Desarrollo de una Interfaz Tangible para el Aprendizaje Basado en Juegos de Forma Natural e Innata

Proyecto Fin de Carrera

Autor D. Carlos A. González Santiago

Tutor Dr. D. David Díez Cebollero

Universidad Carlos III de Madrid

Versión del Documento 1.11
27 de Octubre del 2011

2011

Título: Desarrollo de una interfaz tangible para el aprendizaje basado en juegos de forma natural e innata.

Autor: D. Carlos A. González Santiago.

Director: Dr. D. David Díez Cebollero.

EL TRIBUNAL

Presidente: Dra. Dña. Paloma Díaz Pérez

Vocal: Dr. D. Alejandro Calderón Mateos

Secretario: Dr. D. Ignacio Aedo Cuevas

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

El proyecto ha sido realizado dentro del proyecto URTHEY [1], aportando todos los recursos necesarios para el desarrollo del proyecto.

La perseverancia *"es la actitud de ser firme en alcanzar un objetivo, en momentos que uno se propone llegar a un final definido por el mismo".* Esta es la mejor palabra que puede definir el largo camino hasta la consecución de la ingeniería. A lo largo de este camino es mucha y variada la gente que de una u otra forma me han ayudado a conseguir el objetivo.

Muchos años han pasado desde que empecé a formar parte del grupo de gente del laboratorio, gente que ha estado David, Juanlu, María, gente que está como Darío y Victor, todos ellos me han ayudado a encontrarme a gusto y tener las ganas de ir cada día a trabajar.

Otros tantos compañeros de universidad entre los que quiero destacar y agradecer a Prieto, con el que empecé la aventura de Leganés, Sergio y Jorge, que fueron las personas que encontré al llegar a Leganés.

Mis amigos, Victorina, Püppy, Cepe, Betula, Nacha, Antonia y Juana, siempre me habéis ayudado a reírme de los problemas a quitarle importancia y a seguir adelante… Gracias.

En estos dos últimos años conocía dos personas muy especiales que han sido un gran apoyo en los momentos duros de este último año, Sara y Rosa, "las niñas". Gracias, habéis sido una gran piedra donde apoyarse, siempre habéis estado allí y nunca me habéis faltado cuando os he necesitado

Por último me gustaría agradecer a dos personas, que han aguantado la mayor parte del camino y que siempre han buscado lo mejor para mí. David, gracias por la oportunidad que me diste de formar parte vuestra, me has enseñado mucho desde el punto de vista profesional y personal.

Diana, empezamos juntos la aventura y la hemos acabado juntos, gracias por estar ahí, por ayudarme, animarme y apoyarme durante estos años, este logro también es tuyo.

Resumen

El modelo mental se define como *"una representación de algo que define una lógica y estimación creíble de cómo una cosa es construida o cómo funciona"*. Muchas veces a la hora de diseñar tecnología no se tiene en cuenta a los usuarios inexpertos, desde el punto de vista tecnológico, lo cual, puede dificultar la incorporación de los mismos a la tecnología. Esto puede ser debido a que el modelo mental con el que se han creado los objetos de interacción. Los diseñadores, por lo tanto, deben tratar de que el modelo mental con el cual diseñan los dispositivos se asemeje al de los usuarios a los que va dirigido. Este concepto es lo que se denomina la distancia entre la funcionalidad del dispositivo y lo que el usuario quiere hacer, o "distancia semántica". La distancia semántica se define como *"la distancia entre lo que un usuario quiere hacer y el significado de un elemento de la interfaz".* Una última cosa a tener en cuenta es *"la distancia entre la apariencia física del objeto y cuál es su funcionalidad"*, esto es denominado la "distancia articular".

Un ejemplo de esta situación puede encontrarse en el uso de juegos educativos con niños. Para ellos hemos eliminado los diferentes aparatos tecnológicos de la interfaz, ofreciendo un entorno con diferentes sensores capaces de detectar los movimientos definidos para interactuar con nuestro sistema. Las acciones reconocidas son soltar un objeto, recogerlo y deslizarlo por la superficie de reconocimiento. Estos movimientos son naturales e innatos a las personas y no requieren de aprendizaje. Todo ello se ha evaluado implementando un sistema externo al entorno con a través del cual niños de entre 3 y 4 años sin conocimiento tecnológico puedan participar en diferentes juegos.

Por último y a modo de resumen este proyecto aporta la eliminación de la distancia entre el usuario y los elementos interactivos de las interfaces que ofrecen los sistemas diseñando un sistema con el paradigma invisible computing y dotándolo del estilo de interacción tangible. Otra aportación ofrecida en la solución es la identificación de diferentes propiedades de los objetos físicos a través de la creación de un modelo de información y su instanciación en un conjunto de etiquetas que son asociadas a los objetos. La conexión entre el entorno y cualquier otro sistema a través de un protocolo de comunicación es otra de las características que tiene el entorno desarrollado. Por último, se ofrece la posibilidad de trabajar colaborativamente en el entorno detectando más de un objeto al mismo tiempo.

Palabras clave: HCI, Interacción, Usuario, Tangible, Inexperto, Modelo Mental.

Abstract

The mental model is defined as "a representation of something that defines a logical and credible estimate of how something is constructed or how it works". Many times when technology is designing, we don't take into account to inexperienced users, from technological point of view, which may hinder the incorporation of the user to the technology. This issue is, maybe, caused by the mental model used to create the interaction object. Therefore, designers should seek that the mental model which is used to design devices gets resembled users who are directed. This concept is what is called distance semantics, it means, the distance between the functionality of the device and what the user wants to do. The semantic distance is defined as "the distance between what a user wants to do and the meaning of an interface". At last, Designers take into the articulatory distance defined as *"the distance between the physical appearance of an interface element and what it actually means"*

An example of this situation can be found in the use of educational computer games with children. For them, we have removed the different technological appliances of the interface. We give them an environment with different sensors is able to detecting defined movements to interact with our system. The recognized actions are: releasing an object, pick it up and slide it over the surface recognition. These movements are natural and innate to people and they don't require learning. External system has been implemented to make evaluation that, children aged between 3 and 4 without technological knowledge could participate in different games.

In conclusion, this project removes the distance between the user and the interactive elements of a user interface, designing a system with the invisible computing paradigm and the tangible interaction style. Another contribution in this project is the identification of different physical objects properties through the creation of an information model and its instantiation in a set of tags that are associated with objects. Another feature is that the environment and any other system is connected through a communication protocol. Finally, the environment offers the possibility of working collaboratively detecting more than one object at the same time.

Keywords: HCI, Interaction, User, Tangible, Inexpert, Mental Model

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Glosario de términos

IUT (TUI): Interfaz de Usuario Tangible.

IU (UI): Interfaz de Usuario.

HCI: Interacción Persona-Ordenador.

Actuador: dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un proceso automatizado.

MCRpd: Model-control-representation (physical and digital).

ECG: Educational Computer Game.

GUI: Graphical User Interface.

URLs: Localizador uniforme de recursos.

QR-Code: Código de Barras de rápida respuesta.

I/O (E/S): Entrada/Salida.

Checksum: esquema simple de detección de errores, donde cada mensaje transmitido es acompañado con un valor numérico basado en el número de grupo de bits del mensaje.

1 Introducción

El Education Computer Game (ECG), Juegos de Ordenador Educativos, se define como *"el proceso de crear una actividad o ejercicio recreativo sometida a reglas en el cual se puede ganar o perder, a fin de soportar el proceso de aprendizaje de forma que sea motivador, estimulante y divertido"* [1] [2]. Después de hacer una evaluación de un ECG, centrado en la temática de las matemáticas, Kamran Sedighian and Andishe Sedighia [3] observaron ciertos elementos para poder satisfacer las necesidades de aprendizaje de los niños y sus motivaciones. Algunos de esos elementos son: el aprendizaje con sentido, establecimiento de objetivos, el éxito, los desafíos, la interacción, la comunicación, la asociación a través del placer, la atracción y la estimulación de los sentidos. Todo ello hace que tanto el diseño del juego como de la plataforma utilizada para su despliegue sean elementos esenciales. En particular, el presente trabajo se centra en la proporción de una interfaz sencilla a través de la cual los usuarios inexpertos se puedan desenvolver sin problemas.

En los siguientes apartados de este capítulo se profundizará en el planteamiento del problema, ofreciendo una explicación del problema encontrado, a partir de dicho problema se establecerán los objetivos que pretenden alcanzar con este proyecto. También se explicarán las fases de desarrollo por las que pasará el proyecto y acabando con la identificación de los recursos necesarios para el desarrollo del mismo y la estructura de la memoria.

1.1 Planteamiento del Problema

A la hora de diseñar un ECG, el principal foco de atención está centrado en los aspectos educativos, pero también hay otros aspectos a tener en cuenta durante la realización del diseño. El proyecto aborda uno de esos factores: la interacción del usuario con el ECG. Cuando el usuario quiere hacer uso de objetos en el mundo real para interactuar con el ECG, se formula preguntas sobre cuál es la funcionalidad de los objetos utilizados y evalúa su potencial para poder ayudarle en la resolución de la tarea, es decir, el usuario intenta entender que cosas hace y cómo las hace. Este proceso implica la creación, por parte del jugador, de un modelo mental sobre cuáles son las acciones que proporciona el objeto y cómo se deben ejecutar esas acciones. Algunos objetos son fáciles de utilizar, intuitivos, y no presentan dificultades al usuario incluso para usuarios que nunca antes los hayan visto ni utilizado. Estos objetos tienen un *affordance* que conduce a su facilidad de uso. El término *affordance* se puede entender como la relación existente entre un objeto y un usuario a través de la cual se puede intuir la interacción que ofrece dicho objeto. Otros tipo de objetos son, por su propia naturaleza, más complejos, menos obvios a la hora de entender que es lo que hacen, es decir, son menos transparentes en términos de cómo funcionan. Esto implica que requieren que el usuario asuma muchas cosas sobre su funcionalidad. Por esta razón es una buena práctica a la hora de diseñar la interacción del ECG incorporar objetos a los sistemas diseñados cuyo *affordance* se asemeje lo más posible al modelo mental que el usuario tiene del objeto. El modelo mental se define como *"una representación de algo que define una lógica y estimación creíble de cómo una cosa es construida o cómo funciona"* [4] Los diseñadores, por lo tanto, deben tratar de que el modelo mental con el cual diseñan los dispositivos se asemeje al de los usuarios a los que va dirigido, tal y como muestra la Ilustración 1. Este concepto es lo que se denomina la distancia entre la funcionalidad del dispositivo y lo que el usuario quiere hacer, o

"distancia semántica". La distancia semántica se define como *"la distancia entre lo que un usuario quiere hacer y el significado de un elemento de la interfaz"* [4]. Una última cosa a tener en cuenta es *"la distancia entre la apariencia física del objeto y cuál es su funcionalidad"*, esto es denominado la "distancia articular" [4].

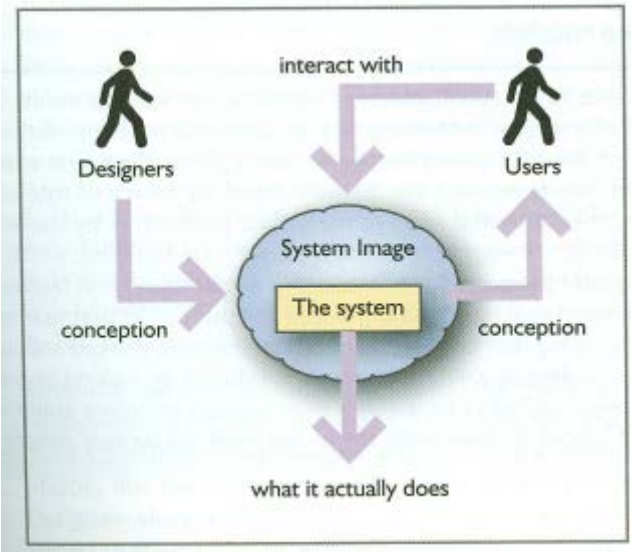


Ilustración 1.Modelo mental [5]

Como ejemplo del problema, imaginemos un juego de reconocimiento de figuras geométricas o de colores y un grupo de niños con edades comprendidas entre 3 y 5 años. Durante el desarrollo del juego pueden ocurrir dos cosas, que los niños cumplan los objetivos o no. En caso de no cumplirlos puede ser debido a falta de conocimiento, el cual se subsanaría empleando diferentes técnicas de aprendizaje, o a dificultades que se les presenten a la hora de interactuar con el sistema. En este caso se les debe proporcionar una interfaz sencilla que ofrezca una interacción natural.

1.2 Objetivos

El objetivo principal del proyecto de fin de carrera es el desarrollo de mecanismos que permitan una interacción natural e innata para la ejecución de juegos orientados al aprendizaje de tareas sencillas, como el reconocimiento de figuras geométricas, colores o los riesgos que pueden tener los objetos, y su aplicación en interfaces para poder salvar las barreras, comentadas en el apartado introductorio del capítulo, que proporcionan los dispositivos a la hora de interactuar con ellos. A fin de alcanzar este objetivo principal se definen los siguientes objetivos secundarios:

a) La representación e identificación de la naturaleza física de los objetos tales como el color, forma, material del que están hechos.
b) La identificación de las propiedades de los objetos relacionadas con su uso y utilidad tales como su naturaleza, el posible tipo de contenido del objeto, el nivel de riesgo existente al utilizar el objeto, etc...
c) La definición de un modelo de datos que permita representar todas las propiedades comentadas en los apartados anteriores.

d) Poder utilizar más de un objeto físico al mismo tiempo permitiendo la resolución cooperativa del ECG y el aumento de las capacidades interactivas del mismo.
e) Uso de objetos físicos como elementos interactivos dentro del sistema sin que requiera un proceso de aprendizaje en la utilización de los mismos, easeof-use, por parte de los usuarios del sistema.

1.3 Fases del desarrollo

La satisfactoria consecución de todo proyecto de desarrollo de un sistema implica la utilización de una metodología de trabajo concreta, la cual debe adaptarse además a las características del proyecto. Con estos mismos objetivos, se ha definido inicialmente la metodología sugerida por Jay Nunamaker Jr. en *"Systems Development in Information Systems Research"* [6]. En su propuesta para el desarrollo de un sistema utilizando un método de investigación destaca cinco fases:

- **Construcción de un modelo conceptual**: en esta fase del desarrollo se establecerá la cuestión de investigación. Después se investigará sobre las funcionalidades del sistema y sus requisitos entendiendo los procedimientos/procesos de construcción del mismo. Además se incluirá un estudio sobre la disciplina en la que se encuentra enmarcado el proyecto.
- **Desarrollo de una arquitectura del sistema**: se diseña una arquitectura que haga el sistema lo más extensible y modular posible. Para el proyecto se utilizara el modelo MCRpd. Además hay que definir la funcionalidad de los componentes del sistema y la relación entre ellos.
- **Análisis y diseño del sistema**: diseño del modelo de datos, diseño de clases diagramas de actividad, todo aquello que defina los procesos a llevar a cabo por el sistema.
- **Construcción de un prototipo**: a partir de los conocimientos adquiridos en los diferentes estudios sobre la disciplina, la elección de la arquitectura y los diferentes diseños del sistema se construye un prototipo que permita observar todos los procesos definidos. Esto nos permitirá ganar más conocimientos sobre el problema originalmente expuesto dando la posibilidad a revisiones y trabajos futuros.
- **Observación y evaluación del sistema**: observar que el sistema cumple los casos de uso y establecer una serie de casos de prueba que permitan evaluar la completa funcionalidad del prototipo.

Este proceso es iterativo lo que quiere decir que en alguno de los pasos se puede aprender nuevo conocimiento aplicable a fases anteriores dando lugar a una redefinición de algunas de las fases a lo largo del desarrollo del proyecto.

1.4 Estructura de la memoria

Para concluir este apartado de introducción describirá la estructura del presente documento. Este consta de siete capítulos principales en los que se describe la información asociada al desarrollo del proyecto. Estos capítulos se completan con una serie de anexos con información adicional que presenta una notable relevancia. Los contenidos que presenta cada capítulo se enumerarán a continuación.

- **Introducción:** Se realizará una breve presentación del proyecto, explicando tanto los objetivos del proyecto como la metodología de trabajo seguida.
- **Estudio del problema:** Capítulo en el cual se comentará la situación actual de las tecnologías relacionadas con el proyecto y las herramientas analizadas para la realización del proyecto. Se incluirán además los problemas encontrados en la actualidad para poder realizar las tareas correctamente.
- **Gestión de proyecto software:** Este capítulo describirá las decisiones tomadas para planificar las tareas a realizar y conocer los recursos necesarios. De la misma manera se llevará a cabo un análisis de los riesgos que pueden afectar al proyecto.
- **Elaboración de la Solución:** En el cuarto capítulo se profundizará en el desarrollo del proyecto. Formarán parte de este capítulo el análisis de requisitos que ha de cumplir la aplicación, seguido del diseño de la arquitectura y del funcionamiento de la aplicación. Por último de cara a la implementación, se incluirán las herramientas y tecnologías utilizadas además de la organización y distribución de paquetes realizada.
- **Validación de la Solución:** En este capítulo contemplará las pruebas realizadas para comprobar el correcto funcionamiento del sistema desarrollado.
- **Conclusiones y Trabajos Futuros:** En el sexto capítulo se analizarán las conclusiones obtenidas tras el desarrollo y documentación del presente proyecto.

Además de estos capítulos al final de la memoria se incluyen una serie de anexos que completarán la información ofrecida en la memoria y una sección con una lista de las referencias bibliográficas utilizadas para el desarrollo de la misma

2 Estado de la Cuestión

La realización del proyecto ha implicado la revisión de algunos aspectos relacionados con la interacción que han ayudado a decidir cuál era la solución que se quiere dar al problema. Además de la revisión de las diferentes técnicas de interacción existentes, se han estudiado diferentes implementaciones de la técnica elegida para la solución. En primer lugar, se han analizado aspectos teóricos, haciendo una base de conocimientos, seguidos de diferentes técnicas de identificación y representación de las propiedades de los objetos físicos que se utilizarán en la aplicación, y terminando con una revisión de las tecnologías utilizadas para la implementación de estos dos aspectos.

2.1 La interacción

Uno de los elementos esenciales y que debe de tenerse muy en cuenta a la hora de diseñar un ECG, es la interacción. Sobre este elemento centraremos todo el estudio del proyecto. La Interacción se define como una *"acción que se ejerce recíprocamente entre dos o más objetos, agentes, fuerzas, funciones, etc."*. [2]. Para diseñar la interacción que se va incorporar un sistema se deben establecer los siguientes componentes: el paradigma de interacción, el modelo de interacción en el que se basará, el estilo de interacción sobre el que se desarrolla el juego y, por último, la elección de la interacción que ofrecerá el ECG. Estos componentes están interrelacionados entre sí. El paradigma de interacción se define como *"el modelo o patrón de HCI que engloba todos los aspectos de interacción, incluyendo los físicos, virtuales, perceptuales y cognitivos"* [4]. La elección de un tipo de paradigma depende de muchos factores, entre los cuales se encuentran las necesidades interactivas del programa y los requisitos expuestos por el tipo de usuarios que utilizarán el sistema. Algunos de los paradigmas de interacción definidos hoy en día son el **Large-Scale Computing**, destinado a MainFrames y Supercomputadores, **Personal Computing**, pensado para ordenadores personales, **Networked Computing**, propuesto para entornos corporativos y proyectos que tengan la necesidad de comunicación en diferentes espacios físicos, **Mobile Computing**, es una evolución del *Personal Computing* para adaptarlo a dispositivos móviles. Otros paradigmas que están empezando a surgir en la actualidad son el **Ubiquitous Computing, Wearable Computing e Invisible Computing**.

El Invisible Computing ha sido el paradigma elegido para el diseño de la interacción del entorno. La idea de este paradigma es conseguir facilitar la realización de las tareas eliminando la manipulación directa de la interfaz del sistema. Normalmente, el usuario no sólo se encuentra con el problema de resolver la tarea si no que, tal y como se muestra en la Ilustración 2, en muchos de los casos se añade el problema adicional de trasladar la tarea a la contexto proporcionado por la interfaz, es decir, conseguir realizar la tarea con los elementos interactivos proporcionados por el sistema. Además, una vez realizada la tarea, el usuario debe interpretar los resultados ofrecidos por la interfaz en el contexto de la tarea, que no siempre son intuitivos. Este proceso involucra algunos pasos intermedios entre la tarea a resolver y la resolución de la misma, que en algunas ocasiones puede provocar diferencias entre lo requerido por la situación y lo que la interfaz muestre. Este paradigma elimina estos pasos intermedios. Esto se puede realizar de dos formas, la primera realizando una interfaz tan simple e intuitiva que un usuario no tenga que perder el tiempo o la energía en resolver los

problemas relacionados/ocasionados por la interfaz, no debería pensar en la interfaz simplemente debería resolver la tarea. La segunda manera es haciendo la computación invisible, eliminando la interfaz.

