rotación $\mathbf{R}_T$ con las tres componentes w_1, w_2, w_3 del vector. Típicamente se expresa el vector $\vec{w}$ como un vector unitario, y consecuentemente se obtiene la relación $\vec{u}=(x_u, y_u, z_u)=\frac{\vec{w}}{\|\vec{w}\|}=(\frac{w_1}{\sqrt{w_1^2+w_2^2+w_3^2}},\frac{w_2}{\sqrt{w_1^2+w_2^2+w_3^2}},\frac{w_3}{\sqrt{w_1^2+w_2^2+w_3^2}})$. Tras aplicar la relación entre las r_{ij} y las w_k mediante la fórmula de Rodrigues se puede proceder a obtener las derivadas parciales que necesitamos proveer al método de minimización de Levenberg-Marquardt. En particular, la derivada parcial para cada una de estas tres componentes w_k para $k \in [1\ldots3]$ viene expresada por:

$$\frac{\partial M}{\partial w_k} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \frac{\partial r_{11}}{\partial w_k} & \frac{\partial r_{12}}{\partial w_k} & \frac{\partial t_x}{\partial w_k} = 0 \\ \frac{\partial r_{21}}{\partial w_k} & \frac{\partial r_{22}}{\partial w_k} & \frac{\partial t_y}{\partial w_k} = 0 \\ \frac{\partial r_{31}}{\partial w_k} & \frac{\partial r_{32}}{\partial w_k} & \frac{\partial t_z}{\partial w_k} = 0 \end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Cada una de las $\frac{\partial r_{ij}}{\partial w_k}$ vienen expuestas a continuación donde $\alpha = \sqrt{w_1^2 + w_2^2 + w_3^2}$:

$$\frac{\partial r_{11}}{\partial w_1} = \sin(\alpha) \cdot \frac{w_1}{\alpha} \cdot \left(\frac{w_1}{\alpha}\right)^2 + (1 - \cos\alpha) \cdot 2 \cdot \left(\frac{\alpha - \frac{w_1^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) \cdot \left(\frac{w_1}{\alpha}\right) + \frac{w_1 \cdot (-\sin\alpha)}{\alpha}$$

$$\frac{\partial r_{11}}{\partial w_2} = \sin(\alpha) \cdot \frac{w_2}{\alpha} \cdot \left(\frac{w_1}{\alpha}\right)^2 + (1 - \cos\alpha) \cdot 2 \cdot \left(\frac{-\frac{w_1 \cdot w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) \cdot \left(\frac{w_1}{\alpha}\right) + \frac{w_2 \cdot (-\sin\alpha)}{\alpha}$$

$$\frac{\partial r_{11}}{\partial w_3} = \sin(\alpha) \cdot \frac{w_3}{\alpha} \cdot \left(\frac{w_1}{\alpha}\right)^2 + (1 - \cos\alpha) \cdot 2 \cdot \left(\frac{-\frac{w_1 \cdot w_3}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) \cdot \left(\frac{w_1}{\alpha}\right) + \frac{w_3 \cdot (-\sin\alpha)}{\alpha}$$

$$\frac{\partial r_{12}}{\partial w_1} = \sin(\alpha) \cdot \frac{w_1}{\alpha} \cdot \left(\frac{w_1}{\alpha} \cdot \frac{w_2}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\alpha - \frac{{w_1}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) \cdot \left(\frac{w_2}{\alpha}\right) + \frac{w_1}{\alpha} \cdot \frac{\frac{-w_2 \cdot w_1}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right] - \left[\left(\frac{\frac{-w_3 \cdot w_1}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) \cdot \sin\alpha + \frac{w_3}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_1}{\alpha} \right]$$

$$\frac{\partial r_{12}}{\partial w_2} = \sin(\alpha) \cdot \frac{w_2}{\alpha} \cdot \left(\frac{w_1}{\alpha} \cdot \frac{w_2}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\frac{-w_1 \cdot w_2}{\alpha} \cdot \frac{w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) + \frac{w_1}{\alpha} \cdot \frac{\alpha - \frac{{w_2}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right] - \left[\left(\frac{\frac{-w_3 \cdot w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2} \right) \cdot \sin\alpha + \frac{w_3}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_2}{\alpha} \right]$$