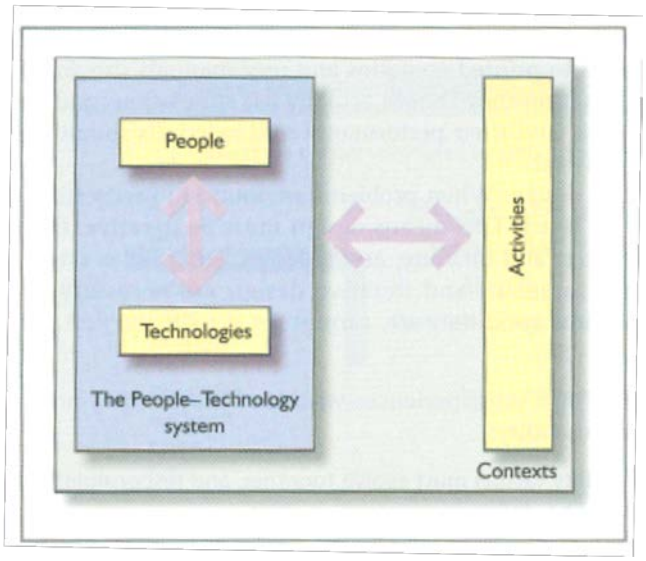


Ilustración 2.Problemática de la acción [5]

Normalmente a los paradigmas de interacción se les puede asociar, o vienen intrínsecamente ligados, ciertos estilos de interacción. Un estilo de interacción se define como *"la manera en la cual las personas interactúan con el ordenador"* [4]. Hoy en día hay muchos estilos de interacción en uso con sus ventajas e inconvenientes. Uno de los más antiguos y que todavía a día de hoy se continúa utilizando en algunos contextos es la **Línea de Comandos**. La interacción ofrecida por este estilo es basada en texto. El usuario escribe una cadena de texto, en el prompt de la consola, que hace referencia a un comando y el ordenador al ejecutar el comando muestra por consola los resultados del comando. Otro de los estilos de interacción que son utilizados hoy en día son las **Interfaces basadas en Menús**. Es una alternativa a la línea de comandos intentado que el uso de los ordenadores se extendiera a un mayor número de personas. Esta interfaz basada en menús ofrece a los usuarios una jerarquía secuencial con una lista de funcionalidades que ofrece el sistema. Con la evolución de las GUIs también evolucionaron los estilos de interacción. Un nuevo estilo es el **Relleno de Formularios**, ideados para la recopilación de información de forma secuencial. Otro estilo de interacción aparecido con esta evolución de las GUIs es la **Pregunta y Respuesta** o también denominado wizard, trata de llevar al usuario, a través de una serie de preguntas y posibles respuestas, a la configuración de una aplicación o sistema. Las GUIs seguían ofreciendo posibilidades para el desarrollo de nuevos estilos de interacción apareciendo la **Manipulación Directa**. Es la forma más común de HCI, como la propia expresión indica, es la manipulación de objetos del sistema por el usuario de manera directa y no como resultado de operaciones intermedias. Si nos ceñimos a la literalidad de la expresión el usuario no puede manipular directamente los objetos virtuales sin un dispositivo intermedio como el ratón, aunque en el caso de las pantallas táctiles o los entornos virtuales si se puede hacer, lo cual crea un pequeño problema en el concepto. Otro estilo de interacción son las **Metáforas.** Es un estilo de interacción baste

utilizado por la mayoría de los paradigmas de interacción, aquellos que utilizan GUIs. Dichas GUIs tratan de hacer más entendible los complejos procesos internos de los que está compuesto el sistema. Una de las formas de conseguir esto es haciendo uso de las metáforas que hacen una relación entre la utilidad y el significado del icono. Otros estilos de interacción son: **Navegación Web, Entornos tridimensionales, Interacción Natural e Interacción Tangible**.

La interacción tangible es un estilo de interacción en el que el usuario interactúa con un sistema digital a través de la manipulación de objetos físicos vinculados directamente a dicho sistema. Se asocian determinados objetos físicos (representaciones físicas) a cierta información digital, empleando estos objetos a la vez como representaciones y como controles de la información. El término, acuñado por Ishii, se refiere a un nuevo estilo de interacción que modifica (o elimina) la tradicional separación funcional entre entradas y salidas de la interfaz de un sistema informático. Estas siguen el modelo MCRpd (Model-control-representation (physical and digital)), que difumina la separación vista-control y en su lugar separa la vista en dos tipos de elementos: los citados objetos reales que permiten la manipulación directa (representación física), junto con dispositivos como pantallas, workbench, altavoces, etc., que permiten visualizar o escuchar información digital sin que medie ningún objeto real manipulable (representación digital).

La idea con una interfaz de usuario tangible, TUI, es tener una relación directa entre el sistema y la forma de controlarlo a través de manipulaciones físicas por tener un significado subyacente o la relación directa que conecta las manipulaciones físicas a las conductas que desencadenan en el sistema. En este último estado es donde radica la clave de las TUI. No es sólo una cuestión de tener un control físico de su sistema digital por tener sino asegurarse de que, mediante el control físico de su sistema digital, su uso tenga sentido para el usuario y un nuevo valor para un control más natural e intuitiva de su diseño. En cierto sentido, la interfaz se vuelve prácticamente invisible, ya que el usuario tiene un conocimiento inherente de manipulaciones tales como agarrar y mover objetos por lo que es capaz de concentrarse más en el sistema y los comportamientos provocado que en la manipulación de la misma. Esta es una de las razones por las cuales se ha elegido este estilo de interacción debido a que el grupo de personas inexpertas para la cual va dirigida la aplicación puede tener dificultades para entender como manipular el sistema si éste no es inherente. De esta misma idea se derivan otras como la de eliminar la distancia entre el mundo físico y digital añadiendo beneficios a ambos. Se tienen todas las ventajas de computación traídas de las interfaces de usuario gráficas, GUIs, además de poner toda la información y computación en nuestras manos. Además este estilo de interacción nos ofrece una liberación en la carga de computación accediendo a nuestro espacio cognitivo y adoptando un estilo de interacción más concreto a la tarea que deseamos realizar. Por último los objetos físicos ofrecen un fuerte affordance comparándolos con sus equivalentes virtuales.

En resumen las características de una TUI son:

- Las representaciones físicas está computacionalmente unida a la información digital subyacente.
- Las representaciones físicas incorporan mecanismos de control interactivo.

- Las representaciones físicas están perceptualmente unidas a las representaciones digitales de forma activa.
- Estado físico de los elementos tangibles incorpora aspectos claves del estado digital de un sistema.
- Eliminación de la distancia entre el mundo digital y el mundo físico.
- Especificación de un entorno para una tarea concreta disminuyendo la carga de computación del sistema.

2.2 Identificación de objetos y sus propiedades

Uno de los objetivos del proyecto es el reconocimiento e identificación de las propiedades de los objetos físicos. Como se ha indicado en la introducción el tipo de propiedades que se quieren representar son dos: de la naturaleza física del objeto y sobre su uso y utilidad tales como el riesgo, tipo de contenido etc… Para lograr este objetivo existen varias técnicas entre las que se encuentran el etiquetado de objetos y el reconocimiento visual.

Para poder llevar a cabo el reconocimiento visual se requieren técnicas de *machine learning,* Aprendizaje automático*.* EL aprendizaje automático es una rama de la Inteligencia Artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender. De forma más concreta, se trata de crear programas capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento. En muchas ocasiones el campo de actuación del Aprendizaje Automático se solapa con el de la estadística, ya que las dos disciplinas se basan en el análisis de datos. Sin embargo, el Aprendizaje Automático se centra más en el estudio de la Complejidad Computacional de los problemas. El Aprendizaje Automático puede ser visto como un intento de automatizar algunas partes del método científico mediante métodos matemáticos. Los diferentes algoritmos de Aprendizaje Automático se agrupan en una taxonomía en función de la salida de los mismos. Algunos tipos de algoritmos son:

- **Aprendizaje supervisado:** el algoritmo produce una función que establece una correspondencia entre las entradas y las salidas deseadas del sistema. Un ejemplo de este tipo de algoritmo es el problema de clasificación, donde el sistema de aprendizaje trata de etiquetar (clasificar) una serie de vectores utilizando una entre varias categorías (clases). La base de conocimiento del sistema está formada por ejemplos de etiquetados anteriores.
- **Aprendizaje no supervisado:** todo el proceso de modelado se lleva a cabo sobre un conjunto de ejemplos formado tan sólo por entradas al sistema. No se tiene información sobre las categorías de esos ejemplos.
- **Aprendizaje por refuerzo:** el algoritmo aprende observando el mundo que le rodea. Su información de entrada es el feedback o retroalimentación que obtiene del mundo exterior como respuesta a sus acciones.
- **Transducción:** similar al aprendizaje supervisado, pero no construye de forma explícita una función. Trata de predecir las categorías de los futuros ejemplos basándose en los ejemplos de entrada, sus respectivas categorías y los ejemplos nuevos al sistema.
- **Aprendizaje multi-tarea:** métodos de aprendizaje que usan conocimiento previamente aprendido por el sistema de cara a enfrentarse a problemas parecidos a los ya vistos.

El Aprendizaje Automático tiene una amplia gama de aplicaciones, incluyendo motores de búsqueda, diagnósticos médicos, detección de fraude en el uso de tarjetas de crédito, análisis del mercado de valores, clasificación de secuencias de ADN, reconocimiento de imágenes, del habla y del lenguaje escrito, juegos y robótica. La utilización del aprendizaje automático servirá para el reconocimiento del objeto y una vez reconocido se enlazará con una base de conocimientos donde se encontrarían todas las características del objeto reconocido. Con esta técnica se podrían generar problemas de ambigüedad a la hora tanto de la detección como de conocer algunas de sus propiedades.

Otra de las técnicas para la identificación de los objetos y sus propiedades es el etiquetado de los objetos a través de códigos en etiquetas electrónicas (ver Ilustración 6). Con esta tecnología se consigue incluir la información del objeto codificada en la etiqueta, de tal forma que, leyendo la etiqueta se puede obtener toda la información del objeto. Algunas de las técnicas utilizadas para la codificación de etiquetas electrónicas son:

- **Código de barras:** es un código basado en la representación mediante un conjunto de líneas paralelas verticales de distinto grosor y espaciado que en su conjunto contienen una determinada información. De este modo, el código de barras permite reconocer rápidamente un artículo en un punto de la cadena logística y así poder realizar inventario o consultar sus características asociadas. Actualmente, el código de barras está implantado masivamente de forma global. Es un sistema que permite la identificación de las unidades comerciales y logísticas de forma única, global y no ambigua. Este conjunto de barras y espacios codifican pequeñas cadenas de caracteres en los símbolos impresos. La correspondencia o mapeo entre la información y el código que la representa se denomina simbología. Estas simbologías pueden ser clasificadas en dos grupos atendiendo a dos criterios diferentes:

    - *Continua o discreta.* Los caracteres en las simbologías continuas comienzan con un espacio y en el siguiente comienzan con una barra (o viceversa). Sin embargo, en los caracteres en las simbologías discretas, éstos comienzan y terminan con barras y el espacio entre caracteres es ignorado, ya que no es lo suficientemente ancho.
    - *Bidimensional o multidimensional:* las barras en las simbologías bidimensionales pueden ser anchas o estrechas. Sin embargo, las barras en las simbologías multidimensionales son múltiplos de una anchura determinada (X). De esta forma, se emplean barras con anchura X, 2X, 3X, y 4X.

  La estructura de un código de barras, como se puede observar en la Ilustración 3, se divide en cuatro secciones:

    1. Quiet Zone.
    2. Carácter inicio (derecha), Carácter terminación (izquierda).
    3. Carácter de datos.
    4. Checksum.

Por último el uso del código de barras proporciona muchas ventajas. Entre otras más específicas del sector donde se usa se encuentran:

- Se imprime a bajos costos.
- Posee porcentajes muy bajos de error.
- Permite capturar rápidamente los datos.
- Los equipos de lectura e impresión de código de barras son flexibles y fáciles de conectar e instalar.
- Permite automatizar el registro y seguimiento de los productos.
- La información se procesa y almacena con base en un sistema digital binario donde todo se resume a sucesiones de unos y ceros. La memoria y central de decisiones lógicas es un computador electrónico del tipo estándar, disponible ya en muchas empresas comerciales y generalmente compatibles con las distintas marcas y modelos de preferencia en cada país. Estos equipos permiten también interconectar entre sí distintas sucursales o distribuidores centralizando toda la información. Ahora el distribuidor puede conocer mejor los parámetros dinámicos de sus circuitos comerciales, permitiéndole mejorar el rendimiento y la toma de decisiones, ya que conocerá con exactitud y al instante toda la información proveniente de las bocas de venta estén o no en su casa central. Conoce los tiempos de permanencia de depósito de cada producto y los días y horas en que los consumidores realizan sus rutinas de compras, pudiendo entonces decidir en qué momento debe presentar ofertas, de qué productos y a qué precios.

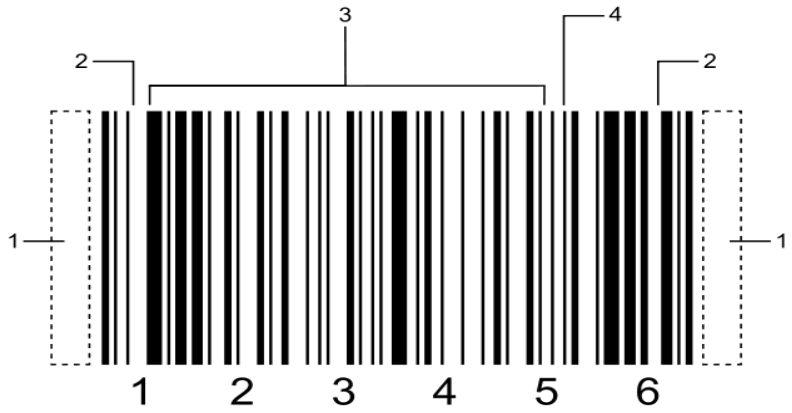


Ilustración 3.Estructura código de barras.

- **Qr-Code:** Un código QR, *Quick Response Barcode*, es un sistema para almacenar información en una matriz de puntos o un código de barras bidimensional creado por la compañía japonesa Denso-Wave en 1994; se caracterizan por los tres cuadrados que se encuentran en las esquinas y que permiten detectar la posición del código al lector. La sigla «QR» se derivó de la frase inglesa «Quick Response» pues el creador aspiraba a que el código permitiera que su contenido se leyera a alta velocidad. Aunque inicialmente se usó para registrar repuestos en el área de la fabricación de vehículos,

hoy, los códigos QR se usan para administración de inventarios en una gran variedad de industrias. Recientemente, la inclusión de software que lee códigos QR en teléfonos móviles, ha permitido nuevos usos orientados al consumidor, que se manifiestan en comodidades como el dejar de tener que introducir datos de forma manual en los teléfonos. Las direcciones y los URLs se están volviendo cada vez más comunes en revistas y anuncios. El agregado de códigos QR en tarjetas de presentación también se está haciendo común, simplificando en gran medida la tarea de introducir detalles individuales de un nuevo cliente en la agenda de un teléfono móvil. Un detalle muy importante sobre el código QR es que su código es abierto y que sus derechos de patente (propiedad de Denso Wave) no son ejercidos. La estructura del QR-Code se puede observar en la Ilustración 4. Algunas de sus aplicaciones son Micro código QR, en el arte, en el ajedrez, Qr-codes personalizados.

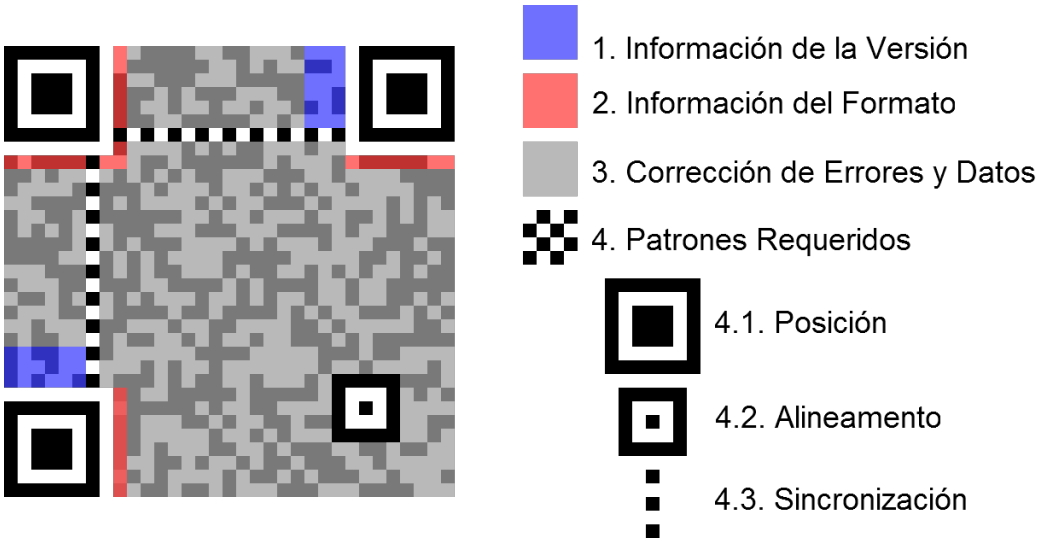


Ilustración 4. Estructura QR-Code

- **Datamatrix:** o codificación de datos 2D, es un nuevo 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 (legible hasta con un 20%-30% dañado). Además no es necesario un alto contraste para reconocer el código. El código está formado por celdas de color blanco y negro (perforadas o no perforadas en el caso de la micropercusión) que forman una figura cuadrada o rectangular. Cada una de esas celdas representa un bit de información. La información puede estar codificada como texto o datos en bruto (raw data en inglés). Aplicación más popular para el código Datamatrix es marcar pequeñas piezas o bien marcar directamente las piezas por deformación sin utilizar pegatinas u otros métodos. El marcado directo de piezas de manera indeleble, asegura que el código marcado no se separará nunca de la pieza marcada. La capacidad de un código Datamatrix de almacenar gran cantidad de información en espacio legible de aproximadamente 2 ó 3 mm2 y el hecho de que puede ser leído con solo un ratio del 20% de contraste lumínico. El código Datamatrix

es parte de una nueva corriente en cuanto a la trazabilidad en muchas industrias, particularmente la aeroespacial, donde los controles de calidad son muy exigentes. Los códigos Datamatrix (y los datos alfanuméricos que los acompañan) identifican los detalles del componente marcado, incluyendo el fabricante, el número de producto y un número de serie único. El símbolo Datamatrix está compuesto de módulos de celdas cuadradas definidas dentro de un perímetro marcado. Es posible codificar hasta 3116 caracteres ASCII y en la micropercusión superan los 100 caracteres. Cada símbolo consiste en zonas de datos que forman módulos cuadrados en una secuencia regular. Los símbolos más grandes contienen varios módulos y cada zona de datos está delimitada por una línea continua en 2 caras y discontinua en otras 2. Cada código individual está rodeado de una zona lisa que haga las veces de margen. Cada código tiene un número determinado de filas y columnas. La mayoría de los códigos Datamatrix son cuadrados y van desde 10×10 hasta 144×144 puntos. De todos modos también es posible encontrar códigos Datamatrix de forma rectangular con tamaños que van desde los 8×18 a los 16×48. Todos los códigos pueden ser reconocidos desde la esquina superior derecha cuando son iluminados (binario 0). ECC200 Es la última versión del código Datamatrix y soporta sistemas de codificación avanzadas búsqueda de errores y corrección algoritmos (como el Reed-Solomon). Este algoritmo permite el reconocimiento de códigos que están dañados hasta en un 60%. Para la industria, los códigos Datamatrix pueden ser marcados de manera directa en los componentes o piezas fabricadas, asegurando así que cada pieza recibe un único código que la identifica de todas las demás. Las técnicas que se utilizan para marcar directamente las piezas son variadas. Después de marcar una pieza con un código Datamatrix, lo más común es utilizar es sistema de reconocimiento compuesto por una cámara y un software especial. Esta verificación del Datamatrix asegura que el código cumple con los estándares y también que podrá ser leído durante toda la vida útil de la pieza.

- **TrackMate-Code:** TrackMate utiliza un código de barras pequeñas, especialmente diseñado circular que almacena la información que puede ser rápidamente interpretado por la aplicación Tracker TrackMate. La etiqueta mide menos de 2.5x2.5cm (0,95 centímetros cuadrados) y contiene una identificación única de seis bytes (Más de 280 billón identificadores únicos son posibles), así como una suma de comprobación de un solo byte para los simples detección de errores. Mediante el uso de una forma circular de la etiqueta, simples algoritmos de rotación invariante se pueden utilizar para encontrar el centro de cada etiqueta, sin necesidad de gama alta procesadores o equipos de imágenes. Dentro del proyecto TrackMate diseñó, un código de colores con fines descriptivos. Técnicamente, la etiqueta consiste en un patrón rotatorio invariante, tres anillos de datos, y cuatro de color analógico de detección zonas (ver Ilustración 5). El anillo exterior (en azul) se utiliza para determinar la orientación de la etiqueta a través de tres segmentos de color negro dentro de una franja blanca. La centro de datos de circunvalación (en rojo) contiene los primeros cuatro bytes (32 bits) de la etiqueta de dirección, codificada como segmentos cortos en blanco y negro. Del mismo modo, el anillo interno de los datos (En verde) contiene tres bytes (24 bits) de información: los dos primeros bytes de la etiqueta de dirección, así como una suma de comprobación invertida de un byte, que se utiliza para la detección de errores. Las cuatro regiones circulares en las esquinas (que se muestra en

cian) son opcionales y se cómo leer valores analógicos (detección de la actual color RGB en el centro de cada región). Al permitir la entrada de color, las etiquetas pueden tener un aumento funcionalidad (incluyendo botones y deslizadores) sin la necesidad de electrónica periférica. Los círculos concéntricos restantes en blanco y negro son importantes para encontrar rápidamente la etiqueta de una imagen vista por el Tracker. Puesto que los anillos son concéntricos, independientemente de la rotación de la etiqueta en la superficie, los algoritmos simples pueden ser utilizados para determinar los lugares que son altamente probables de ser el centro de cada etiqueta. Las etiquetas TrackMate son, de escala no invariante (que debe ser el tamaño correcto) o de perspectiva invariante. A pesar de las limitaciones, esta elección fue hecha por tres razones: el algoritmo de detección de las etiquetas puede ser muy simple y rápido, tiene buena inmunidad al ruido (es decir, en detectará las etiquetas que parece un muy buen partido), y permite una alta densidad de datos (es decir, de forma fiable puede leer 48 bits de información en un pequeño gráfico).

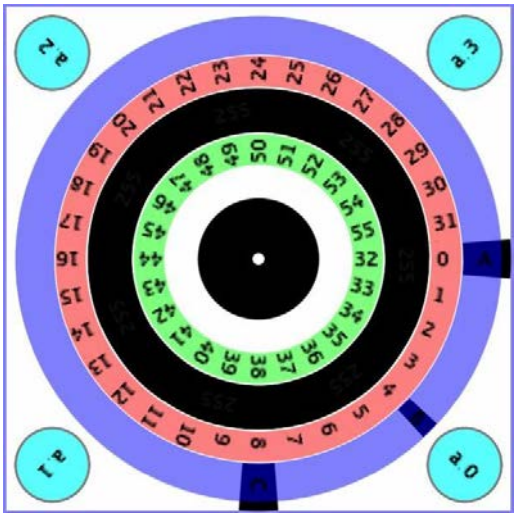


Ilustración 5.Estructura TrackMate-Code

Ilustración 6.Diferente Tipos de Etiquetas electrónicas

2.3 Revisión de Tecnología

Después de la realización de una base de conocimientos en los apartados anteriores, sobre diferentes aspectos de la interacción, se determina que para la solución propuesta se utilizará el paradigma del invisible computing y el estilo de interacción tangible. Además también se hace un estudio sobre diferentes métodos para la identificación de propiedades en objetos físicos en el técnica elegida es el TrackMate-Code. Para la elección de los ejemplos en este apartado se ha seguido el criterio de que los ejemplos utilicen algunos de los elementos elegidos para el desarrollo de la solución. En base a este criterio se puede encontrar a:

a) **AudioPad**: En 2002, James Patten desarrolló una interfaz tangible basada en la música sobre una mesa con fichas físicas que se vinculaban a los sonidos digitales. Audiopad, un aplicación para mezclar audio para DJs electrónicos, utiliza un conjunto de fichas en una mesa redonda, que representan varias muestras de sonido, un micrófono, y una herramienta para seleccionar de forma dinámica diferentes sonidos, mientras que un proyector superpone la información sobre el audio que se reproduce (ver Ilustración 7). Audiopad está construido utilizando la plataforma Sensetable, una superficie analógica que puede localizar la posición de los circuitos, especialmente certera cuando se encuentran cerca. La interfaz se desarrolló rápidamente, pensando más en un instrumento que en un prototipo, surgió como un instrumento para dar rendimiento a algunas características (por ejemplo, efectos en tiempo real, el tempo, y los sonidos de agrupación en una estructura jerárquica). Además de la aplicación Audiopad, la infraestructura Sensetable también, fue utilizada para una amplia gama de prototipos de interfaz tangible, incluyendo la red herramientas de simulación y análisis. Las aplicaciones que requieren tanto la posición y rotación de la información (por ejemplo, una implementación de URP, una mesa de trabajo con la planificación urbana modelos arquitectónicos) fueron capaces de obtener mediante el uso de dos símbolos unidos a un solo objeto, entonces la solución para el ángulo entre sus ubicaciones.

Ilustración 7. AudioPad

b) **Reactable**: En 2005, Sergi Jord, Kaltenburnner Martin, Geiger Gnter, y Ross Bencina crearon Reactable, un material electro-acústico de instrumentos musicales con soporte multi-touch. En contraste con Audiopad, que requiere un circuito de detección para su funcionalidad, Reactable, utiliza un seguimiento óptico para encontrar los objetos en su superficie. Un proyector, se utiliza también para superponer la información, pero como la luz del proyector podría interferir con la detección, la cámara utiliza un filtró para sólo ver la luz infrarroja. Con el uso de una "ameba", un patrón de puntos impresos de los marcadores de referencia en blanco y negro, el software de Reactable encuentra cada ficha mediante la búsqueda de contornos de la imagen específica (ver Ilustración 8). Este la información es procesada para identificar de forma única cada token (por lo general hasta 180 patrones diferentes, pero el número puede ser aumentado mediante el uso de una etiqueta de tamaño más grande), como así como encontrar su posición en 2D y orientación. Aunque muchos proyectos antes había utilizado métodos ópticos de diferentes formas, el proyecto Reactable usa especialmente diseñado etiquetas que eran fáciles de procesar y también decidió hacer su software de seguimiento código abierto. Los hackers que vieron Reactable ya sea en línea o en persona comenzaron a buscar maneras de hacer su propio sistema similar. Se crearon software para automatizar diseños de etiquetas y otros proyectos no afiliados comenzaron a incorporar el sistema Reactable de seguimiento en sus proyectos.

Ilustración 8. Reactable

c) **Phidgets**: Son un gran conjunto de sensores plug-and-play y actuadores que se pueden utilizar para crear sus propios prototipos de interfaces tangibles. Son utilizados en el concepto “do it yourselft”, es decir, háztelo tú mismo. Con una colección completa de dispositivos como acelerómetros, sensores de distancia y movimiento, LED, motores, etc…, que cuando se conecta a un PC y son muy fáciles de usar en bloques para los diseñadores de interacción que quieren experimentar con las interfaces de usuario físico.

d) **TrackMate**: TrackMate es una iniciativa de código abierto para la creación a bajo costo, do-it-yourself, de un sistema de seguimiento tangible. La aplicación TrackMate tracker permite a cualquier ordenador reconocer los objetos etiquetados

y su correspondiente posición, rotación, y la información de color cuando se coloca sobre una superficie. Todos los datos se envían desde el gestor a través de LusidOSC (Un protocolo de capa única de los dispositivos de entrada espacial), permitiendo a cualquier cliente basado en aplicaciones LusidOSC trabajar con el sistema. Como se muestra en la Ilustración 9, hay muchos componentes en el sistema TrackMate, la infraestructura técnica se muestra en azul, el usuario de la aplicación en amarillo, y las herramientas de la comunidad en verde. Los problemas a los que se enfrentan las interfaces tangibles son amplios y complejos, que van desde los costos y arquitectura del sistema hasta el diseño de aplicaciones convincentes y participación de la comunidad. TrackMate pretende dar un gran paso adelante al abordar el mayor número de estas cuestiones como sea posible con un enfoque en el aprendizaje de los proyectos anteriores y la combinación de sus distintas fortalezas a través de un sistema unificado en una la iniciativa a gran escala. Las etiquetas para el sistema de TrackMate permiten el reconocimiento rápido, se pueden crear a partir de miles de millones de posibles identificadores únicos, son de tamaño pequeño, y se pueden producir con cualquier impresora. El uso de cualquier estándar y una cámara web conectada a un ordenador (Sistema de Windows, Mac y Linux son soportados), permite al software de seguimiento ser fácilmente configurado para leer las etiquetas y transferir la información a las aplicaciones que desee. Para para empezar, TrackMate no requiere de E/S, liberando al usuario de las limitaciones de un equipo costoso (como un proyector o pantalla de gran tamaño) y los complicados procedimientos de configuración (tales como la alineación de los objetos con representaciones gráficas o el uso de la iluminación de infrarrojos). La E/S se puede lograr si se desea, pero ha sido intencionalmente reservados para usuarios avanzados trabajando en proyectos complejos que busquen un techo más alto. Todo el código, el contenido y las instrucciones para TrackMate son de código abierto y publicado a Internet para que cualquiera lo use, critique y modifique según sus necesidades.

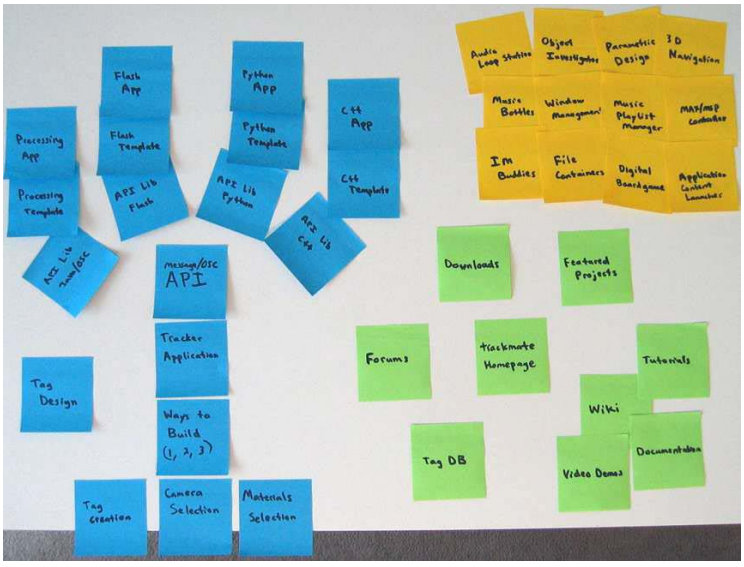


Ilustración 9.Componentes TrackMate

3 Gestión del Proyecto Software

Este capítulo del documento se centra en estudiar las necesidades del producto realizando las correspondientes estimaciones en tiempo, recursos y costes, así como su encuadre dentro de un plan de trabajo.

3.1 Definición del proyecto

Se pretende desarrollar un entorno físico a través del cual los usuarios puedan realizar diferentes tareas para el aprendizaje. El entorno físico estará dotado de un mecanismo de comunicación a través del cual se podrá conectar a diferentes sistemas externos tanto físicos como virtuales. Los juegos son otra parte del proyecto a través de los cuales se realizarán los aprendizajes correspondientes. A todo el sistema, se le unen diferentes objetos físicos que servirán como elementos interactivos en la interfaz ofrecida por el entorno físico. A dichos objetos se les identificarán varias de sus propiedades físicas tales como el color, forma, material del que están hechos y también propiedades relacionadas con su uso y utilidad tales como su naturaleza, el posible tipo de contenido del objeto, el nivel de riesgo existente al utilizar el objeto. Otro de los objetivos del proyecto es que se puedan utilizar varios objetos a la vez para dar la posibilidad de resolver los desafíos ofrecidos por los juegos de manera cooperativa y así poder aumentar las capacidades interactivas del entorno. Por último en el proyecto se tratará de que el entorno desarrollado ofrezca una interfaz cuyo estilo de interacción permita acciones naturales facilitando el proceso de aprendizaje.

3.2 Ciclo de vida del proyecto

Debido a la naturaleza del desarrollo del prototipo del sistema descrito en el '*3.1 Definición del proyecto*' del documento, define el ciclo de vida del presente proyecto de desarrollo como una mezcla entre el modelo *Basado en Prototipado* y el modelo en *Espiral* (ver Ilustración 10). En este caso, la experiencia del desarrollo del prototipo existente y su evaluación permitirá una definición de las especificaciones más completa y segura para el producto definitivo. Para asegurar la satisfacción con el producto generado, tras el desarrollo final del sistema se realizarán nuevos procedimientos de evaluación. Los resultados de esta evaluación se trasladarán primero al diseño y posteriormente a la implementación, definiendo con ello el segundo ciclo del modelo en espiral.

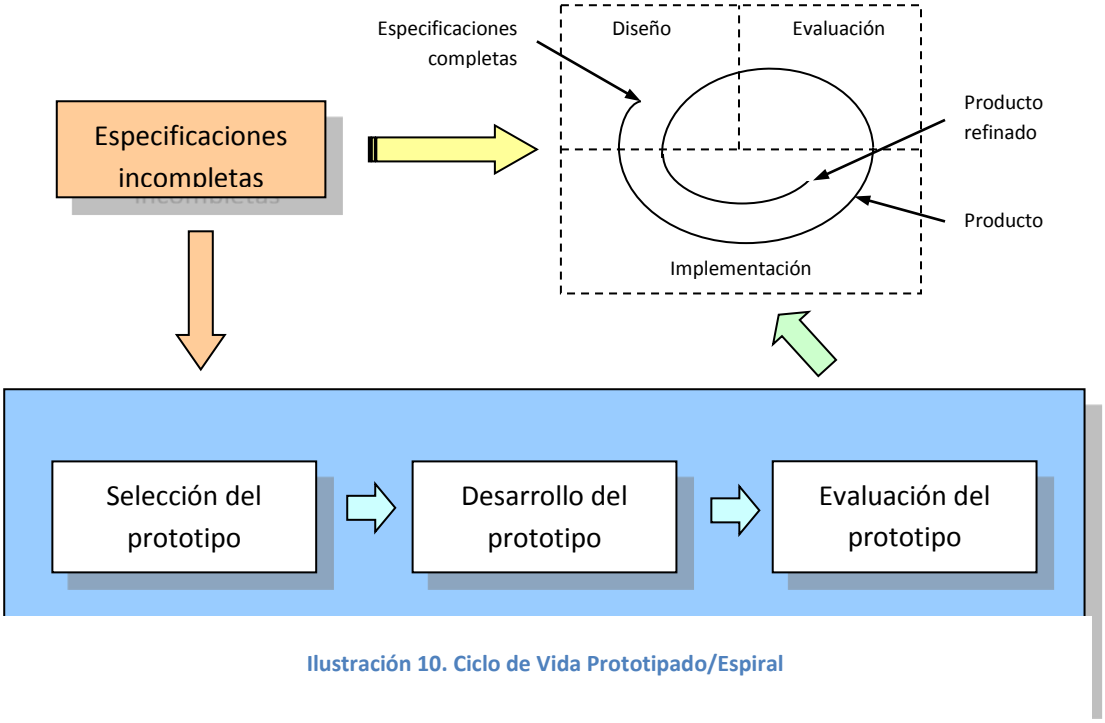


Ilustración 10. Ciclo de Vida Prototipado/Espiral

3.3 Gestión de Recursos

En cuanto a los recursos que se tendrán que utilizar se mencionan los humanos, tecnológicos y físicos. Como recursos humanos es imprescindible contar con un equipo de trabajo experimentado en el campo, que sean capaces de entender el dominio del problema y que se encarguen de proponer soluciones que cumplan los requerimientos del cliente. Como recursos físicos, se necesitará material de oficina y equipos informáticos en los cuales se realizará el desarrollo del proyecto. A continuación se enumeran cada uno de los roles del equipo de proyecto así como la explicación general de su cometido:

- **Jefe de proyecto:** responsable del proyecto. Se encargará de la dirección para la realización del mismo, así como el análisis y el diseño de la arquitectura del proyecto. Será la persona que mantenga contacto con el cliente.
- **Programador sénior:** miembro responsable de la realización y revisión de las tareas de diseño y e implementación del sistema.
- **Programador:** miembro del equipo de trabajo centrado en las tareas de implementación y pruebas del sistema.

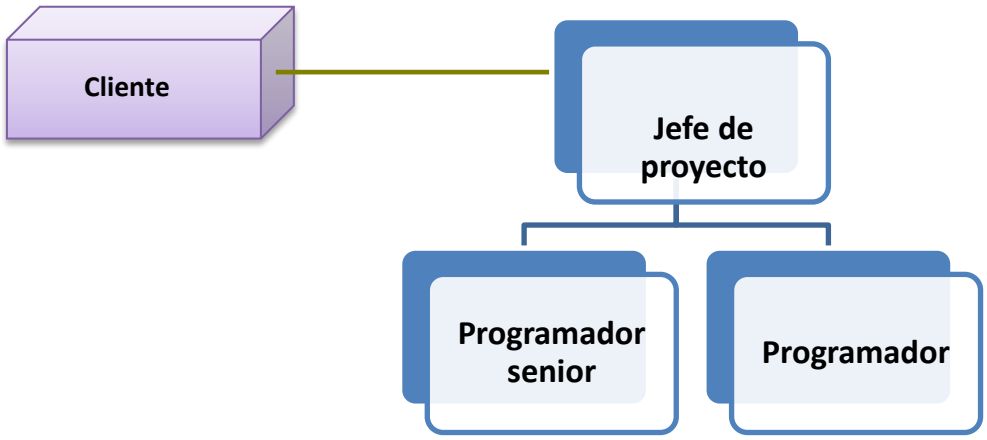


Ilustración 11.Organigrama del Equipo de Trabajo

3.4 Planificación de tareas

Las dos grandes tareas del proyecto son la planificación y seguimiento, y el proceso de desarrollo.

- **Planificación y Seguimiento.** Para asegurar el cumplimiento de los plazos establecidos para el desarrollo del proyecto, mensualmente todos los miembros del equipo de trabajo deberán realizar informes de evolución del trabajo hasta ese punto. A partir de esta información, el responsable de la planificación, en este caso el Jefe de proyecto, tendrá que realizar los ajustes necesarios sobre la distribución de tareas de tal forma que se satisfagan las restricciones temporales establecidas.

Las tareas necesarias dentro del proceso de desarrollo, que se deberán llevar a cabo teniendo en cuenta la metodología **Métrica V3** [7], son las siguientes:

- **EVS (estudio de viabilidad):** su objetivo es recoger los requisitos que los clientes proporcionan o que han sido adquiridos mediante entrevistas u otras técnicas.
- **Análisis.** El objetivo de la fase de análisis es capturar todos los requisitos del sistema, clasificarlos y, de esta forma, poder garantizar el cumplimiento de los mismos al final del proyecto. El análisis debe definir claramente el alcance del proyecto. Esta fase se dividirá a su vez en dos tareas: la primera de identificación global de los requisitos y la segunda de especificación de las características asociadas a dichos requisitos.
- **ASI-DAS (análisis del sistema de información):** El objetivo de la fase de análisis es capturar todos los requisitos del sistema, clasificarlos y, de esta forma, poder garantizar el cumplimiento de los mismos al final del proyecto. El análisis debe definir claramente el alcance del proyecto. Esta fase se dividirá a su vez en dos tareas: la primera de identificación global de los requisitos y la segunda de especificación de las características asociadas a dichos requisitos.

- **DSI (diseño del sistema de información):** su propósito es realizar la definición de la arquitectura del sistema y del entorno tecnológico que le va a dar soporte, junto con la especificación detallada de los componentes del sistema de información. Esta fase se subdivide a su vez en 3 subfases:

    - ***Diseño del sistema.*** *En esta fase se modelarán las características y restricciones del sistema desde el punto de vista general. Esta fase incluye las tareas de definición de la arquitectura del sistema y la especificación de las tecnologías involucradas en el desarrollo e implantación del producto.*
    - ***Diseño detallado.*** *Para esta segunda fase del diseño del sistema se dejará la realización de los modelos específicos. Estos se definirán en tres tareas que abarcan la persistencia, el dominio y la interfaz del sistema.*
    - ***Modificaciones post-evaluación.*** *Esta fase abarcará las posibles modificaciones que tengan que realizarse sobre el diseño una vez realizada la evaluación de la satisfacción con el sistema.*

- **CSI (construcción del sistema de información):** esta tarea se encarga de generar el código de los componentes del sistema de información. Se desarrollan todos los procedimientos de operación y seguridad y se elaboran todos los manuales de usuario final y de explotación con el objetivo de asegurar el correcto funcionamiento del sistema para su posterior implantación. Esta tarea se divide a su vez en 2 subfases:

    - ***Codificación.*** *Esta fase abarca la generación del código fuente del sistema. Para esta fase se han identificado dos subfases principales: en la primera de ellas se implementará la totalidad del sistema, mientras que en la segunda se realizarán las posibles modificaciones sobre el código identificadas en la evaluación de la satisfacción.*
    - ***Empaquetado.*** *En esta fase se generarán los elementos necesarios para la implantación final del producto. Esto incluye los ficheros de configuración y empaquetado del software.*

- **IAS (implantación y aceptación del sistema):** su objetivo es la entrega y aceptación del sistema en su totalidad, y la realización de todas las actividades necesarias para que éste pase a producción. En esta fase, se definirán y aplicarán el conjunto de pruebas que el sistema deberá pasar para asegurar su correcto funcionamiento. Además se realizará una evaluación del producto con el fin de asegurar la satisfacción final de los usuarios

A continuación se determinará el tiempo que se empleará en cada tarea teniendo en cuenta los objetivos que se quieren alcanzar y los recursos disponibles (ver Ilustración 12). Los recursos temporales son de 3 meses, por lo que el cómputo total de desarrollo del proyecto no debe ser superior. Para representar la planificación se ha decidido utilizar el Diagrama de Gantt (ver Ilustración 13). En él se pueden observar todas las tareas, su duración y si éstas se solapan con otras tareas. Para terminar se ha creado un organigrama de las tareas y los recursos personales asociados a cada una de las tareas (ver Ilustración 14). De esta forma se muestra como están organizadas las tareas de manera jerárquica.