$$\frac{\partial r_{12}}{\partial w_3} = \sin(\alpha)\cdot\frac{w_3}{\alpha}\cdot\left(\frac{w_1}{\alpha}\cdot\frac{w_2}{\alpha}\right) + (1-\cos\alpha) \cdot\left[\left(\frac{\frac{-w_1\cdot w_3}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\frac{w_2}{\alpha}+\frac{w_1}{\alpha}\cdot\frac{\frac{-w_3\cdot w_2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right] - \left[\left(\frac{\alpha-\frac{{w_3}^2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\sin\alpha + \frac{w_3}{\alpha}\cdot(\cos\alpha)\cdot\frac{w_3}{\alpha}\right]$$

$$\frac{\partial r_{21}}{\partial w_1} = \sin(\alpha)\cdot\frac{w_1}{\alpha}\cdot\left(\frac{w_1}{\alpha}\cdot\frac{w_2}{\alpha}\right) + (1-\cos\alpha) \cdot\left[\left(\frac{\alpha-\frac{{w_1}^2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\frac{w_2}{\alpha}+\frac{w_1}{\alpha}\cdot\frac{\frac{-w_2\cdot w_1}{\alpha}}{w_1^2+w_2^2+w_3^2}\right] + \left[\left(\frac{\frac{-w_3\cdot w_1}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\sin\alpha + \frac{w_3}{\alpha}\cdot(\cos\alpha)\cdot\frac{w_1}{\alpha}\right]$$

$$\frac{\partial r_{21}}{\partial w_2} = \sin(\alpha)\cdot\frac{w_2}{\alpha}\cdot\left(\frac{w_1}{\alpha}\cdot\frac{w_2}{\alpha}\right) + (1-\cos\alpha) \cdot\left[\left(\frac{\frac{-w_1\cdot w_2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\frac{w_2}{\alpha}+\frac{w_1}{\alpha}\cdot\frac{\alpha-\frac{{w_2}^2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right] + \left[\left(\frac{\alpha-\frac{{w_3}^2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\sin\alpha + \frac{w_3}{\alpha}\cdot(\cos\alpha)\cdot\frac{w_2}{\alpha}\right]$$

$$\frac{\partial r_{21}}{\partial w_3} = \sin(\alpha)\cdot\frac{w_3}{\alpha}\cdot\left(\frac{w_1}{\alpha}\cdot\frac{w_2}{\alpha}\right) + (1-\cos\alpha) \cdot\left[\left(\frac{\frac{-w_1\cdot w_3}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\frac{w_2}{\alpha}+\frac{w_1}{\alpha}\cdot\frac{\frac{-w_2\cdot w_3}{\alpha}}{w_1^2+w_2^2+w_3^2}\right] + \left[\left(\frac{\alpha-\frac{{w_3}^2}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\sin\alpha + \frac{w_3}{\alpha}\cdot(\cos\alpha)\cdot\frac{w_3}{\alpha}\right]$$

$$\frac{\partial r_{22}}{\partial w_1} = \sin(\alpha)\cdot\frac{w_1}{\alpha}\cdot\left(\frac{w_2}{\alpha}\right)^2 + (1-\cos\alpha)\cdot 2\cdot\left[\left(\frac{\frac{-w_2\cdot w_1}{\alpha}}{w_1^2+w_2^2+w_3^2}\right)\cdot\frac{w_2}{\alpha}\right] - (\sin\alpha)\cdot\frac{w_1}{\alpha}$$

$$\frac{\partial r_{22}}{\partial w_2} = \sin(\alpha) \cdot \frac{w_2}{\alpha} \cdot \left(\frac{w_2}{\alpha}\right)^2 + (1 - \cos\alpha) \cdot 2 \cdot \left[\left(\frac{\alpha - \frac{{w_2}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \frac{w_2}{\alpha}\right] - (\sin\alpha) \cdot \frac{w_2}{\alpha}$$

$$\frac{\partial r_{22}}{\partial w_3} = \sin(\alpha) \cdot \frac{w_3}{\alpha} \cdot \left(\frac{w_2}{\alpha}\right)^2 + (1 - \cos\alpha) \cdot 2 \cdot \left[\left(\frac{\frac{-w_2 \cdot w_3}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \frac{w_2}{\alpha}\right] - (\sin\alpha) \cdot \frac{w_3}{\alpha}$$

$$\frac{\partial r_{31}}{\partial w_1} = \sin(\alpha) \cdot \frac{w_1}{\alpha} \cdot \left(\frac{w_1}{\alpha} \cdot \frac{w_3}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\alpha - \frac{{w_1}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \left(\frac{w_3}{\alpha}\right) + \frac{w_1}{\alpha} \cdot \frac{\frac{-w_1 \cdot w_3}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right] - \left[\left(\frac{\frac{-w_2 \cdot w_1}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \sin\alpha + \frac{w_2}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_1}{\alpha}\right]$$