Diagrama de Gantt

	Modo de	Nombre de tarea	Duración	Comienzo	Fir	Nombres de los recursos	Costo
1		**Planificación y Seguimiento**	**3,25 mss**	**lun 02/05/11**	**vie 29/07/11**		**339,90 €**
2		Punto de Control 1	1 hora	lun 02/05/11	lun 02/05/11	Jefe de Proyecto	45,00 €
3		Punto de Control 2	1 hora	jue 05/05/11	jue 05/05/11	Jefe de Proyecto	45,00 €
4		Punto de Control 3	1 hora	lun 16/05/11	lun 16/05/11	Jefe de Proyecto;Programador Senior	75,00 €
5		Punto de Control 4	1 hora	lun 27/06/11	lun 27/06/11	Jefe de Proyecto;Programador Senior;Programador	87,45 €
6		Punto de Control 5	1 hora	lun 25/07/11	lun 25/07/11	Programador;Jefe de Proyecto;Programador Senior	87,45 €
7		**Proceso de Desarrollo**	**3 mss**	**lun 09/05/11**	**vie 29/07/11**		**24.409,80 €**
8		**ASI-DAS (Análisis)**	**31 días**	**lun 09/05/11**	**lun 20/06/11**		**5.280,00 €**
9		Definición de Requisitos	6 días	lun 09/05/11	lun 20/06/11	Jefe de Proyecto;Programador Senior[50%]	2.400,00 €
10		Espicificación de Requisitos	11 días	vie 13/05/11	vie 17/06/11	Jefe de Proyecto;Programador Senior[50%]	2.880,00 €
11		**DSI (Diseño)**	**50 días**	**lun 16/05/11**	**vie 22/07/11**		**10.080,00 €**
12		**Diseño del Sistema**	**21 días**	**lun 16/05/11**	**lun 13/06/11**		**5.040,00 €**
13		Arquitectura	7 días	lun 16/05/11	lun 13/06/11	Jefe de Proyecto	2.520,00 €
14		Tecnología	7 días	jue 19/05/11	vie 27/05/11	Jefe de Proyecto	2.520,00 €
15		**Diseño detallado**	**15 días**	**lun 23/05/11**	**vie 10/06/11**		**3.330,00 €**
16		Modelo de Datos	5 días	lun 23/05/11	vie 27/05/11	Programador Senior	1.170,00 €
17		Modelo de Información	5 días	lun 30/05/11	vie 03/06/11	Programador Senior	1.200,00 €
18		Interfaz	5 días	lun 06/06/11	vie 10/06/11	Programador Senior	960,00 €
19		Modificaciones Post-Evaluación	7,88 días	mié 13/07/11	vie 22/07/11	Jefe de Proyecto[50%];Programador Senior	1.710,00 €
20		**CSI (Implementación)**	**38 días**	**mié 01/06/11**	**vie 22/07/11**		**4.051,80 €**
21		**Codificación**	**38 días**	**mié 01/06/11**	**vie 22/07/11**		**4.014,45 €**
22		Codificación General	35 días	mié 01/06/11	mar 19/07/11	Programador[50%];Programador Senior[25%]	3.033,00 €
23		Versión Beta	0,13 días	vie 22/07/11	vie 22/07/11	Programador	12,45 €
24		Codificación Post- Evaluación	10 días	lun 11/07/11	vie 22/07/11	Programador[50%];Programador Senior[25%]	969,00 €
25		**Empaquetado**	**2 días**	**jue 21/07/11**	**vie 22/07/11**		**37,35 €**
26		Ficheros de Configuración	2 días	jue 21/07/11	vie 22/07/11	Programador[20%]	37,35 €
27		**IAS (Implantación)**	**15 días**	**lun 11/07/11**	**vie 29/07/11**		**4.998,00 €**
28		Pruebas Preliminares	10 días	lun 11/07/11	vie 22/07/11	Jefe de Proyecto[25%];Programador[25%];Programador Senior[50%]	1.749,00 €
29		Pruebas Finales	5 días	lun 25/07/11	vie 29/07/11	Jefe de Proyecto;Programador;Programador Senior	3.249,00 €

Ilustración 12.Planificación de Tareas y Recursos

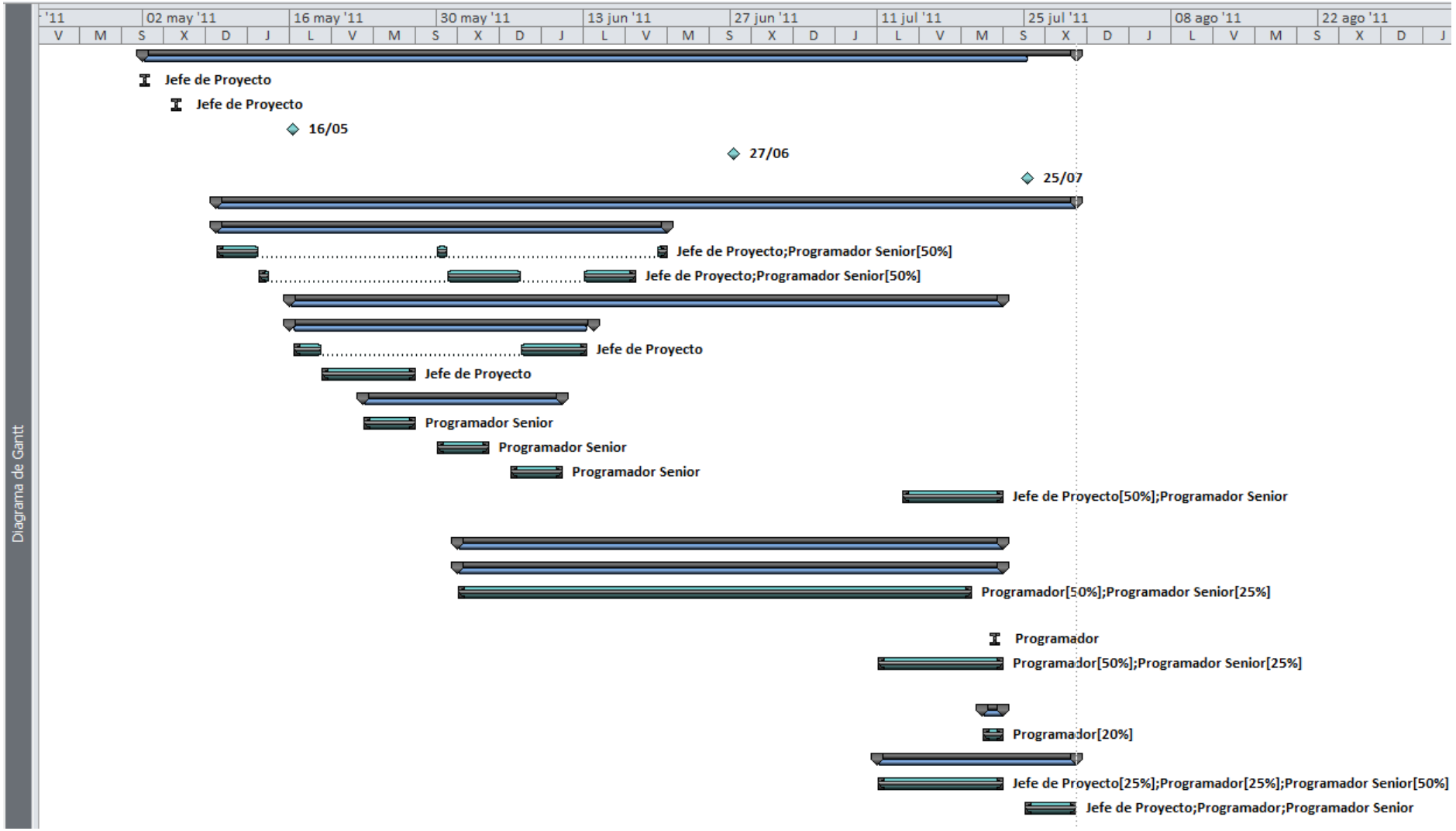
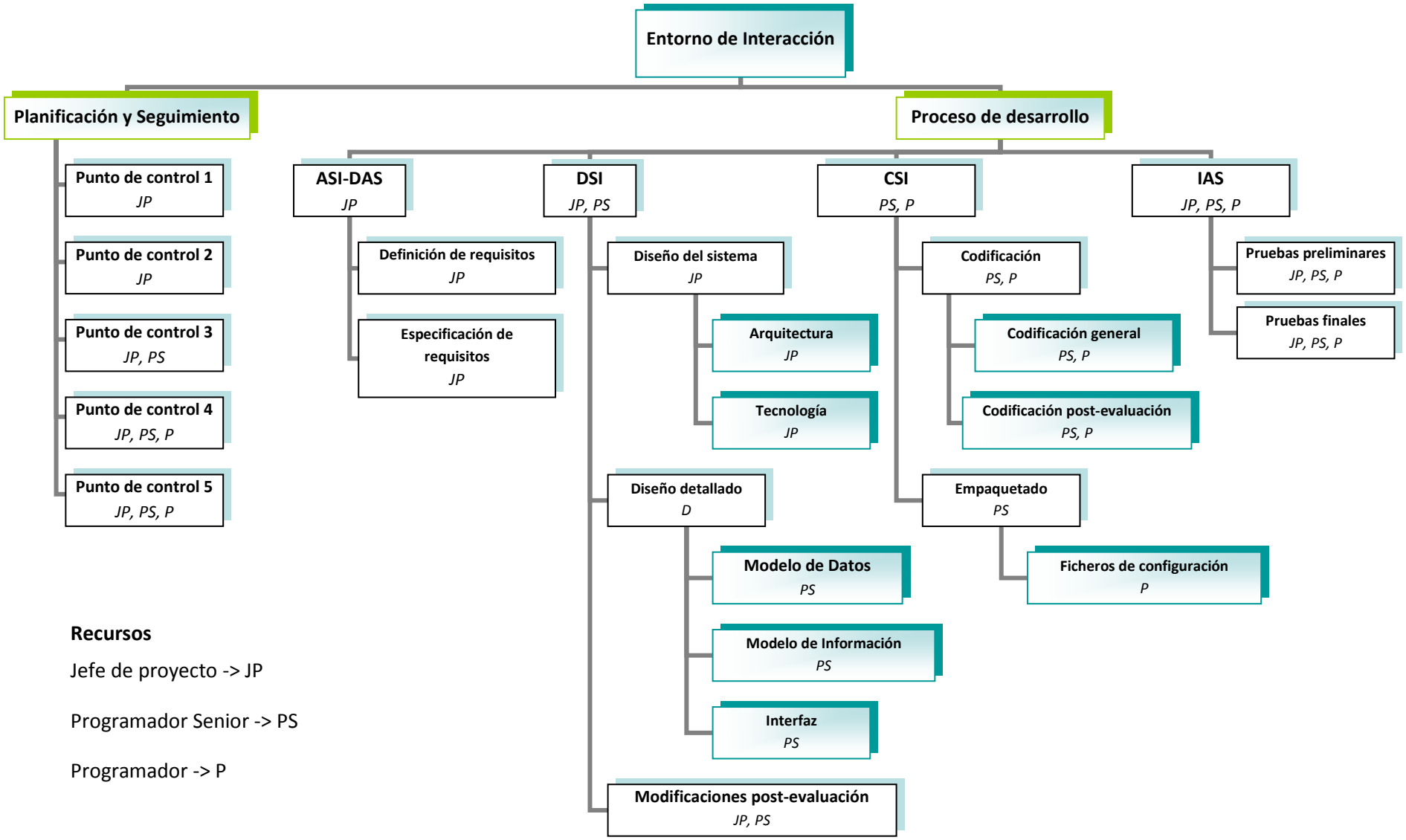


Ilustración 13.Diagrama de Gantt



Recursos

Jefe de proyecto -> JP

Programador Senior -> PS

Programador -> P

Ilustración 14. Jerarquía de Tareas

3.5 Presupuesto

El presupuesto final por el desarrollo del proyecto descrito en este documento contempla: salarios de los trabajadores, equipos informáticos, material fungible y otros costes asociados con el desarrollo de la misma. A continuación se detallan el presupuesto del proyecto detallando el origen de cada coste.

En la Ilustración 15 se detallan las horas empleadas en la realización de cada tarea por cada recurso personal. Además se muestra el costo de la realización de las tareas y el costo total de la realización del proyecto por parte del personal. Por último en la Ilustración 15, también se muestra un resumen del número de horas mensuales que ha dedicado al proyecto. Basado en estos datos en la Tabla 1 se hace un resumen de los costes totales de cada recurso personal al proyecto.

En la Tabla 2 se hace un resumen de los costes asociados a los recursos físicos del proyecto. Como recursos físicos se han utilizado:

- Dos ordenadores que se han utilizado para la implementación del proyecto, el desarrollo de la memoria y la ejecución del entorno y el sistema externo.
- Una estantería y una lámina de plexiglás que darán forma al entorno. La lámina de plexiglás será la superficie de reconocimiento.
- La iluminación está compuesta de unas bombillas que creen el equilibrio en la iluminación del entorno para que el reconocimiento de los objetos sea de la forma más precisa y eficiente posible.
- Una impresora con la cual se imprimen las etiquetas que después se añadirán a los objetos y permitirá al entorno identificar las propiedades de los objetos.
- El set de figuras son los objetos que serán utilizados como elementos de interacción y servirán para completar los desafíos presentados por los juegos.
- La webcam es el elemento de reconocimiento de las etiquetas a través de la cual se conocerán las propiedades de los objetos físicos que se utilizarán como elementos de interacción con el entorno.

En la Tabla 3, se puede observar los costes relativos al software. Como software se ha utilizado:

- El sistema operativo en el cual se ha desarrollado el proyecto y en el que se ejecuta la parte del entorno es Windows 7 Professional. El sistema externo podría ejecutarse en cualquier sistema operativo siempre y cuando respetase el protocolo de comunicación definido para el entorno.
- Microsoft Office 2010, es la herramienta de ofimática utilizada para poder desarrollar la documentación necesaria para la completa elaboración del proyecto.
- Como herramienta case se ha decidido la utilización de Altova UModel 2011 Enterprise Edition. Esta herramienta ha permitido la elaboración de los diagramas de componentes de la memoria del proyecto.
- Netbeans IDE 7.0, es la herramienta de implementación elegida para el desarrollo del sistema externo y la modificación del código fuente del proyecto Trackmate, permitiendo la adaptación del mismo al proyecto de fin de carrera.

- TrackMate, es el proyecto en el que se basa el PFC.

En la Tabla 4, se especifican los costes del material de oficina utilizado en el desarrollo del proyecto de fin de carrera. Para terminar con la parte de los costes en la Tabla 5 se hace una recopilación de todos los gastos del proyecto ofreciendo el costo total del proyecto.

Teniendo en cuenta los riesgos que pueden ocurrir, la empresa debe ser capaz de afrontarlos sin tener que cobrar más al cliente, por eso se incluye un porcentaje del 15% para imprevistos. Por último, hay que incluir un margen de beneficios para la empresa que dado la duración y dificultad de la aplicación será de un 20%. El presupuesto total para la realización del proyecto viene explicado en la Tabla 6.

Nombre del recurso	Trabajo	Costo	may	jun	jul
Jefe de Proyecto	293 horas	13.185,00 €	138h	66h	89h
Punto de Control 1	*1 hora*	*45,00 €*	1h		
Punto de Control 2	*1 hora*	*45,00 €*	1h		
Punto de Control 3	*1 hora*	*45,00 €*	1h		
Punto de Control 4	*1 hora*	*45,00 €*		1h	
Punto de Control 5	*1 hora*	*45,00 €*			1h
Definición de Requisitos	*40 horas*	*1.800,00 €*	40h		
Espicificación de Requisitos	*48 horas*	*2.160,00 €*	16h	32h	
Arquitectura	*56 horas*	*2.520,00 €*	23h	33h	
Tecnología	*56 horas*	*2.520,00 €*	56h		
Modificaciones Post-Evaluación	*28 horas*	*1.260,00 €*			28h
Pruebas Preliminares	*20 horas*	*900,00 €*			20h
Pruebas Finales	*40 horas*	*1.800,00 €*			40h
Programador Senior	300 horas	9.000,00 €	76h	105,37h	118,63h
Punto de Control 3	*1 hora*	*30,00 €*	1h		
Punto de Control 4	*1 hora*	*30,00 €*		1h	
Punto de Control 5	*1 hora*	*30,00 €*			1h
Definición de Requisitos	*20 horas*	*600,00 €*	16h	4h	
Espicificación de Requisitos	*24 horas*	*720,00 €*	4h	20h	
Modelo de Datos	*39 horas*	*1.170,00 €*	39h		
Modelo de Información	*40 horas*	*1.200,00 €*	16h	24h	
Interfaz	*32 horas*	*960,00 €*		32h	
Modificaciones Post-Evaluación	*15 horas*	*450,00 €*			15h
Codificación General	*43 horas*	*1.290,00 €*		24,37h	18,63h
Codificación Post- Evaluación	*24 horas*	*720,00 €*			24h
Pruebas Preliminares	*20 horas*	*600,00 €*			20h
Pruebas Finales	*40 horas*	*1.200,00 €*			40h
Programador	206 horas	2.564,70 €		89h	117h
Punto de Control 4	*1 hora*	*12,45 €*		1h	
Punto de Control 5	*1 hora*	*12,45 €*			1h
Codificación General	*140 horas*	*1.743,00 €*		88h	52h
Versión Beta	*1 hora*	*12,45 €*			1h
Codificación Post- Evaluación	*20 horas*	*249,00 €*			20h
Ficheros de Configuración	*3 horas*	*37,35 €*			3h
Pruebas Preliminares	*20 horas*	*249,00 €*			20h
Pruebas Finales	*20 horas*	*249,00 €*			20h

Ilustración 15. Tabla del costo y uso de los recursos personales

Cargo	Euros/Hora	Horas/Proyecto	Bruto Proyecto
Jefe de proyecto/Analista/Arquitecto	40 €	293 h	13.185,00€
Diseñador/Desarrollador	30 €	300 h	9.000,00 €
Desarrollador	12,45 €	206 h	2.564,00 €
		Total:	**24.749,00 €**

Tabla 1. Salario bruto mensual.

Descripción	Coste/Unidad	Unidades	Coste Total
Portátil Dell xps 1640	1.582,56	1	1.582,56
Hacer Vertion Core 2 Quad 8400	823,34	1	823,34
Estantería Cromada	105,23	1	105,23
Lámina de plexiglas	31,52	1	31,52
Iluminación	34,10	1	34,10
Set de figuras	7,99	1	7,99
HP Laserjet 2055DN	242,90	1	242,90
WebCam HD 720p HP	43,52	1	43,52
		Total:	**2.871,16 €**

Tabla 2. Coste de equipos y dispositivos.

Descripción	Coste/Unidad	Unidades	Coste Total
Windows 7 Professional	309,00	1	309,00
Microsoft Office 2010	699,00	1	699,00
Altova UModel 2011 Enterprise Edition	199,00	1	199,00
Netbeans IDE 7.0	0	1	0
TrackMate	0	1	0
		Total:	**1.207,00 €**

Tabla 3. Coste de software.

Descripción	Coste/unidad	Unidades	Coste Total
Material fungible	N/A	N/A	200,00
		Total:	**200,00 €**

Tabla 4. Costes de material de oficina.

Descripción	Coste	Impuesto	Total impuestos	Total
Implementación	33.712,41	-	-	33.712,41
Instalación	2.500,00	-	-	2.500,00
Software	1.207,00	18 %	217,26	1.424,26
Equipos y dispositivos	2.871,16	18 %	516,80	3.387,96
Material fungible	200,00	18 %	36,00	236,00
Total:			**770.06 €**	**41.260,63€**

Tabla 5. Costes totales.

Descripción	Coste
Coste total	41.260,63€
Beneficio	8.252,13
Riesgos	6.189,1 €
Total	**55.701,86**

Tabla 6. Cálculo total.

4 Elaboración de la Solución

En este apartado se explicará la solución que ha sido elegida, aportando una serie de requisitos que se deberán cumplir para satisfacer las distintas funcionalidades. A continuación, se expondrá el diseño que respete los requisitos establecidos. Por último se realizará la implementación.

4.1 Descripción

Este proyecto propone una solución a diferentes problemas que se pueden encontrar los usuarios inexpertos desde el punto de vista tecnológico. Para ello se diseñará un entorno que ofrecerá un tipo de interacción que permita al usuario inexperto familiarizarse rápidamente con la manera de realizar las diferentes acciones sobre el mismo. A la hora de realizar las acciones, para poder lograr un objetivo, el usuario puede encontrar ciertas dificultades tecnológicas, que nacen de la traducción del objetivo a las acciones específicas requeridas por la interfaz de usuario para llegar a dicha meta. Estas dificultades se encuentran en la fase de ejecución y en la fase de evaluación del modelo de interacción definido por Norman [8]. Los problemas que pudieran surgir en la fase de ejecución del modelo nacen en la denominada Brecha de Ejecución definida como *"la discrepancia entre la representación mental que tenemos de la acción y la que nos ofrece la interfaz del sistema"* [4]. El resto de inconvenientes provienen de la fase de evaluación del modelo a través de la Brecha de Evaluación definido como *"el grado en que el sistema/artefacto ofrece representaciones que pueden ser directamente percibidas e interpretadas en términos de las expectativas y las intenciones del usuario"* [8]. Este problema es el que se refleja en la Ilustración 16. Esta distancia se hace más notable cuando el usuario que trata de utilizar el dispositivo es un usuario inexperto, desde el punto de vista de la tecnología, como por ejemplo un grupo de niños con edades comprendidas entre 3 y 6 años.

Para ello se quiere desarrollar un entorno cuya interacción sea a través de acciones básicas o elementales para el usuario, acciones naturales que no tenga que aprender, que le sean innatas. La interacción elegida para la solución es la interacción tangible, dentro del paradigma Invisible Computing. Con este paradigma se elimina la interfaz, eliminando a su vez la posible distancia entre el diseño y la interpretación que hagan este tipo de usuarios sobre el mismo. Se ha elegido esta interacción porque las acciones de coger elementos y soltarlos en una superficie o arrastrarlos por esa misma, son movimientos que se aprenden desde muy pequeños y que no resultan desconocidas al grupo de usuarios, para el que está pensado el proyecto, pudiendo así reducir considerablemente la brecha de ejecución con el sistema. Otro problema, este desde el punto de vista técnico, es cómo reconocer los objetos que el usuario coloca sobre la superficie. Para solucionar este detalle se optar por el etiquetado de los objetos. Ese etiquetado contiene la información necesaria del objeto, descrita más adelante, permitiendo saber al sistema de que objeto se trata y cuáles son sus propiedades que se han definido esenciales. El formato de la etiqueta así como la técnica de reconocimiento han sido cogidos del proyecto del MIT [9]. El modelo de datos ha sido diseñado en función de las necesidades del prototipo. Éste se explicará con posterioridad.

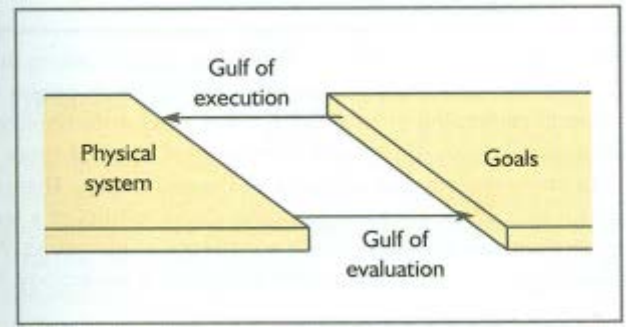


Ilustración 16.Brecha entre el sistema y los objetivos [5]

4.2 El proceso de desarrollo

En este capítulo se definirán todos los requisitos de usuario. Tomando como base estos requisitos se definirán los requisitos de sistema, que serán la base para poder diseñar e implementar el sistema. Una vez realizada la implementación se realizarán las pruebas para garantizar que se cumplen todos los requisitos esenciales redactados.

4.2.1 Análisis

En la fase actual se dará forma al sistema que se quiere desarrollar. Se definirán todos los requisitos que son necesarios para que el sistema cumpla con los objetivos buscados.

4.2.1.1 Definición de requisitos

En este apartado se van a definir los requisitos basándose en las necesidades del proyecto. Los requisitos se definirán en dos tipos:

- **Requisitos de Usuario**: especifican las características que necesita el sistema desde el punto de vista del usuario.
- **Requisitos del Sistema**: una vez analizados los requisitos de usuario, se especifican la forma en que el sistema debe alcanzar los objetivos y realizar las distintas funciones.

Cada requisito contará con una serie de campos que lo definen. Los campos que contiene cada requisito son los siguientes:

- **Identificador**: valor único que identifica de manera unívoca al requisito.
- **Nombre**: identificador textual breve que aporta significado al requisito.
- **Prioridad**: grado de importancia del requisito en el proceso de diseño e implementación.
- **Necesidad**: grado de compromiso a la hora de incluir el requisito en el producto final.
- **Verificabilidad**: grado de dificultad a la hora de comprobar si el requisito ha sido implementado de forma correcta.
- **Descripción**: definición textual completa del requisito.

4.2.1.1.1 Requisitos de usuario

En este apartado se especificarán los requisitos del sistema obtenidos a través de diferentes técnicas como la observación, test y entrevistas realizadas a un grupo de usuarios inexpertos.

RE-US-01			
Nombre	Reconocimiento de figuras geométricas.		
Prioridad	[ ]Alta [X]Media [ ]Baja	**Necesidad**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El sistema deberá ser capaz de reconocer figuras geométricas.		

Tabla 7. RE-US-01 – Juego de reconocimiento de figuras geométricas.

RE-US-02			
Nombre	Reconocimiento de elementos de una cocina.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El sistema deberá ser capaz de reconocer objetos que se puedan encontrar en una cocina.		

Tabla 8. RE-US-02 – Juego de reconocimiento de colores.

RE-US-03			
Nombre	Reconocimiento de objetos matemáticos.		
Prioridad	[ ]Alta [X]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El sistema deberá ser capaz de reconocer los objetos con formas de dígitos del 0 al 9 y con los signos +, -, *, :.		

Tabla 9. RE-US-03 – Juego de operaciones matemáticas.

RE-US-04			
Nombre	Propiedades de los objetos		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El entrono deberá reconocer las siguientes propiedades de los objetos: color, tipo, contenido, riesgo, material, forma.		

Tabla 10. RE-US-04 – Propiedades de los objetos físicos.

RE-US-05			
Nombre	Reconocimiento de varios objetos físicos simultáneamente		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El sistema debe poder reconocer tantos objetos físicos simultáneamente como elementos quepan en la superficie de detección		

Tabla 11. RE-US-05 – Reconocimiento de varios objetos físicos.

RE-US-06			
Nombre	Detener el reconocimiento de objetos físicos		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El sistema tiene que tener un estado por el cual el objeto físico deje de ser analizado, reconocido o identificado.		

Tabla 12. RE-US-06 – Detención de detección de objetos.

RE-US-07			
Nombre	Uso del protocolo de comunicación.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad**	[X]Esencial [ ]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	El sistema deberá contar con un sistema de comunicación entre la aplicación de reconocimiento de etiquetas y el sistema al que va dirigida la información.		

Tabla 13. RE-US-07 Protocolo de comunicación.

RE-US-08			
Nombre	Log de estado del sistema.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[X]Esencial [ ]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [X]Media [ ]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	La aplicación creará un log de estado del sistema en el que se guardarán: • Trazas de errores. • Acciones realizadas. • Otros datos importantes.		

Tabla 14. RE-US-08 – Log de estado del sistema.

RE-US-09			
Nombre	Manual de usuario.		
Prioridad	[ ]Alta [X]Media [ ]Baja	**Necesidad**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		

Descripción	El sistema deberá funcionar de forma correcta, ofreciendo todas las funciones definidas en los requisitos de software.

Tabla 15. RE-US-09 – Manual de usuario.

RE-US-10			
Nombre	Elementos de iluminación		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	Los elementos de iluminación deberán estar colocados en un lugar no accesible por los usuarios		

Tabla 16. RE-US-10 – Elementos de iluminación.

4.2.1.1.2 Requisitos de Sistema

Una vez especificados y analizados los requisitos obtenidos del usuario del apartado anterior se definirán los requisitos desde el punto de vista técnico para la elaboración del sistema.

RE-S-01			
Nombre	Figuras geométricas.		
Prioridad	[ ]Alta [X]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El entorno deberá contar con figuras geométricas tales como círculos, rectángulos, triángulos, cuadrados.		

Tabla 17. RE-S-01 – Juego de reconocimiento de figuras geométricas.

RE-S-02			
Nombre	Objetos de diferentes colores.		
Prioridad	[ ]Alta [X]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El entorno deberá contar con objetos físicos de diferentes colores.		

Tabla 18. RE-S-02 – Juego de reconocimiento de colores.

RE-S-03			
Nombre	Objetos matemáticos.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El entorno deberá contar con objetos con formas de dígitos del 0 al 9 y con los signos +, -, *, :.		

Tabla 19. RE-S-03 – Objetos matemáticos.

RE-S-04			
Nombre	Objetos de cocina.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Media. Puede cambiar en función de las necesidades de los sistemas externos a desarrollar.		
Descripción	El entorno deberá contar con objetos con formas de dígitos del 0 al 9 y con los signos +, -, *, :.		

Tabla 20. RE-S-04 – Objetos de cocina.

RE-S-05			
Nombre	Condiciones de Iluminación		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad:**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	El entorno debe tener las condiciones de iluminación adecuadas para posibilitar la correcta detección de los objetos físicos.		

Tabla 21. RE-S-05 – Etiquetado de los objetos.

RE-S-06			
Nombre	Log de estado del sistema.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad**	[X]Esencial [ ]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	El sistema generará archivos con información sobre el estado del sistema. Estos archivos ayudarán a realizar un mantenimiento y corrección de errores más preciso.		

Tabla 22. RE-S-06 – Log de estado del sistema.

RE-S-07			
Nombre	Log errores internos.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad**	[X]Esencial [ ]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	El sistema generará archivos donde guardará toda la información cuando se produce un error. Deberá almacenar la fecha, hora y traza del error.		

Tabla 23. RE-S-07– Log errores internos.

RE-S-08			
Nombre	Lenguaje de programación C++.		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		

Descripción	El desarrollo del procesamiento de imágenes para la identificación de etiquetas se realizará en C++.

Tabla 24. RE-S-08 – Lenguaje de programación C++.

RE-S-09			
Nombre	Protocolo de comunicación LusidOSC		
Prioridad	[X]Alta [ ]Media [ ]Baja	**Necesidad**	[ ]Esencial [X]Deseable [ ]Opcional
Verificabilidad	[ ]Alta [ ]Media [X]Baja		
Estabilidad	Alta. Durante toda la vida del sistema.		
Descripción	La comunicación entre el entorno y los sistemas a los que se encuentre conectado.		

Tabla 25. RE-S-08 – Protocolo de Comunicación LusidOSC.

4.2.2 Diseño

En este apartado se realizará una descomposición del sistema en componentes más pequeños, denominados subsistemas. Cuanto mayor sea la descomposición del sistema, más sencillos serán los subsistemas resultantes, aunque de esta forma aumenta la complejidad de las interfaces que los interrelacionan. Por ello, se optará por una solución de compromiso. En este proyecto se empleará como criterio de división la funcionalidad. De esta manera, cada subsistema cubrirá unas necesidades del entorno. Para realizar el diseño del sistema se utilizará el método de diseño UML y como herramienta de diseño se utilizará la herramienta Altova UModel 2011.

4.2.2.1 Diseño de sistema

En este apartado se van a especificar la arquitectura con la que será implementado el sistema y las diferentes tecnologías utilizadas.

4.2.2.1.1 Arquitectura

En este apartado se establecerán las guías generales que se utilizarán para el desarrollo del sistema. Para independizar la comunicación de las operaciones interactivas, y todo esto a su vez de la presentación al usuario, se va a implementar el patrón arquitectónico modelo-vista-controlador (MVC), con un añadido para implementar la comunicación de la infraestructura externa, es decir, el entorno, que es de donde la aplicación se encargará de obtener la información necesaria para saber qué acciones está realizando el usuario, con el sistema receptor de la información. Para la implementación de nuestra solución se desarrollará una aplicación con Java Swing de la cual no se especificará diseño alguno al no considerarse parte de la solución sino sólo un añadido para comprobar la correcta comunicación entre el entorno y un sistema ajeno al mismo. Debido a que el proyecto incorpora aspectos de interacción con objetos físicos el patrón de diseño MVC se debe adaptar a esta situación haciendo una división de los subsistemas en el ámbito físico y digital. De esta forma se optará por la utilización de una evolución del patrón MVC para interfaces tangibles, el Modelo-Controlador-Representación_Fisica-Representación_Digital (MCRpd). En esta evolución del patrón se tienen en cuenta la representación del objeto tanto en el plano físico como una posible

representación en el plano digital. La separación de los distintos subsistemas en los planos físicos y digital queda representada en la Ilustración 17.

Este modelo divide el sistema en cuatro subsistemas que son los siguientes:

- **Representación**: Este subsistema es el que sufre una mayor modificación con respecto al modelo original MVC-I. El subsistema de divide a la vez en dos: la representación física del objeto (plano físico), y la representación digital del objeto (plano digital).
- **Controlador:** capa encargada de reaccionar ante los eventos ocurridos en la aplicación, ya sean acciones del usuario, cambios en el negocio o en la vista física.
- **Modelo:** esta capa se encargará de la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.
- **Infraestructura**: capa a través de la cual la aplicación se va a comunicar con la infraestructura externa, en este caso con el entorno tangible, donde se genera la información que se transmitirá al plano digital del sistema.

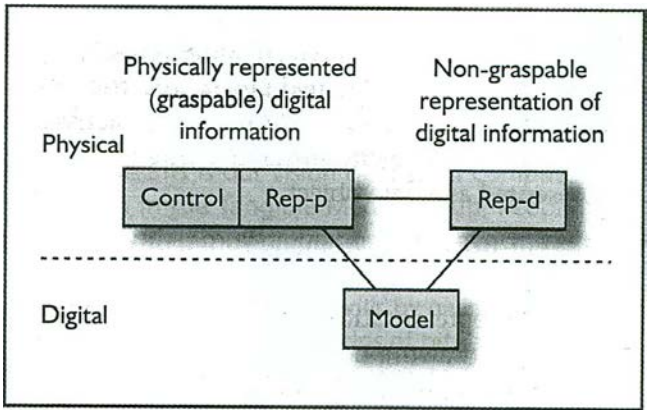


Ilustración 17. Modelo MCRpd [5]

A continuación se especificarán los subsistemas de los que se compone la aplicación:

4.2.2.1.1.1 Subsistema Vista

SUB-01	Vista Física
Funciones	• Objetos físicos con los que el usuario puede interactuar de manera tangible con el sistema. • Varía en función de los juegos en los que quiera participar el usuario.

Tabla 26. Subsistema Vista Física.

4.2.2.1.1.2 Subsistema Controlador

SUB-03	Controlador
Funciones	• Procesa las peticiones recibidas por el subsistema de la vista física.

Tabla 27. Subsistema Controlador.

4.2.2.1.1.3 Subsistema Negocio

SUB-04	Modelo
Funciones	• Se encarga de proporcionar la información que tiene que enviar el subsistema infraestructura. • Reconocimiento de objetos físicos.

Tabla 28. Subsistema Modelo.

4.2.2.1.1.4 Subsistema Infraestructura

SUB-04	Infraestructura
Funciones	• Se encarga enviar la comunicación del entorno al juego.

Tabla 29. Subsistema Infraestructura.

El módulo del subsistema infraestructura son los siguientes:

- **Módulo Comunicación**: desde este módulo se gestionará toda la comunicación entre sistema externo y el entorno.

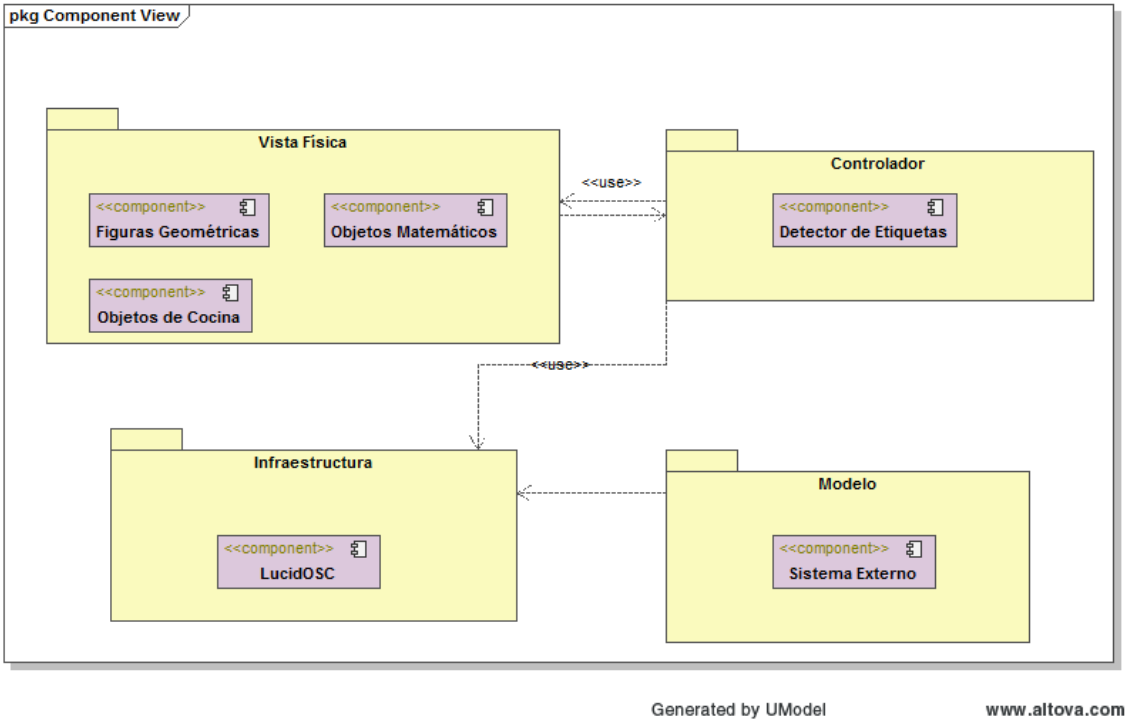


Ilustración 18. Diagrama de Subsistema

4.2.2.1.2 Tecnología

Para el desarrollo del proyecto se usarán diferentes tecnologías que ayudaran al desarrollo de los diferentes ámbitos de los que se compone el proyecto:

- **Comunicación:** para la comunicación entre el entorno y un sistema externo, se realizará por medio del protocolo de comunicación, layer for unique spatial input devices (LusidOSC), *"Capa para un espacio único de dispositivos de entrada"* [10], tal y como aparece reflejado en la Ilustración 19
- **Reconocimiento de Objetos:** para la identificación y reconocimiento de objetos se realizará a través del uso de etiquetas codificadas. El diseño de la etiqueta será el propuesto por el proyecto TrackMate [9]. La codificación será explicada en el próximo apartado.
- **Reconocimiento de Etiquetas:** para el reconocimiento de las etiquetas que identificarán los objetos físicos y algunas de sus propiedades, se utilizará una cámara de alta definición y el software diseñado en el proyecto TrackMate [11].
- **Sistema Externo:** como sistema externo para comprobar la comunicación entre entorno a desarrollar y otro sistema externo se desarrollará, a modo de prototipo, un juego implementado con la tecnología java swing basado en el reconocimiento de las distintas propiedades de los objetos físicos.
- **Generador de etiquetas:** tomando como base el código fuente del generador de etiquetas proporcionado por el proyecto Trackmate [12], se creará un programa modificado para generar las etiquetas de manera más automática, sencilla y específica en la información necesaria para el proyecto fin de carrera.

Ilustración 19. Protocolo de comunicacíon LusidOSC [13].

4.2.2.2 Diseño detallado

Debido a que el proyecto de fin de carrera no es un proyecto software muchos de los pasos en el diseño, tales como diagrama de clases, diagramas de secuencia, diagramas de paquetes etc.., nos son posibles. Por ello dentro de este apartado sólo se especificará el diseño del modelo de datos utilizado en la codificación de las etiquetas de los objetos físicos. El software dedicado al reconocimiento de las etiquetas, se utilizará el proporcionado en el proyecto trackmate cuyo nombre es *Trackmate-tracker.* Por otro lado, como se ha comentado en el apartado anterior, para comprobar la correcta comunicación con un sistema externo se desarrollará un prototipo que consiste en una serie de juegos para reconocer diferentes propiedades de los objetos físicos. Al no ser parte de la solución propuesta para el problema no se realizará ninguna especificación sobre el diseño y la implementación del mismo.

4.2.2.2.1 Modelo de Información

A lo largo de este apartado se describirá la codificación del modelo proporcionado por el proyecto del trackmate. Como el proyecto está enfocado a un grupo de usuarios inexpertos y tomando como referencia de los mismos niños con edades comprendidas entre los 3 y 6 años, la información que se requiere codificar son las propiedades relacionadas con su naturaleza física, su uso y su utilidad. Esas propiedades son: color, tipo, material, forma, contenido, riesgo. Las etiquetas además estarán provista de un campo checksum que comprobará si la información recibida ha sufrido alguna alteración durante el envío.

La etiqueta tiene la estructura que se puede ver en la Ilustración 20. Esta se divide en 3 sectores principalmente: el color, la orientación y la información. El color viene determinado por los 4 círculos externos de la etiqueta marcados con el identificador desde el a.0 al a.3. Cada círculo puede tomar valores entre 0 y 255 en función de la escala de grises con la que esté rellenado. De esta forma juntando la información de los tres primeros círculos puedes obtener la escala de colores RGB. Aunque la etiqueta ofrece esta posibilidad para identificar la propiedad física del color del objeto, como se explicará más adelante, en este proyecto se optará por codificar el color dentro de la sección de la información para minimizar las posibilidades de error. El cuarto círculo se reservará para el checksum, asegurándose de que la información es íntegra. La orientación del objeto se puede obtener en función de la posición de los rectángulos marcados con los identificadores A, B y C. Se cogerá A y C como referencia de un punto cardinal, Norte, Sur, Este u Oeste, y en función de la posición detectada por la cámara se calcula su orientación. Por último se encuentra la sección de la información. Está compuesta por 7 círculos concéntricos que, de fuera a dentro, tienen el siguiente significado: separador, cuatro bytes de información, separador, dos bytes de información más y el checksum, separador y los dos últimos círculos concéntricos determinan el centro de la etiqueta. Cada número es equivalente a un bit, si está en blanco es equivalente al valor 1 y si está en negro es equivalente al valor 0.

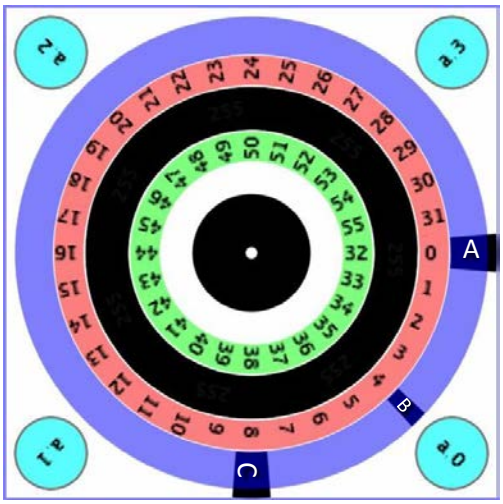


Ilustración 20. Formato de las etiquetas.

Una vez explicada la estructura se pasará a explicar el modelo de datos del proyecto. La primera propiedad codificada es el color. El color ocupa los tres primeros bytes de la sección reservada para la información. Se ha decidido incluir el color en la información y no de la sección dedicado a ello porque la decodificación resulta más fácil y porque existía ese espacio dentro del modelo de datos diseñado. Los tres bytes han sido distribuidos de la siguiente manera:

- **0 al 7:** se codifica la cantidad de azul con valores entre 0 y 255, tal y como se muestra en la Ilustración 23.
- **8 al 15:** se codifica la cantidad de verde con valores entre 0 y 255, tal y como se muestra en la Ilustración 22.
- **16 al 23:** se codifica la cantidad de rojo con valores entre 0 y 255, tal y como se muestra en la Ilustración 21.

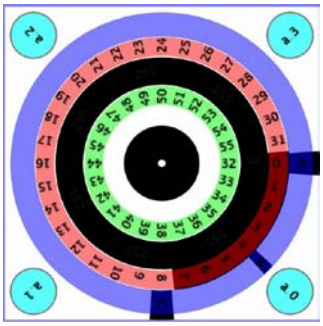


Ilustración 23. Codificación Blue.

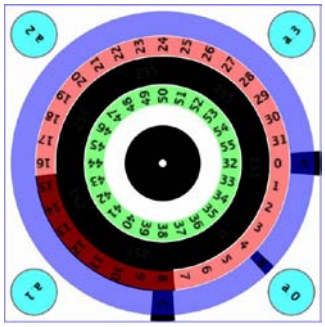


Ilustración 22. Codificación Green.

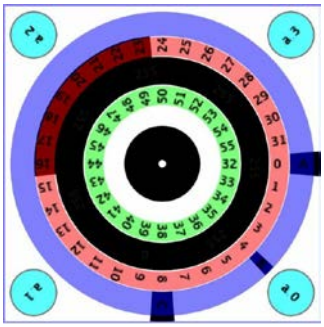


Ilustración 21. Codificación Red.