$$\frac{\partial r_{31}}{\partial w_2} = \sin(\alpha) \cdot \frac{w_2}{\alpha} \cdot \left(\frac{w_1}{\alpha} \cdot \frac{w_3}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\frac{-w_2 \cdot w_1}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \left(\frac{w_3}{\alpha}\right) + \frac{w_1}{\alpha} \cdot \frac{\frac{-w_3 \cdot w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right] - \left[\left(\frac{\alpha - \frac{{w_2}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \sin\alpha + \frac{w_2}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_2}{\alpha}\right]$$

$$\frac{\partial r_{31}}{\partial w_3} = \sin(\alpha) \cdot \frac{w_3}{\alpha} \cdot \left(\frac{w_1}{\alpha} \cdot \frac{w_3}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\frac{-w_1 \cdot w_3}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \left(\frac{w_3}{\alpha}\right) + \frac{w_1}{\alpha} \cdot \frac{\alpha - \frac{{w_3}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right] - \left[\left(\frac{\frac{-w_2 \cdot w_3}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \sin\alpha + \frac{w_2}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_3}{\alpha}\right]$$

$$\frac{\partial r_{32}}{\partial w_1} = \sin(\alpha) \cdot \frac{w_1}{\alpha} \cdot \left(\frac{w_2}{\alpha} \cdot \frac{w_3}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\frac{-w_2 \cdot w_1}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \left(\frac{w_3}{\alpha}\right) + \frac{w_2}{\alpha} \cdot \frac{\frac{-w_3 \cdot w_1}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right] + \left[\left(\frac{\alpha - \frac{{w_1}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \sin\alpha + \frac{w_1}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_1}{\alpha}\right]$$

$$\frac{\partial r_{32}}{\partial w_2} = \sin(\alpha) \cdot \frac{w_2}{\alpha} \cdot \left(\frac{w_2}{\alpha} \cdot \frac{w_3}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\alpha - \frac{{w_2}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \left(\frac{w_3}{\alpha}\right) + \frac{w_2}{\alpha} \cdot \frac{\frac{-w_3 \cdot w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right] + \left[\left(\frac{\frac{-w_1 \cdot w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \sin\alpha + \frac{w_1}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_2}{\alpha}\right]$$

$$\frac{\partial r_{32}}{\partial w_3} = \sin(\alpha) \cdot \frac{w_3}{\alpha} \cdot \left(\frac{w_2}{\alpha} \cdot \frac{w_3}{\alpha}\right) + (1 - \cos\alpha) \cdot \left[\left(\frac{\frac{-w_3 \cdot w_2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \left(\frac{w_3}{\alpha}\right) + \frac{w_2}{\alpha} \cdot \frac{\alpha - \frac{{w_3}^2}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right] + \left[\left(\frac{\frac{-w_1 \cdot w_3}{\alpha}}{w_1^2 + w_2^2 + w_3^2}\right) \cdot \sin\alpha + \frac{w_1}{\alpha} \cdot (\cos\alpha) \cdot \frac{w_3}{\alpha}\right]$$

BIBLIOGRAFÍA

[Abt70] Abt, C. 1970. Serious Games. Viking Press, New York.

[Aco06] Acosta R, Catalá A, Esteve J.M., Mocholí J.A., Jaén J. "Ecoology: un sistema para aprender jugando". Revista Novática nº 182, pp. 63-67, julio-agosto 2006 (ISSN 0211-2124).

[Ale00] Aleinikov A., Kackmeister S., Koenig R. (Eds.). "Creating creativity: 101 definitions". Midland, MI: Alden B. Dow Creativity Center, Northwoods University, 2000.

[Ama83] Amabile T. M. "The social psychology of creativity". New York: Springer-Verlag, 1983.

[Artw3] ARToolkit website: http://www.hitl.washington.edu/artoolkit/

[Baa03] Baader F., Calvanese D., McGuinness D. L., Nardi D., Patel-Schneider P. F. (Eds.). "The Description Logic Handbook: Theory, Implementation, and Applications". Cambridge University Press, 2003.

[Bao06] Bao J., Caragea D., Honavar V. "A Distributed Tableau Algorithm for Package-based Description Logics". In 2nd International Workshop On Context Representation And Reasoning (CRR 2006), ECAI 2006 , 2006.