La siguiente propiedad a codificar es el tipo de objeto. Con esta propiedad se pretende dotar al entorno de la capacidad de identificarlos diferentes objetos que formen parte del sistema. Esta propiedad ocupa un byte y se encuentra codificada entre el bit 24 y 31 como muestra la Ilustración 24. Con este tamaño se pueden codificar hasta 256 objetos.

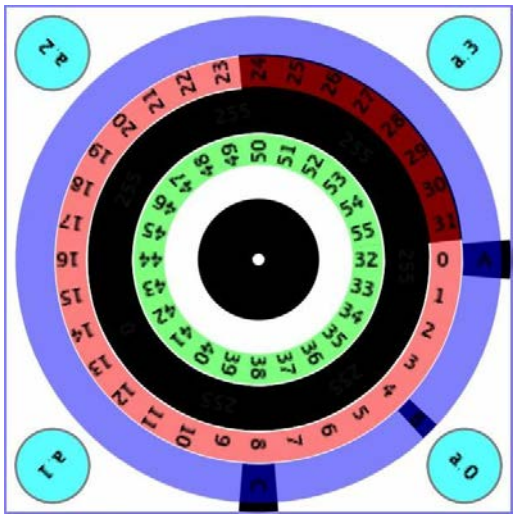


Ilustración 24. Codificación Tipo.

La siguiente propiedad a codificar es el contenido de objeto. Con esta propiedad se pretende conocer el contenido del objeto. Esto puede resultar útil porque una botella de agua puedes estar rellena de lejía lo cual es útil diferenciar. Esta propiedad ocupa cinco bits y se encuentra codificada entre el bit 32 y 36 como muestra la Ilustración 25. Con este tamaño se pueden codificar 32 contenidos diferentes.

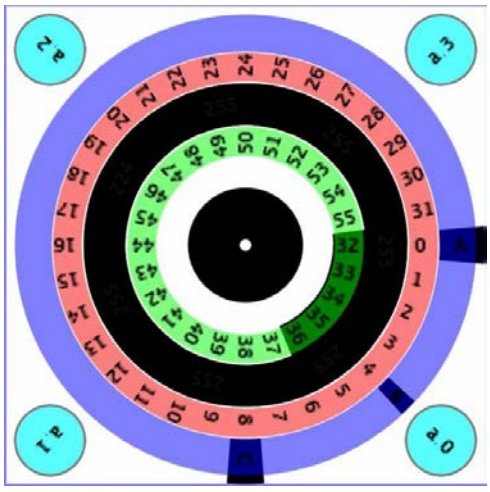


Ilustración 25. Codificación Contenido.

La siguiente propiedad a codificar es el riesgo de objeto. Esta propiedad ocupa 3 bits y se encuentra codificada entre el bit 37 y 39 como muestra la Ilustración 26. Con este tamaño se podrán codificar 8 diferentes niveles de riesgo.

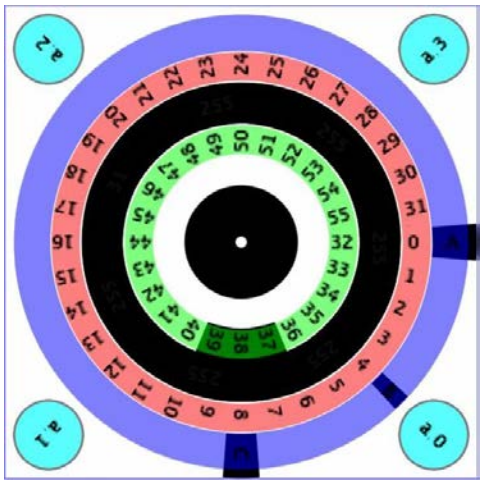


Ilustración 26. Codificación Riesgo.

La siguiente propiedad a codificar es el material de objeto. Con esta propiedad se pretende dotar al entorno de la capacidad de identificar de qué material está hecho el objeto. Con esta propiedad se podrá distinguir, por ejemplo, un cuchillo de plástico de uno de metal con el consecuente cambio en la propiedad de riesgo del objeto. Esta propiedad ocupa cinco bits y se encuentra codificada entre el bit 40 y 44 como muestra la Ilustración 27. Con este tamaño se pueden codificar 32 materiales diferentes.

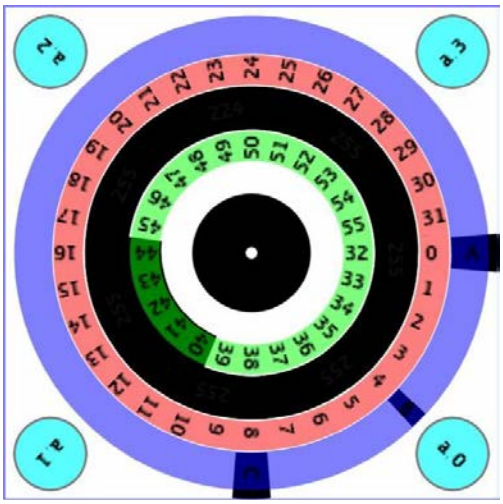


Ilustración 27.Codificación Material.

La siguiente propiedad a codificar es la forma del objeto. Esta propiedad ocupa tres bits y se encuentra codificada entre el bit 45 y 47 como muestra la Ilustración 28. Con este tamaño se podrán codificar 8 diferentes formas de objetos.

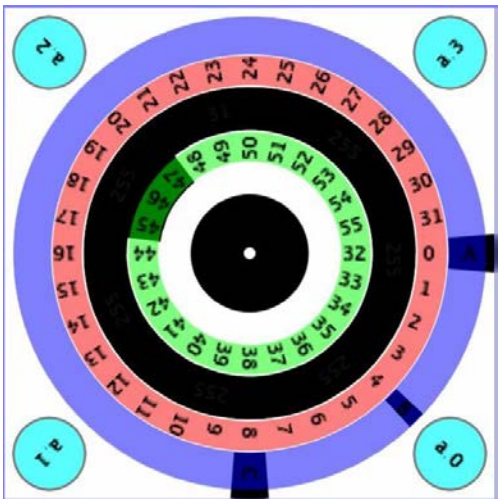


Ilustración 28.Codificación Forma.

Por último, se codificará es el checksum de objeto. Con esta codificación se pretende asegurar la integridad de los datos durante la fase de comunicación, es decir, que los datos que salen del entorno sean los mismos que los que llegan al sistema externo. Esta propiedad ocupa un byte y se encuentra codificada entre el bit 48 y 55 como muestra la Ilustración 29.

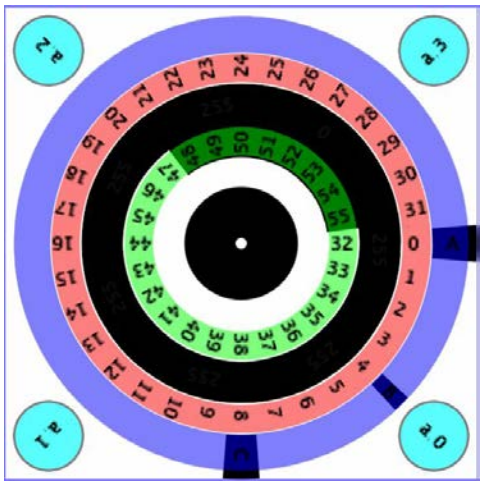


Ilustración 29. Codificación Checksum.

4.2.3 Implementación

El proyecto se divide en 3 capas principales: el entorno, la comunicación y el sistema externo (ver Ilustración 35). La capa del entorno implementa todo lo que tiene que ver con el reconocimiento de los objetos y elaboración de un identificador único para el objeto. Ese identificador se genera en función de sus propiedades. En Ilustración 30 se puede observar el algoritmo utilizado para generar ese ID a partir de las propiedades obtenidas de la etiqueta asociada al objeto. De este algoritmo cabe destacar el desplazamiento de 5 bits de las propiedades de riesgo y figura para colocar los 3 bits asociados a esa propiedad en la cabeza del bit, es decir, el bit 00100110 si le aplico un desplazamiento a la izquierda de 5 bits se transformará en 11000000.

```
private void generarID() {
    objectID[0] = objectColor.getBlue();
    objectID[1] = objectColor.getGreen();
    objectID[2] = objectColor.getRed();
    objectID[3] = kindObject.getKindObject();
    objectID[4] = contentObject.getContentObject() + (objectHazard.getHazardObject() << 5);
    objectID[5] = materialObject.getMaterialObject() + (shapeObject.getShapeObject() << 5);
    int sum = objectID[0] + objectID[1] + objectID[2] + objectID[3] + objectID[4] + objectID[5];
    objectID[6] = (0x0000FF & ~sum); // inverted checksum (inversion keeps
    // allZero error case from happening).

}
```

Ilustración 30.Generar ID de un objeto

La siguiente capa que entra en juego es la de la comunicación, LusidOSC. Esta capa es la encargada de llevar la información obtenida por el entorno y hacérsela llegar al sistema externo que esté conectado. Para ello, como se puede observar en la Ilustración 31, se arranca un servidor que recibirá la información del entorno como se muestra en la Ilustración 32. De la Ilustración 31 también hay que señalar el registro de los eventos add, update y remove, que se encargan de informar al sistema externo de los cambios de estado en el entorno.

```
public class LusidClient implements OSCListener {

        private int port = 3333;
        private OSCPortIn oscPort;

        private Hashtable objectList = new Hashtable();

        private Vector aliveObjectList = new Vector();
        private Vector newObjectList = new Vector();

        private long currentFrame = 0;
        private long currentMicroSecTime = 0;
        private boolean updateSetAndAlive = false;

        Object parent;
        Method addLusidObject, removeLusidObject, updateLusidObject;

        public LusidClient(Object parent) {
                this(parent,3333);
        }

        public LusidClient(Object parent, int port) {
                this.parent = parent;
                this.port=port;
                //parent.registerDispose(this);

                System.out.println("LusidOSC :: Version 1.0 :: Feb. 2, 2009");

                try { addLusidObject = parent.getClass().getMethod("addLusidObject", new Class[] { LusidObject.class }); }
                catch (Exception e) { System.out.println("LusidOSC :: missing or wrong addLusidObject method implementation."); }

                try { removeLusidObject = parent.getClass().getMethod("removeLusidObject", new Class[] { LusidObject.class }); }
                catch (Exception e) { System.out.println("LusidOSC :: missing or wrong removeLusidObject method implementation."); }

                try { updateLusidObject = parent.getClass().getMethod("updateLusidObject", new Class[] { LusidObject.class }); }
                catch (Exception e) { System.out.println("LusidOSC :: missing or wrong updateLusidObject method implementation."); }

                try {
                        oscPort = new OSCPortIn(port);
                        oscPort.addListener("/lusid/1.0",this); // register an OSC listener for LusidOSC v1.0
                        oscPort.startListening();
                        System.out.println("LusidOSC :: listening for messages on port "+port+"...");
                } catch (Exception ex) {
                        System.out.println("LusidOSC :: Failed to connect to port "+port);
                        System.out.println("LusidOSC :: It is probably in use by another application.");
```

Ilustración 31.Creación de la Comunicación y Listeners

```
public void sendLusidOSCData() {
    OSCBundle oscBundle = new OSCBundle();

    OSCMessage fseqMessage = new OSCMessage();
    fseqMessage.setAddress("/lusid/1.0");
    fseqMessage.addArgument("fseq");
    fseqMessage.addArgument((int) frameNumber);
    long milliSec = millis();
    fseqMessage.addArgument((int) (milliSec / 1000)); // seconds
    fseqMessage.addArgument((int) (milliSec % 1000) * 1000); // microseconds
    oscBundle.addPacket(fseqMessage);

    OSCMessage aliveMessage = new OSCMessage();
    aliveMessage.setAddress("/lusid/1.0");
    aliveMessage.addArgument("alive");
    // now add each of the objects! :)

      for(int i=0; i<objects.length; i++){
          if(objects[i].isActive){
            oscBundle.addPacket(objects[i].getSetMessage());
            aliveMessage.addArgument(objects[i].uniqueID);
          }
        }

    oscBundle.addPacket(aliveMessage);

    try {
        oscPort.send(oscBundle);
    } catch (Exception e) {
        e.printStackTrace();
    }

}
```

Ilustración 32.Envío de la información de un objeto

La última capa, para que el sistema esté completo, es el sistema externo. Es el encargado de recibir la información del entorno y actuar en consecuencia de la información recibida. La información le llega a través de los eventos que se generan en la capa de comunicación: add, updete y remove. Estos métodos reciben como argumento un objeto del tipo LusidObject (ver Ilustración 33) que entre otras informaciones contiene el ID del objeto, en la cual están codificadas las propiedades del objeto. Una vez obtenido el objeto con su ID se pasa a la etapa de decodificación del ID para obtener sus propiedades (ver Ilustración 34). Al igual que a la hora de generar el ID, se realizan dos operaciones que merecen la pena reseñar: la operación binaria &31 y el desplazamiento de 5 bits a la derecha. Con la operación binaria &31 se pretende obtener los últimos 5 bits de un byte, es decir, 10100110 & 00011111 = 00000110. Con la operación de desplazamiento de 5 bits a la derecha se pretende quedarse con los 3 primeros bits de un byte, es decir, 10100110 >> 5 = 00000101.

```
// ---------------------------------------------------------------------
// these methods are called whenever a LusidOSC event occurs.
// ---------------------------------------------------------------------
// called when an object is added to the scene
public void addLusidObject(LusidObject lObj) {
    game.addLusidObject(lObj);
    frame.getRightsCountTag().setText(String.valueOf(game.getRightNumber()));
    frame.getFailsCountTag().setText(String.valueOf(game.getWrongNumber()));
    frame.getInfoPanel().repaint();
    paintNewObject(lObj);
}

// called when an object is removed from the scene
public void removeLusidObject(LusidObject lObj) {
    game.removeLusidObject(lObj);
    frame.getRightsCountTag().setText(String.valueOf(game.getRightNumber()));
    frame.getFailsCountTag().setText(String.valueOf(game.getWrongNumber()));
    frame.getInfoPanel().repaint();
}

// called when an object is moved
public void updateLusidObject(LusidObject lObj) {
    game.updateLusidObject(lObj);
    frame.getRightsCountTag().setText(String.valueOf(game.getRightNumber()));
    frame.getFailsCountTag().setText(String.valueOf(game.getWrongNumber()));
    frame.getInfoPanel().repaint();
}
```

Ilustración 33.Eventos LusidOSC

```
public void generateDataObject(int[] idObject) {

    objectColor = new Color(idObject[2], idObject[1], idObject[0]);
    kindObject = new KindObject(idObject[3]);
    contentObject = new Content(idObject[4] & 31);
    objectHazard = new Hazard(idObject[4] >> 5);
    materialObject = new Material(idObject[5] & 31);
    shapeObject = new Shape(idObject[5] >> 5);
}
```

Ilustración 34. Generar objeto a partir de su ID

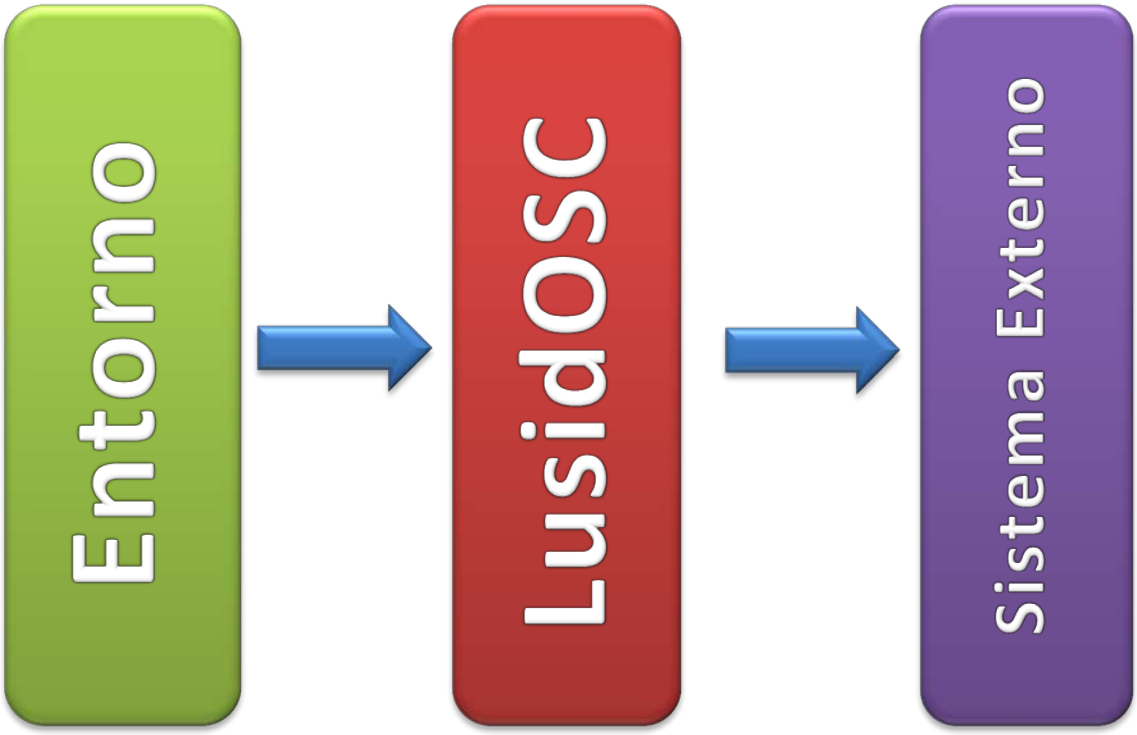


Ilustración 35.Capas del Sistema

4.2.3.1 *Herramientas*

En este apartado se detallan todas las herramientas que se han utilizado en el desarrollo del entorno y del prototipo de sistema externo desarrollado para comprobar el correcto funcionamiento de la comunicación.

Herramienta	Especificaciones
Sistema Operativo	Windows 7
IDE	Eclipse Helios Java EE IDE.
IDE	Netbeans 7.0 IDE.
IDE	Processing.
Java	Java SE 1.6
Diagramas y Modelos	Altova UModel 2011
Diagramas y Modelos	Microsoft Visio 2007
Documentación	Microsoft Word 2010
Comunicación	Protocolo LusidOSC
Calibración Entorno	TrackMate-Tracker
Control de Versión	DropBox
Sincronización de documentos	DropBox

Tabla 30. Herramientas utilizadas

Dentro de este apartado se quieren puntualizar unos aspectos de los elementos que intervienen en el proceso de la comunicación entre nuestro sistema y el sistema externo. La información detectada en la etiqueta se codificara en hexadecimal asignado un valor

hexadecimal cada 4 bits. Al enviar la información se enviará de izquierda a derecha es decir empezando a codificar por el bit 0 hasta el 55.

4.2.3.2 Organización del proyecto

Se ha utilizado una organización por carpetas en las que se indica el contenido y la fecha. Cada archivo creado contiene una versión probada y estable del sistema desarrollado.

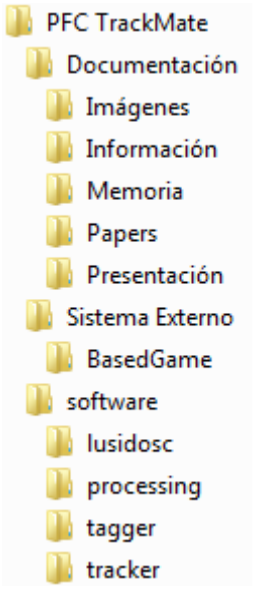


Ilustración 36. Estructura del Proyecto

La estructura del proyecto se divide en tres partes tal y como se puede observar en la Ilustración 36:

- **Documentación:** en esta carpeta se colocan toda información necesaria para el desarrollo de la documentación del proyecto y la presentación.
- **Sistema Externo:** en esta sección se encuentra el software de los sistema externos que se desarrollan para comunicarse con el entorno
- **Software:** en esta carpeta se encuentran el software de terceros utilizados para dar funcionalidad al entorno.

4.2.3.3 Despliegue

Para que el reconocimiento de las etiquetas se realice de la manera apropiada y por lo tanto el entorno tenga una funcionalidad completa se necesita una primera fase de configuración y calibración de la cámara que reconocerá dichas etiqueta. Los pasos a seguir para que dicha calibración resulte exitosa son:

- Colocación de la cámara a 31 cm de distancia con respecto a la superficie donde se colocarán los objetos para su identificación.
- Arrancar la aplicación Trackmate Tracker. Una vez arrancada comprobar que la aplicación recibe la señal de la cámara. Como se observa en la Ilustración 37, puede que desde arriba haya una iluminación excesiva, esto en un principio no debería ser

importante ya que la iluminación importante es la que haya por debajo de la superficie de identificación.

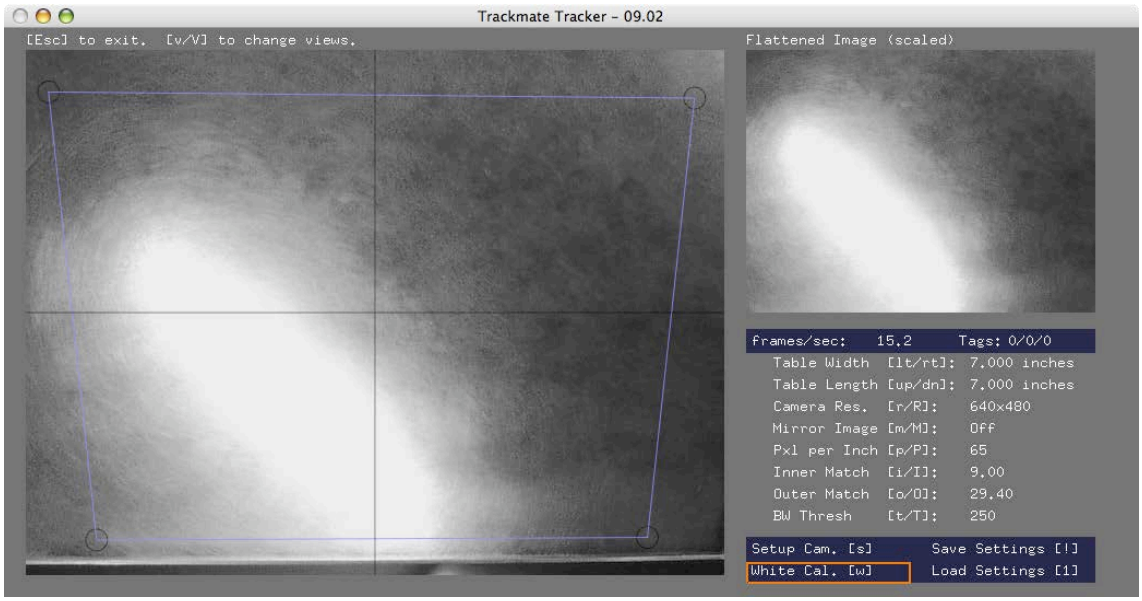


Ilustración 37. Inicio Trackmaet Tracker

- El siguiente paso es colocar la hoja de calibración tal y como se muestra en la Ilustración 38. Después colocar las cuatro esquinas del cuadrado azul en los límites de la serie de etiquetas. En la imagen plana deberá aparecer la red de etiquetas que hay en la hoja debidamente cuadrada.

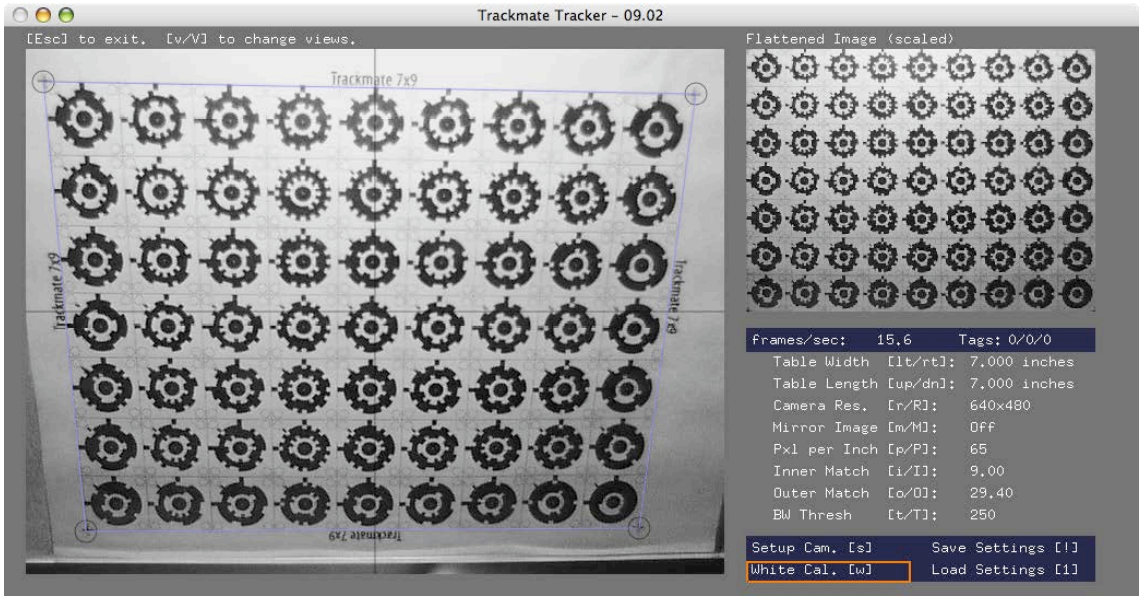


Ilustración 38.Colocación de la hoja de calibración

- Pulsando la tecla ‘v’ del teclado, pasaremos al siguiente paso de la configuración del entorno tal y como se muestra en Ilustración 39. En este paso tratará de que la cámara sea capaz de encontrar le centro de las etiquetas. Para ello se debe indicar el tamaño correcto de la red de etiqueta utilizada para la calibración. Por defecto es de una

matriz “7 x 7”. Con las teclas derecha e izquierda del teclado se configurará el ancho de la matriz y con las teclas arriba y abajo del teclado se establecerá la altura de la matriz.

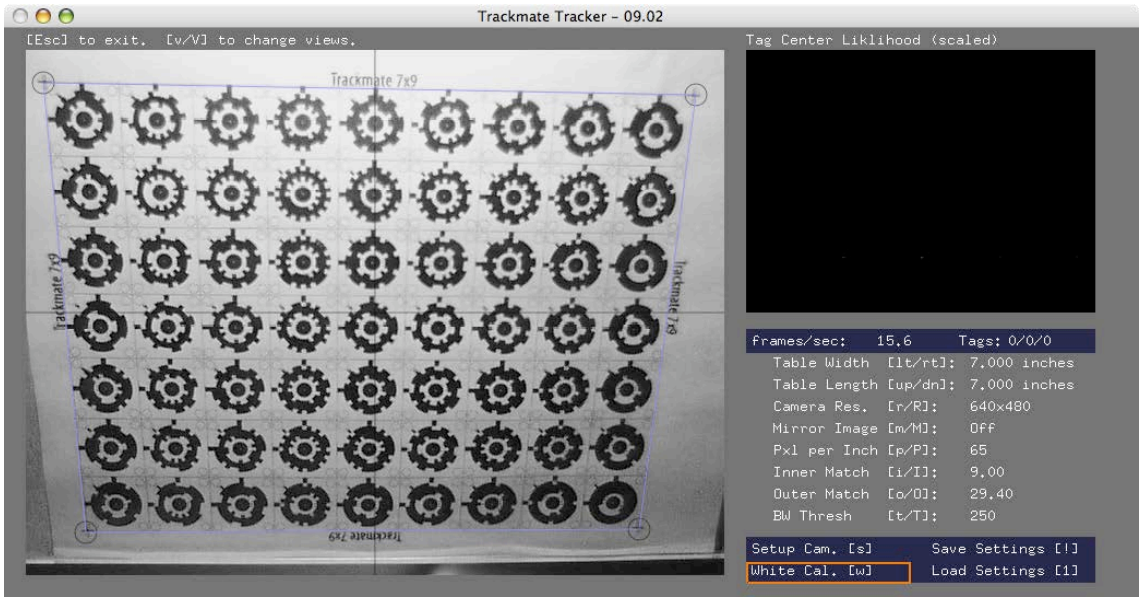


Ilustración 39. Estableciendo el tamaño de la red de etiquetas.

- Una vez configurado la correcta dimensión de la matriz de etiquetas los centros de las mismas irán apareciendo en la pantalla de la derecha tal y como se muestra en la Ilustración 40.

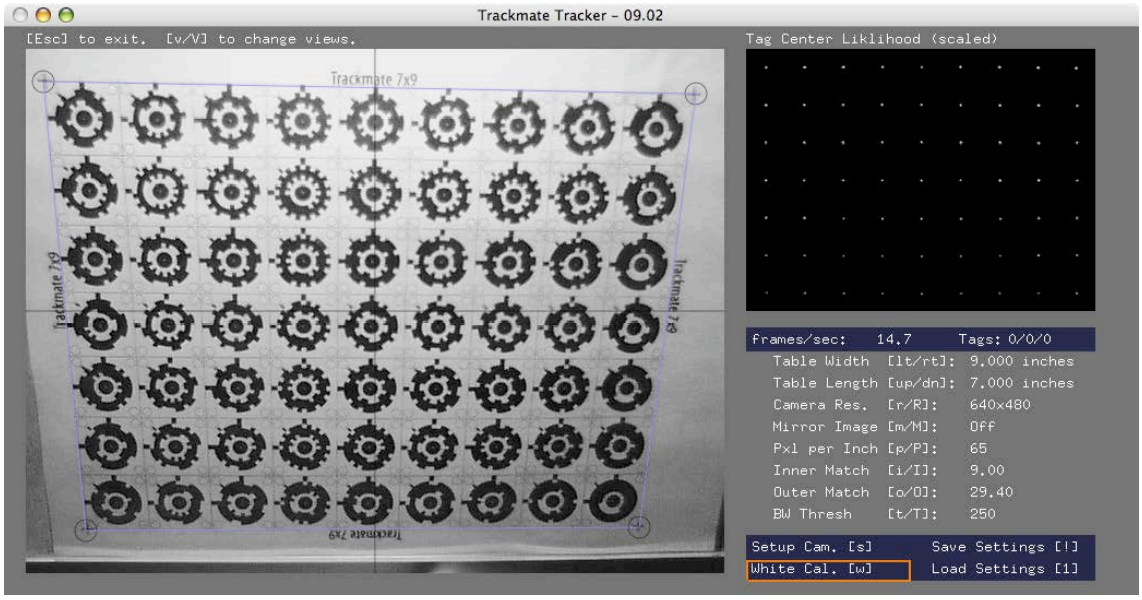


Ilustración 40.Mostrando el centro de las etiquetas

- Una vez calibrados los centros de las etiquetas se pulsará la tecla ‘v’ del teclado para acceder al siguiente paso en la configuración del entorno. En este paso se procederá a calibrar el umbral de luminosidad que está recibiendo el entorno. Como se puede observar en la Ilustración 41 sin una calibración previa hay zonas de la matriz que

reciben más luminosidad que otras por lo que se dificultaría el procedo de detección de las etiquetas.

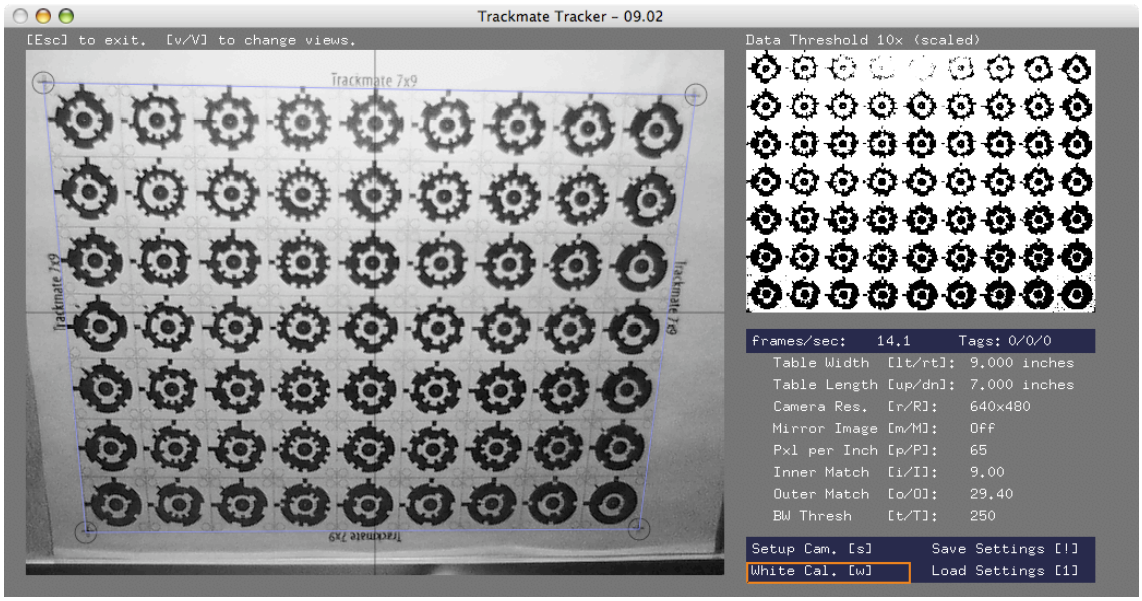


Ilustración 41. Umbral de luminosidad de la cámara.

- Pala la calibración de la luminosidad que hay en el entorno primero se coloca una hoja en blanco, tal y como se puede observar en la Ilustración 42, en la misma zona donde antes estaba colocada la hoja con la matriz de etiquetas. Después se pulsa la tecla 'w' del teclado para que la cámara realice el proceso de configuración y consiga tener una luminosidad uniforme en toda la superficie de detección.

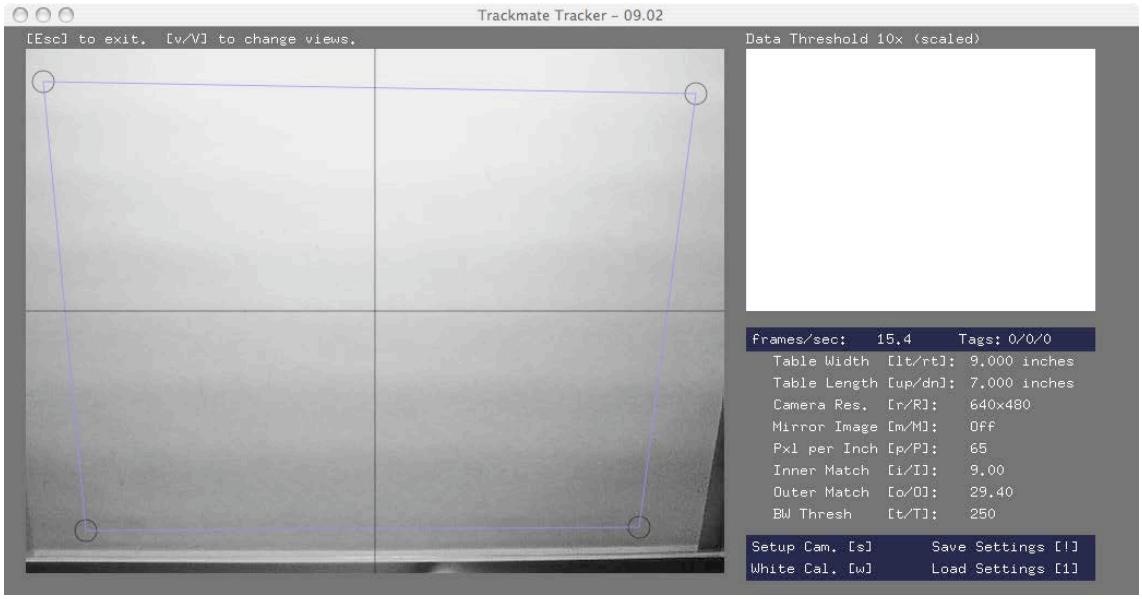


Ilustración 42. Calibrando la luminosidad del entorno.

- Una vez calibrada la luminosidad, se volverá a colocar la hoja con la matriz de etiquetas exactamente en la misma posición que estaba colocada con anterioridad. Haciendo esto obtendremos un resultado similar al obtenido en la Ilustración 43.

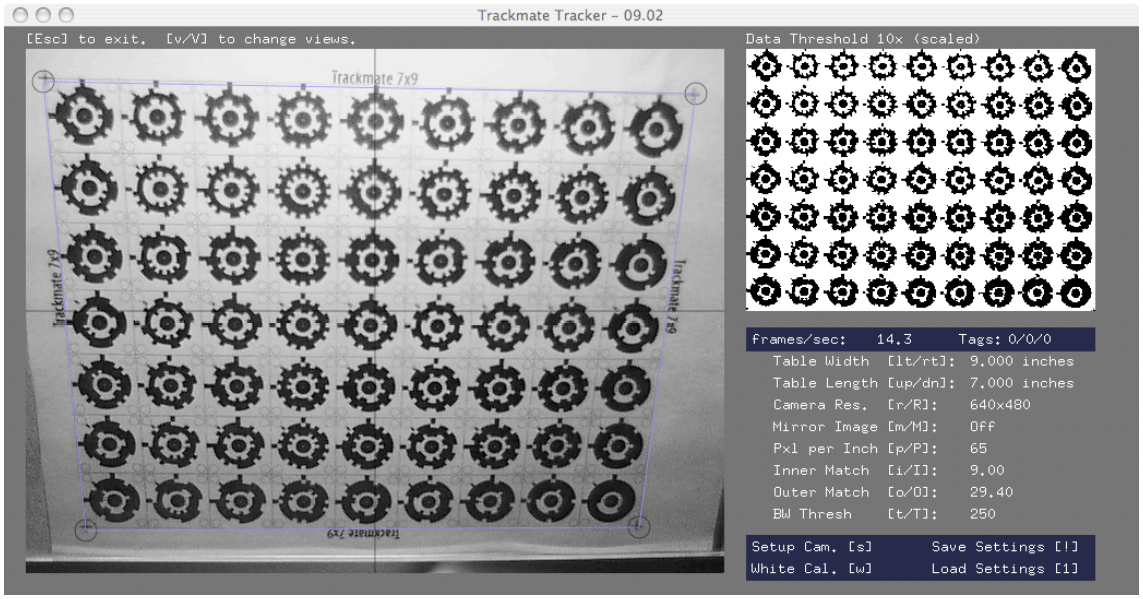


Ilustración 43. Umbral de luminosidad calibrado

- El último paso es observar cómo ha ido la configuración del entorno para ello pulsaremos otra vez la tecla 'v' del teclado. En este último paso se podrá observar algo parecido a lo mostrado en la Ilustración 44. El tracker trata de leer las etiquetas pero es posible que por la orientación de la cámara este proceso cause error. Eso es lo que significan lo cuadrados rojos, que el tracker intenta leer la etiqueta localizada en esa zona pero le es imposible. También implica, por consecuencia, que los datos de las etiquetas, que parece que leer correctamente, no son correctos. Los cuadrados azules indican que el tracker está intentando leer la etiqueta con los datos de una etiqueta anterior, que en este caso resulta ser falsa. Sólo las etiquetas con el cuadrado del centro en verde se puede determinar que tienen una información correcta. Este problema ocurre porque el tracker trata de leer la información a la inversa.

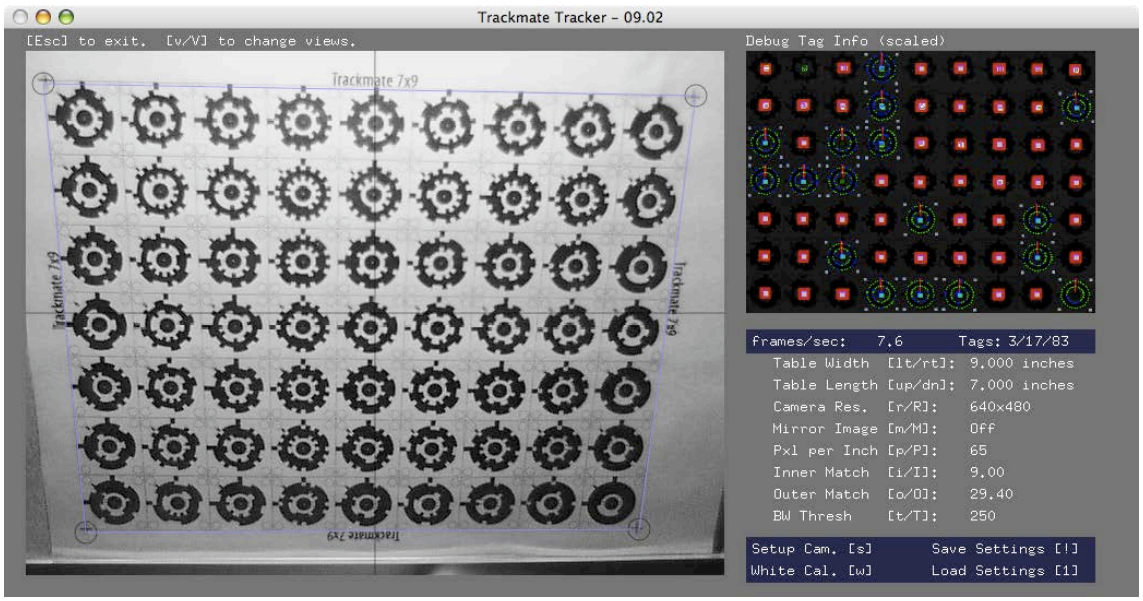


Ilustración 44. Vista inicial de la detección de etiquetas.

- Para resolver este problema pulsaremos la tecla ‘m’ del teclado lo que hará que la lectura se haga en modo espejo. El resultado debería mejorar lo anterior dando lugar a un aspecto parecido al mostrado por la Ilustración 45.

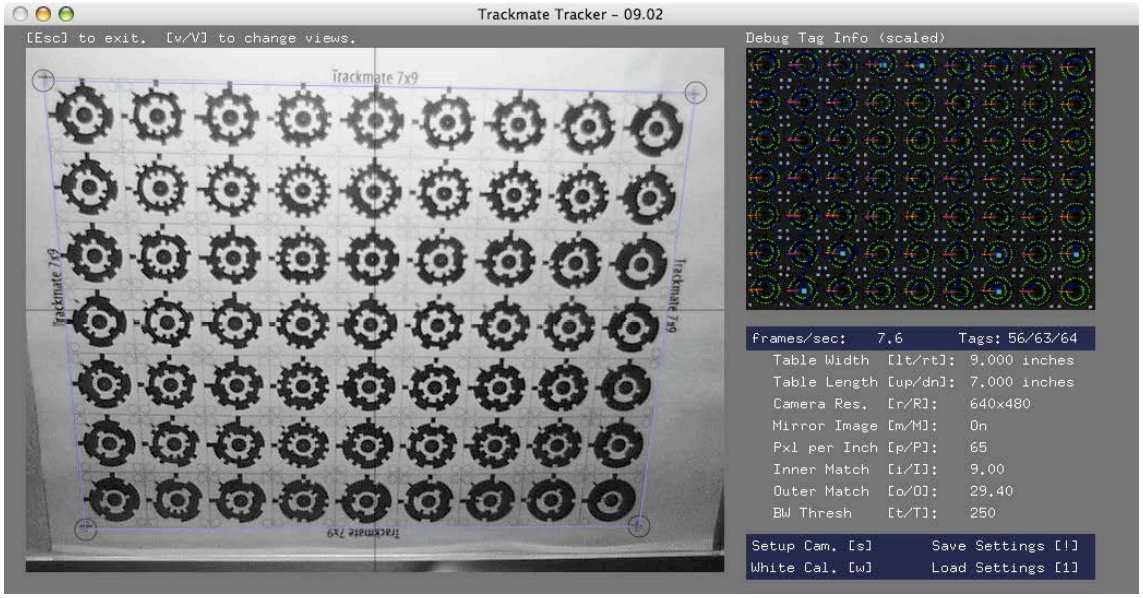


Ilustración 45. Tracker emitiendo la información.