[Bal07] Baldoni R., Beraldi R., Quema V., Querzoni L., Tucci-Piergiovanni S. "TERA: topic-based event routing for peer-to-peer architectures". In Proc. of the 2007 inaugural international Conference on Distributed Event-Based Systems (Toronto, Ontario, Canada, June 20 - 22, 2007). DEBS '07, vol. 233. ACM, New York, NY, 2-13, 2007.

[Ber01] Berners-Lee T., Hendler J., Lassila O. "The Semantic Web," Scientific Am., pp. 34-43, May 2001.

[Bor06] Borgida A., Serafini L. "Distributed description logics: Directed domain correspondences in federated information sources". In CoopIS/DOA/ODBASE, pp. 36-53, 2002.

[Bra77] Brachman J.R. "What's in a concept: Structural foundations for semantic networks". Int' Journal of Man.Machine Studies, vol. 9, no. 2, pp. 127-152, 1977.

[Bra78] Brachman J.R. "Structured inheritance networks". InW.A. Woods and R.J. Brachman editors, Research in Natural Language Understanding, Quarterly Progress Report no. 1, BBN Report no. 3742, pp. 36-78, Bolt, Beranek and Newman Inc., Cambridge, MA, 1978.

[Bra07] Bratt S. "Semantic Web, and Other W3C Technologies to Watch", Talks at W3C, January 2007. Disponible en línea en: http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/0130-sb-W3CTechSemWeb.pdf

[Bro89] Brown J. S., Collins A., Dugiud P. "Situated cognition and the culture of learning. Educational Researcher vol. 18, no. 1, pp. 32-42, 1989.

[Bro02] Brown B., Larson R.W., Saraswathi, T.S. "The worlds' youth: adolescence in eight regions of the world". Cambridge University Press, 2002.

[Bru60] Bruner J. "The Process of Education". Cambridge, MA: Harvard University Press, 1960.

[Car00] Carzaniga A., Rosenblum D.S., Wolf A.L. "Achieving Expressiveness and Scalability in an Internet-Scale Event Notification Service". ACM Symp. on Principles of Distributed Computing, Portland OR. July, 2000.

[Com07] "Estrategia Nacional de Ciencia y Tecnología", Comisión Interministerial de Ciencia y Tecnología, Fundación Española para la Ciencia y Tecnología (FECYT), Ministerio de Educación y Ciencia, 2007. Disponible en línea en: http://web.micinn.es/05_Investigacion/01@APoliticas/02@Encyt/Encyt.pdf

[Csi88] Csikszentmihalyi M., Csikszentmihalyi I., “Optimal Experience. Psychological Studies of Flow in Consciousness”. Cambridge University Press, 1988.

[DeF07] De Freitas S. “Learning in Immersive Worlds: a review of game based learning”, Joint Information Systems Committee (JISC), 2007.

[Dec85] Deci E.L., Ryan R. M. “Intrinsic Motivation and Self-Determination in Human Behaviour. Plenum Press, 1985.

[Del02] Delle Fave A., Bassi M., Massimini F., “Quality of experience and daily social context of Italian adolescents”. In A.L. Comunian, U.P. Gielen eds. It’s all about relationships, pp. 159-172, 2002.

[Dew63] Dewey J. “Experience and Education”. New York: Collier, 1963.

[DLw3] Resources on Description Logics Website, http://dl.kr.org/

[Ell06] Ellis H., Heppell S., Kirriemuir J., Krotoski A., McFarlane A. “Unlimited Learning: The role of computer and video games in the learning landscape”. ELSPA: Entertainment and Leisure Software Publishers Association, 2006.

[Eug03] Eugster P.T., Felber P. A., Guerraoui R., Kermarrec A. “The many faces of publish/subscribe”. ACM Comput. Surv. vol. 35, no. 2, pp. 114-131, 2003.

[Fac04] Facer K., Joiner R., Stanton D., Reid J., Hull R., Kirk D. “Savannah: mobile gaming and learning”, Journal of Computer Assisted Learning vol. 20, pp. 399-409, 2004.

[Fia04] Fiala M. “ARTag Revision 1. A Fiducial Marker System Using Digital Techniques”, National Research Council Canada, NRC 47419, November 24, 2004.

[Fia05] Fiala M., Shu C. “Fully Automatic Camera Calibration Using Self-Identifying Calibration Targets”, National Research Council Canada, NRC 48306, November 2005.

[For73] Forney G.D. Jr. “The Viterbi Algorithm”. Proceedings of IEEE, vol. 61, no. 3, 1973.

[Gab08] Gabrielli S., Belluti S., Jameson A. "A Single-User Tabletop Card Game System for Older Persons: General Lessons Learned from an In-Situ Study". IEEE Tabletops and Interactives Surfaces, IEEE International Workshop on Horizontal Interactive Human-Computer Systems, pp.91-94, 2008.

[Gar93] Gardner H. "Creating minds". New York: Basic Books, 1993.

[Gee05] Gee J.P. "Learning by Design: good video games as learning machines". E-Learning, vol. 2, no.1, 2005.

[Glo03] Glover I., Grant P. "Digital Communications", Prentice-Hall, 2003.

[Gra06] Grant L. "Space Mission: Ice Moon". Research Report. FutureLab, UK, 2006.

[Gro07] Gros B. "The Design of Learning Environments Using Videogames in Formal Education". DIGITEL, pp. 19-24, The First IEEE International Workshop on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL'07), 2007.

[Gro04] Gros B., Grup F9. "Pantallas, juegos y alfabetización digital". Comunicación y Pedagogía, vol. 208, pp. 25-27, 2004.

[Hal07] Halaschek-Wiener C., Hendler J. "Toward expressive syndication on the web". In Proc. of the 16th int. Conf. on World Wide Web. ACM, pp. 727-736. May 2007.

[Han05] Han J. Y. "Low-cost multi-touch sensing through frustrated total internal reflection". In *Proceedings of the 18th Annual ACM Symposium on User interface Software and Technology* (Seattle, WA, USA, October 23 - 26, 2005). UIST '05. ACM, New York, NY, 115-118, 2005.

[Har95] Hartley R. I. "In defence of the 8-point algorithm". *In Proceedings of the Fifth international Conference on Computer Vision* (June 20 - 23, 1995). ICCV. IEEE Computer Society, Washington, DC, pp. 1064, 1995.

[Har03] Hartley R., Zisserman A. "Multiple View Geometry in Computer Vision" (2rd Edition). 2ª edición. Cambridge University Press. ISBN: 0201398559. 2003.

[Hil73] Hilgard E.R., Bower G.H. “Teorías del aprendizaje”. México, Trillas, 1973.

[Hor08] Hornecker E. “I don’t understand it either, but it is cool”- Visitor Interactions with a Multi-touch Table in a Museum. IEEE Tabletops and Interactives Surfaces, IEEE International Workshop on Horizontal Interactive Human-Computer Systems, pp. 121-129, 2008.

[Hor04] Horridge M., Knublauch H., Rector A., Stevens R., Wroe C. “A practical guide to building OWL ontologies using the Protégé-PWL Plugin and CO-ODE Tools Edition 1.0”, The University of Manchester, August 27, 2004. Available online at http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf

[isew3] InterSense Web site: http://www.isense.com/products.aspx?id=45

[Jor03] Jordà S. “Interactive Music Systems for Everyone: Exploring Visual Feedback as a Way for Creating More Intuitive, Efficient and Learnable Instruments”. Proceedings of the Stockholm Music Acoustics Conference (SMAC 03), Stockholm (Sweden), 2003.

[Joy92] Joyce B., Wil M., Showers B. “Models of Teaching”, Boston: Allyn and Bacon, 1992.

[Kal07] Kaltenbrunner M., Bencina R. “reacTIVision: a computer-vision framework for table-based tangible interaction”. *In Proceedings of the 1st international Conference on Tangible and Embedded interaction* (Baton Rouge, Louisiana, February 15 - 17, 2007). TEI '07. ACM, New York, NY, pp. 69-74, 2007.

[Kat99] Kato H., Billinghurst M. “Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System”. In Proceedings of the 2nd IEEE and ACM international Workshop on Augmented Reality. IWAR. IEEE Computer Society, Washington, DC, pp. 85-94, 1999.

[Kha07] Khandelwal M., Mazalek A. “Teaching table: a tangible mentor for pre-k math education”. In Proceedings of the 1st international Conference on Tangible and Embedded interaction (Baton Rouge, Louisiana, February 15 - 17, 2007). TEI '07. ACM, New York, NY, pp. 191-194, 2007.

[Keb08] Kebritchi M., Hirumi A. "Examining the pedagogical foundations of modern educational computer games". Comput. Educ. vol. 51, no. 4 (Dec. 2008), pp. 1729-1743, 2008.

[Kin02] Kindley R. "The power of simulation-based e-learning", (SIMBEL) The E-learning Developers' Journal. pp. 1-8, September 17th, 2002.

[Kol94] Kolb D. A. "Experiential Learning: Experience as the source of learning and development". Englewood CLiffs, NJ: Prentice-Hall, 1984.

[Kur02] Kurtev, I., Bézivin, J., Aksit, M.: "Technological Spaces: An Initial Appraisal". Int. Federated Conf. (DOA, ODBASE, CoopIS), Industrial track, Irvine, 2002.

[Lan99] Langton C. "Tutorial 12. Convolutional Coding and Decoding Made Easy", Charan Langton Ed. 1999. Disponible en línea en http://www.complextoreal.com/chapters/convo.pdf

[Las05] Lassila O., Khushra D. "Contextualizing Applications via Semantic Middleware". In Proc. of the int. Conf. on Mobile and Ubiquitous Systems: Networking and Services, IEEE Computer Society, pp. 183-191, July 2005.

[Legw3] Lego MINDSTORMS NXT website: http://mindstorms.lego.com/

[Li04] Li H., Jiang G. "Semantic message oriented middleware for publish/subscribe networks", Proc. SPIE Int. Soc. Opt. Eng. vol. 5403, no. 1, pp. 124-133, 2004.

[Liu03] Liu Y., Plale B. "Survey of publish subscribe event systems", Technical Report TR574, Indiana University, May 2003.

[Lop02] López de Ipiña, D., Mendonça P.R.S., Hopper A. "TRIP: a Low-Cost Vision-Based Location System for Ubiquitous Computing. Personal and Ubiquitous Computing Journal", Springer, vol. 6, no. 3, pp. 206–219, May 2002.

[Mag04] Magerkurth C., Memisoglu M., Engelke T., Streitz N. A. "Towards the Next Generation of Tabletop Gaming Experiences". In Procs. Of Graphics Interface, vol. 62. pp. 73-80, 2004.

[Mag03] Magerkurth C., Stenzel R., Prante T. “STARS - a ubiquitous computing platform for computer augmented tabletop games”. In Proc. of the Fifth International Conference on Ubiquitous Computing (UBICOMP), 2003.

[Manw3] The Manchester OWL Syntax reference: http://www.co-ode.org/resources/reference/manchester_syntax/

[Maz06] Mazalek A., Reynolds M., Davenport G. "TViews: An Extensible Architecture for Multiuser Digital Media Tables," *Computer Graphics and Applications, IEEE* , vol.26, no.5, pp.47-55, Sept.-Oct. 2006.

[Maz07] Mazalek A., Mironer B., O'Rear E., Devender D. V. “The TViews table role-playing game”. In Proc. 4th International Symposium on Pervasive Gaming Applications (2007), Shaker Verlag, pp. 127-134, 2007.

[McF02] McFarlane A., Sparrowhawk A., Heald Y. “Report on the educational use of games”. Teem: Teachers Evaluating Educational Multimedia, 2002.

[Mic05] Michael D., Chen S. “Serious Games: Games that Educate, Train, and Inform”. Course Technology PTR, 2005.

[Min07] “Las cifras de la Educación en España. Estadísticas e indicadores. Edición 2008”, Estadísticas de la Educación, Ministerio de Educación, Política Social y Deporte, 11 Octubre, 2007. Disponible en línea en: http://www.mepsyd.es/mecd/jsp/plantilla.jsp?id=3131&area=estadisticas&contenido=/estadisticas/educativas/cee/2007A/cee-2007A.html

[Monw3] The Monkey Wrench Conspiracy website: http://www.games2train.com/site/html/tutor.html

[Ode76] Odenwalder J.O. “Error Control Coding Handbook”, Linkabit Corp., Sandiego, California, July 15, 1976.

[Olgw3] Open Life Grid website, http://openlifegrid.com/

[OSiw3] The Open Simulator (OpenSim) website, http://opensimulator.org

[Osgw3] OSGrid: the Open Source Metaverse website http://osgrid.org/

[owlapi] The OWL-API Website: http://owlapi.sourceforge.net/

[OWLw3] OWL Working Group at W3C. Web Ontology Language. http://www.w3.org/2004/OWL/

[OWLSw3] OWL Working Group. OWL Web Ontology Language. Semantics and Abstract Syntax. http://www.w3.org/TR/owl-semantics/

[Pan07] Pan J. Z., Horrocks I. "RDFS(FA): Connecting RDF(S) and OWL DL," Knowledge and Data Engineering, IEEE Transactions on , vol.19, no.2, pp.192-206, Feb. 2007.

[Pap93] Papert S. "The Children'sMachine: Rethinking school in the Age of the Computer". New York: Basic Books, 1993.

[Peaw3] Peace Maker Game website: http://www.peacemakergame.com/

[Pelw3] Pellet: The Open Source OWL DL Reasoner, http://clarkparsia.com/pellet/

[Per08] Pérez De Pablos, S. "Es preocupante que España tenga un 30% de fracaso escolar", Entrevista a Fernando Reimers, Catedrático de Educación en la Universidad de Harvard. El País (Edición Impresa), Sección Educación, Madrid, 17 Noviembre 2008. También disponible en línea: http://www.elpais.com/articulo/educacion/preocupante/Espana/tenga/fracaso/escolar/elpepiedu/20081117elpepiedu_1/Tes/

[Pet03] Petrovic M., Burcea I., Jacobsen H.-A. "S-ToPSS - a semantic publish/subscribe system". In Proc of Conf. on Very Large Databases, pp. 1101-1004, Sept. 2003.

[Pet05] Petrovic M., Liu H., Jacobsen H. "G-ToPSS: fast filtering of graph-based metadata". In Proc. of the 14th int. Conf. on World Wide Web. ACM, pp. 539-547, 2005.

[Pia67] Piaget J. "Six psycchological studies". New York: Vintage books, 1967.

[Picw3] PicoCricket website: http://www.picocricket.com/

[Pie02] Pietzuch P. R., Bacon J. "Hermes: A Distributed Event-Based Middleware Architecture". In Proc. of the 22nd int. Conf. on Distributed Computing Systems. ICDCSW. IEEE Computer Society, Washington, DC, pp. 611-618, 2002.

[Pip06] Piper A. M., O'Brien E., Ringel Morris M., Winograd T. “SIDES: A Cooperative Tabletop Computer Game for Social Skills Development”. In Proceedings CSCW 2006. pp. 1-10, 2006.

[Pis07] “PISA 2006: Science Competencies for Tomorrow’s World”, Programme for International Student Assesment (PISA), Organization for Economic Co-operation and development (OECD), OECD Publishing, 14 Dec. 2007. Disponible en línea en: http://www.pisa.oecd.org/

[Prow3] Protégé Website: http://protege.stanford.edu/

[Rek00] Rekimoto J., Ayatsuka Y. “CyberCode: designing augmented reality environments with visual tags”. In *Proceedings of DARE 2000 on Designing Augmented Reality Environments* (Elsinore, Denmark). DARE '00. ACM, New York, NY, pp. 1-10, 2000.

[Res88] Resnick M., Ocko S., Papert S. “LEGO, Logo, and Design,” Children’s Environments Quarterly 5, no. 4, pp. 14–18, 1988.

[Res96] Resnick M., Martin F., Sargent R., Silverman B. "Programmable Bricks: Toys to Think With". IBM Systems Journal vol. 35, pp. 443-452, 1996.

[Res98] Resnick M. “Technologies for Lifelong Kindergarten”. Educational Technology Research and Development, vol. 46, no. 4, 1998.

[Res02] Resnick M. “Rethinking Learning in the Digital Age”. In The Global Information Technology Report: Readiness for the Networked World, edited by G. Kirkman. Oxford University Press, 2002.

[Robw3] RoboCup website, http://www.robocup.org/

[Row01] Rowstron A., Kermarrec A-M., Castro M., Druschel P. “SCRIBE: The design of a larg-scale event notification infrastructure”. In Networked Group Communication, pp. 30-43, 2001.

[Sca02] Scarlatos, L. “An application of tangible interfaces in collaborative learning environments”. In *ACM SIGGRAPH 2002 Conference Abstracts and Applications* (San Antonio, Texas, July 21 - 26, 2002). SIGGRAPH '02. ACM, New York, NY, pp. 125-126, 2002.

[Secw3] Second Life Web site: http://secondlife.com/

[Ser05] Serafini L., Tamilin A. "DRAGO: Distributed reasoning architecture for the semantic web". In: Proc. of the Second European Semantic Web Conference (ESWC'05), 2005.

[Sha06] Shadbolt N., Berners-Lee T., Hall W. "The Semantic Web Revisited". *IEEE Intelligent Systems*, vol. 21, no. 3. pp. 96-101, 2006 (ISSN 1541-1672).

[Sir07] Sirin E., Parsia B., Cuenca B., Kalyanpur A., Katz Y. "Pellet: A practical OWL-DL reasoned". J. of Web Semantics, vol. 5, no. 2, 2007.

[Ski53] Skinner B. F. "Science and Human Behavior". Macmillan Free Press, 1953.

[Slu04] Sluis R. J., Weevers I., van Schijndel C. H., Kolos-Mazuryk L., Fitrianie S., Martens J. B. "Read-It: five-to-seven-year-old children learn to read in a tabletop environment". In Proceedings of the 2004 Conference on interaction Design and Children: Building A Community IDC '04. ACM, New York, NY, pp. 73-80, 2004.

[Ste99] Sternberg J.S. et al. "Handbook of Creativity", Robert. J. Stenberg Ed., Cambridge University Press, 1999, (ISBN 9780521576048).

[Stuw3] Studierstube Tracker Website: http://studierstube.icg.tu-graz.ac.at/handheld_ar/stbtracker.php

[Swe91] Sweeney P. "Error Control Coding. An Introduction". Prentice-Hall, 1991.

[Tan03] Tanenbaum A. S. "Computer Networks", Prentice-Hall, 2003.

[Tho13] Thorndike E. L. "The psychology of learning". New York: New York Teacher's College, 1913.

[Tre96] Treffinger D. J., Isaksen S. G., Dorval K. B. "Climate for creativity and innovation: Educational implications". Sarasota, FL: Center for Creative Learning.

[Tre02] Treffinger D. J., Young G. C., Selby E. C., Shepardson C. "Assessing creativity: A guide for educators" (RM02170). Storrs, CT: The National Research Center on the Gifted and Talented, University of Connecticut, 2002.

[Tsa06] Tsarkov D., Horrocks I. “FaCT++ Description Logic Reasoner: System Description”. In Proc. of the Int. Joint Conf. on Automated Reasoning, vol. 4130 of LNCS in Artificial Intelligence, Springer, pp. 292-297, 2006.

[Usc96] Uschold M., Gruninger M. “Ontologies: principles, methods, and applications”. Knowledge Engineering Review, vol. 11, no. 2, pp. 93-155, 1996.

[Usc03] Uschold M. et al. “A semantic infosphere”. In D. Fensel et al, editor, Proc. ISWC 2003.

[Ver03] Verma S., Larsson R.W. “Examining Adolescent Leisure Time Across Cultures: Developmental Opportunities and Risks”. New Directions in Child and Adolescent Development Series, 2003.

[Vit67] Viterbi A., "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," IEEE Transactions on Information Theory, vol.13, no.2, pp. 260-269, Apr 1967.

[Vyg78] Vygotsky L.S. “Mind in Society: The development of Higher Psychological Processes”. Harvard University Press, 1978.

[Wag07] Wagner, D., Schmalstieg, D. “ARToolKitPlus for Pose Tracking on MobileDevices”, In proceedings of 12th Computer Vision Winter Workshop (CVWW'07), February 2007.

[Wan04] Wang J., Jin B., Li J. “An ontology-based publish/subscribe system”. In Proc. of the 5th ACM/IFIP/USENIX int. Conf. on Middleware. vol. 78. Springer-Verlag, pp. 232-253, Oct. 2004.

[Wan07] Wang Y., Bao J., Haase P., Qi G. "Evaluating Formalisms for Modular Ontologies in Distributed Information Systems", In: Web Reasoning and Rule Systems, pp. 178-193, Springer, 2007.

[Wat13] Watson J. B. “Psychology as the behaviorist views it”. Psychological Review, vol. 20, pp. 158-177, 1913.

[Zha99] Zhang Z. “A Flexible New Tecnique for Camera Calibration”. Informe técnico MSR-TR-98-71, Marzo 1999.

[Zhu07] Zhu Y., Hu Y. "Ferry: A P2P-Based Architecture for Content-Based Publish/Subscribe Services," IEEE Transactions on Parallel and Distributed Systems, vol. 18, no. 5, pp. 672-685, May, 2007.

[Zhu01] Zhuang S. Q., Zhao B. Y., Joseph A. D., Katz, R. H., Kubiatowicz J. D. “Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination”. In Proceedings of the 11th international Workshop on Network and Operating Systems Support For Digital Audio and Video (Port Jefferson, New York, United States). NOSSDAV '01. ACM, New York, NY, 11-20, 2001.