- En este punto el tracker está configurando y emitiendo la información detectada de las etiquetas. Para hacer que la tasa de frames por segundo de la cámara aumente, se puede volver a pulsar ‘v’ y la aplicación entraría en modo rápido, *‘fast mode’*.

5 Validación de la Solución

5.1 Proceso de evaluación

La aplicación desarrollada se evaluará con un conjunto de pruebas que validarán todos los requisitos funcionales establecidos. Las principales características que la aplicación debe cumplir son las siguientes y deben estar presentes durante todo el proceso de validación:

- Reconocimiento e identificación de la naturaleza física de los objetos tales como el color, forma, material del que están hechos.
- Reconocimiento e identificación de las propiedades de los objetos relacionadas con su uso y utilidad tales como el riesgo, tipo de contenido, material etc...
- Poder utilizar más de un objeto físico al mismo tiempo.

Es igual de importante validar los tres objetivos anteriores como garantizar que todos los requisitos funcionales han sido implementados y para ello se han elaborado las pruebas que se muestran en el siguiente apartado.

5.1.1 Casos de prueba

Cada caso de prueba contará con una serie de campos que lo definen. Los campos que contiene cada caso de prueba son los siguientes:

- **Identificador**: valor único que identifica de manera unívoca al caso de prueba.
- **Objetivo**: cuál es la meta que se pretende con el caso de prueba..
- **Procedimiento**: pasos a seguir para llevar a cabo el caso de prueba y que esté se considere completado con éxito.
- **Resultado**: cuál debería ser el efecto en los diferentes componentes del sistema.

CP-01	
Objetivo	Reconocer figuras geométricas
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir una figura geométrica. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre las propiedades del objeto físico de manera correcta.(ver Ilustración 46)

Tabla 31. Prueba CP-01.

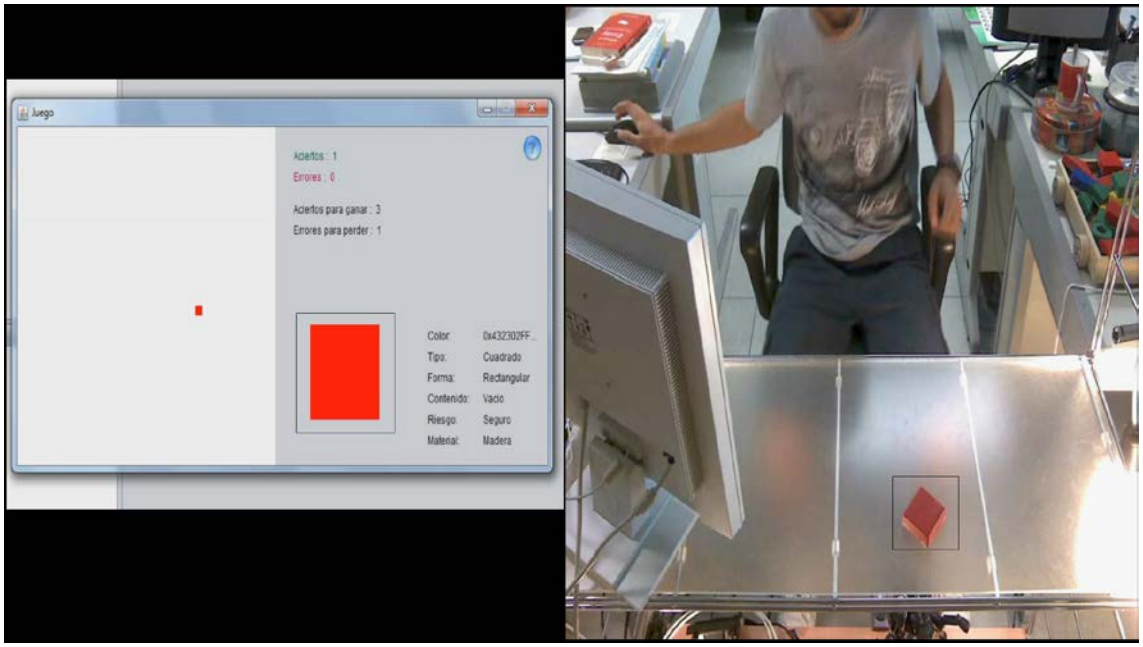


Ilustración 46.Prueba de reconocimiento de figuras geométricas

CP-02	
Objetivo	Reconocer operadores matemáticos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un operador matemático. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre las propiedades del objeto físico de manera correcta.

Tabla 32. Prueba CP-03.

CP-03	
Objetivo	Reconocer dígitos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un dígito. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre las propiedades del objeto físico de manera correcta.

Tabla 33. Prueba CP-03.

CP-04	
Objetivo	Reconocer objetos de cocina
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto de cocina. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre las propiedades del objeto físico de manera correcta.

Tabla 34. Prueba CP-04.

CP-05	
Objetivo	Reconocer el color de los objetos físicos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre la propiedad color del objeto físico de manera correcta. (ver Ilustración 47)

Tabla 35. Prueba CP-05.

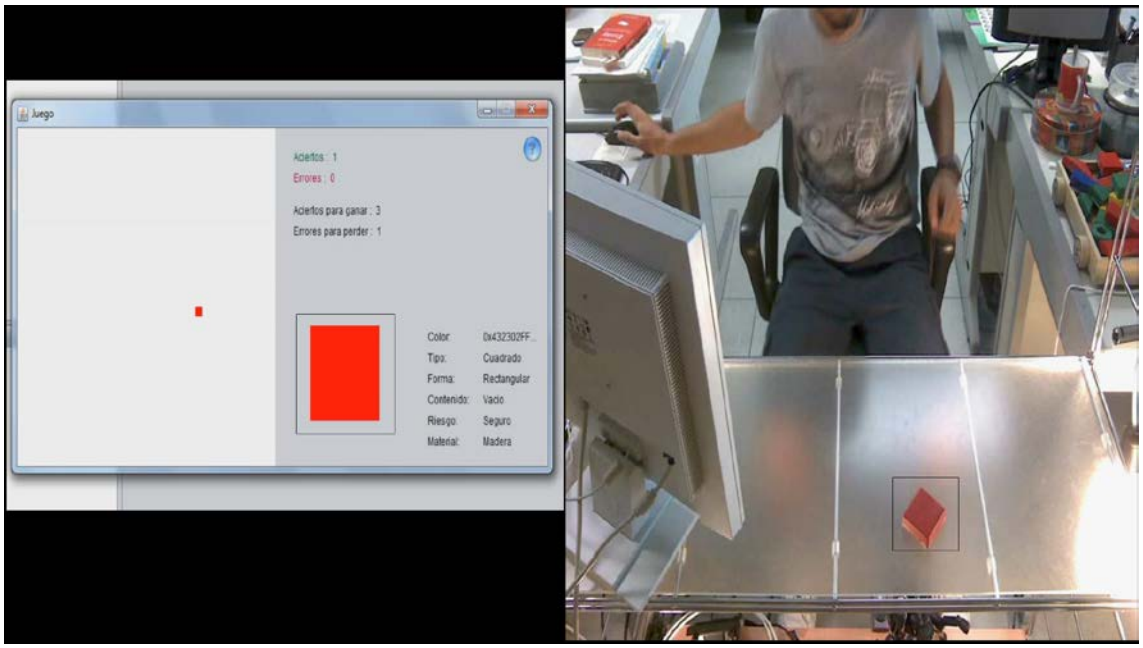


Ilustración 47.Prueba Reconocimiento de Color

CP-06	
Objetivo	Reconocer el contenido de los objetos físicos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre la propiedad contenido del objeto físico de manera correcta.(ver Ilustración 48)

Tabla 36. Prueba CP-06.

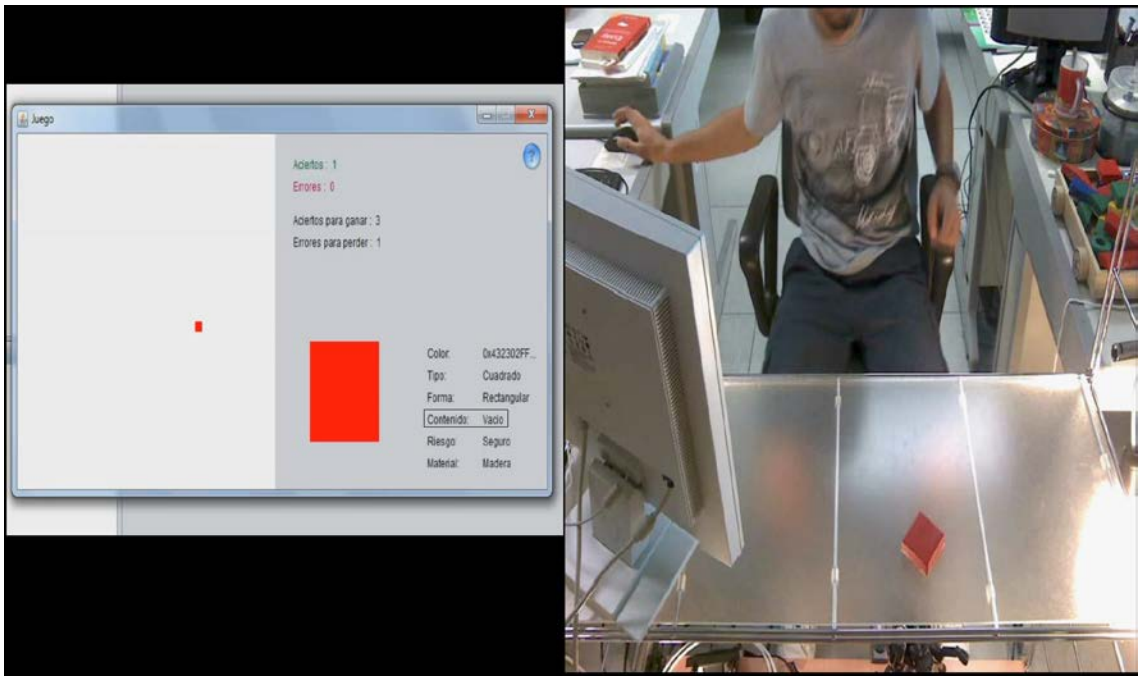


Ilustración 48.Prueba Reconocimiento del Contenido

CP-07	
Objetivo	Reconocer el material de los objetos físicos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre la propiedad material del objeto físico de manera correcta. (ver Ilustración 49)

Tabla 37. Prueba CP-07.

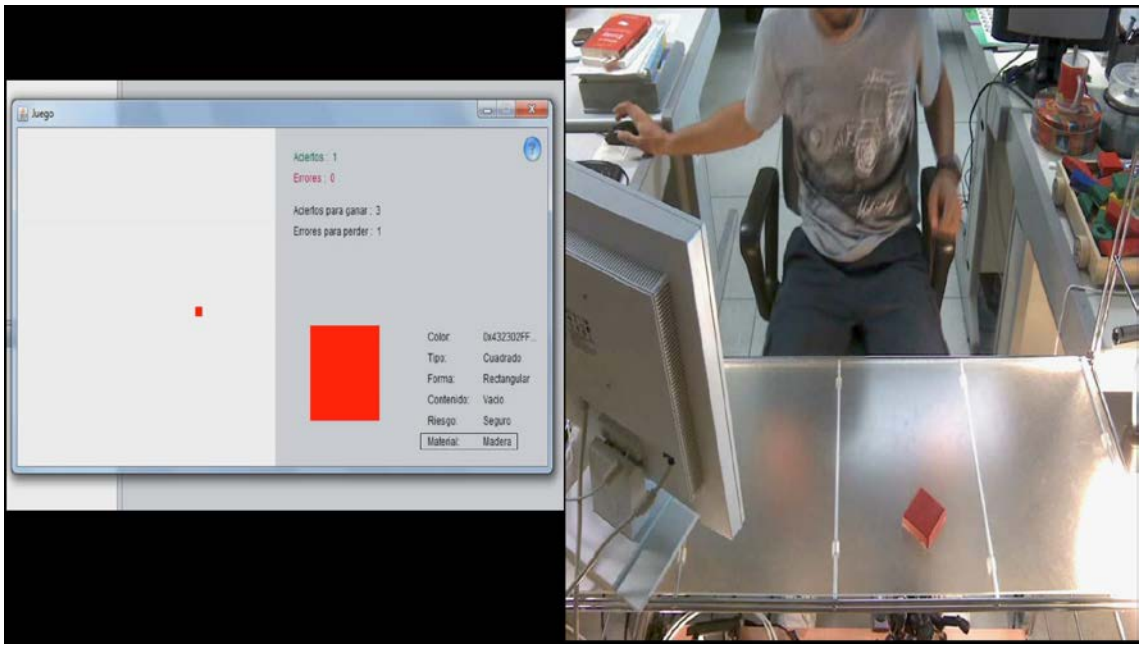


Ilustración 49. Prueba Reconocimiento del Material

CP-08	
Objetivo	Reconocer la forma de los objetos físicos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre la propiedad forma del objeto físico de manera correcta. (ver Ilustración 50)

Tabla 38. Prueba CP-08.

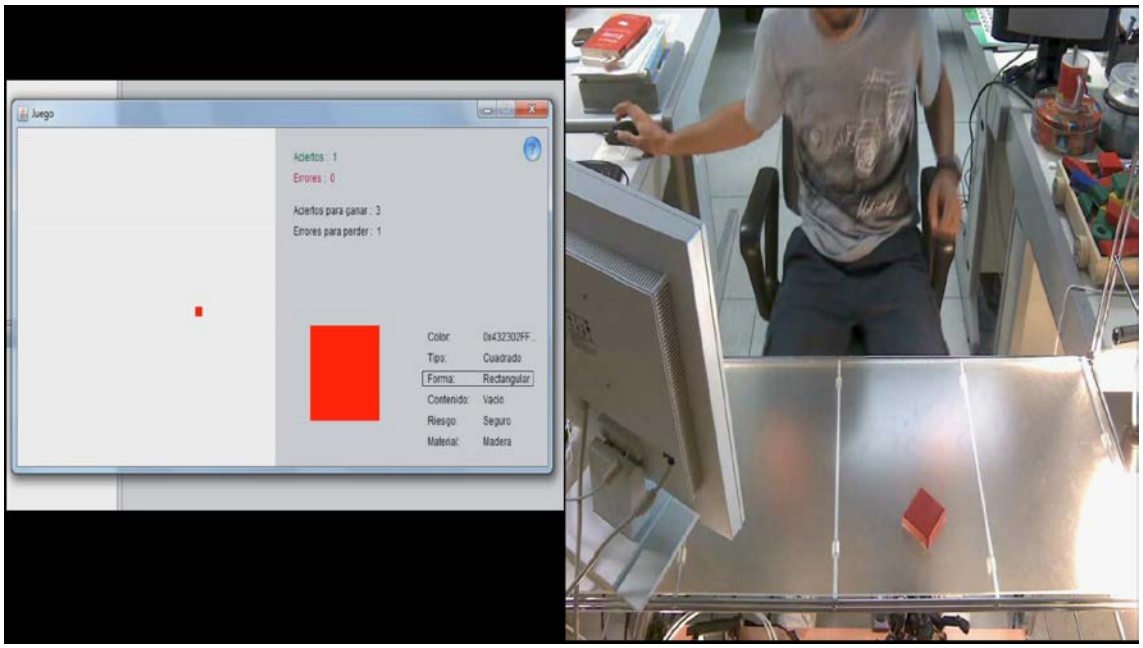


Ilustración 50.Prueba Reconocimiento de Forma

CP-09	
Objetivo	Reconocer el riesgo de los objetos físicos
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre la propiedad riesgo del objeto físico de manera correcta. (ver Ilustración 51)

Tabla 39. Prueba CP-09.

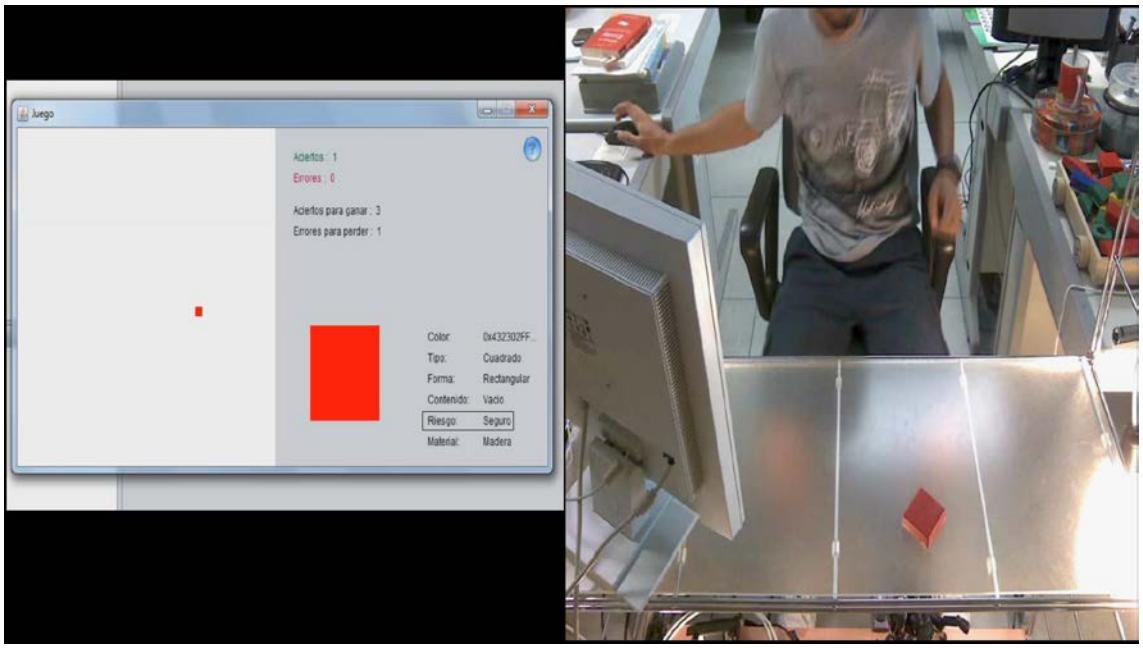


Ilustración 51.Prueba Reconocimiento del Riesgo

CP-10	
Objetivo	Dejar la detección de un objeto físico
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento. • Una vez detectado el objeto quitarlo de la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de la etiqueta en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre las propiedades del objeto físico de manera correcta. Una vez realizada dicha detección el objeto desaparecerá del Trackmate-Tracker y el sistema externo al que esté conectado el entorno recibirá un evento de eliminación del objeto de la lista de objetos detectados. (ver Ilustración 52)

Tabla 40. Prueba CP-10.

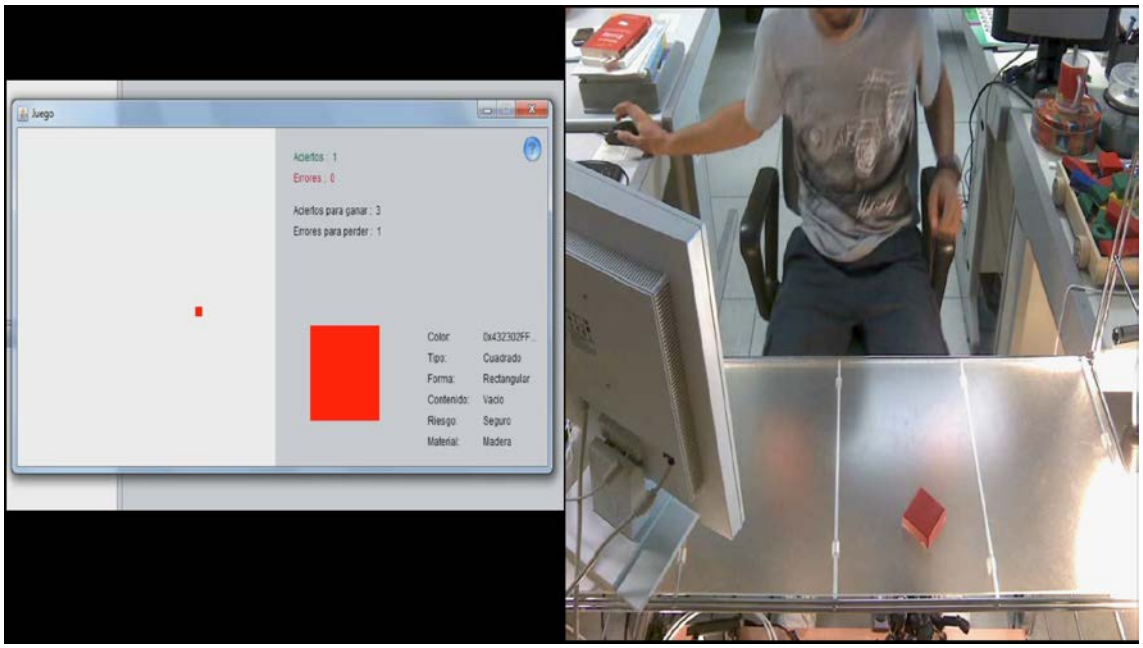


Ilustración 52.Prueba Reconocimiento de Objetos Físicos

CP-11	
Objetivo	Reconocimiento de varios objetos físicos simultáneamente
Procedimiento	• Calibrar el entorno para el reconocimiento de objetos. • Elegir un objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento. • Elegir un segundo objeto físico a reconocer. • Colocar el objeto sobre la superficie de reconocimiento.
Resultado	Como resultado se debe apreciar el reconocimiento de las etiquetas en el Trackmate-Tracker y el sistema externo con el que esté conectado el entorno debe recibir la información sobre las propiedades de los objetos físicos de manera correcta.

Tabla 41. Prueba CP-11.

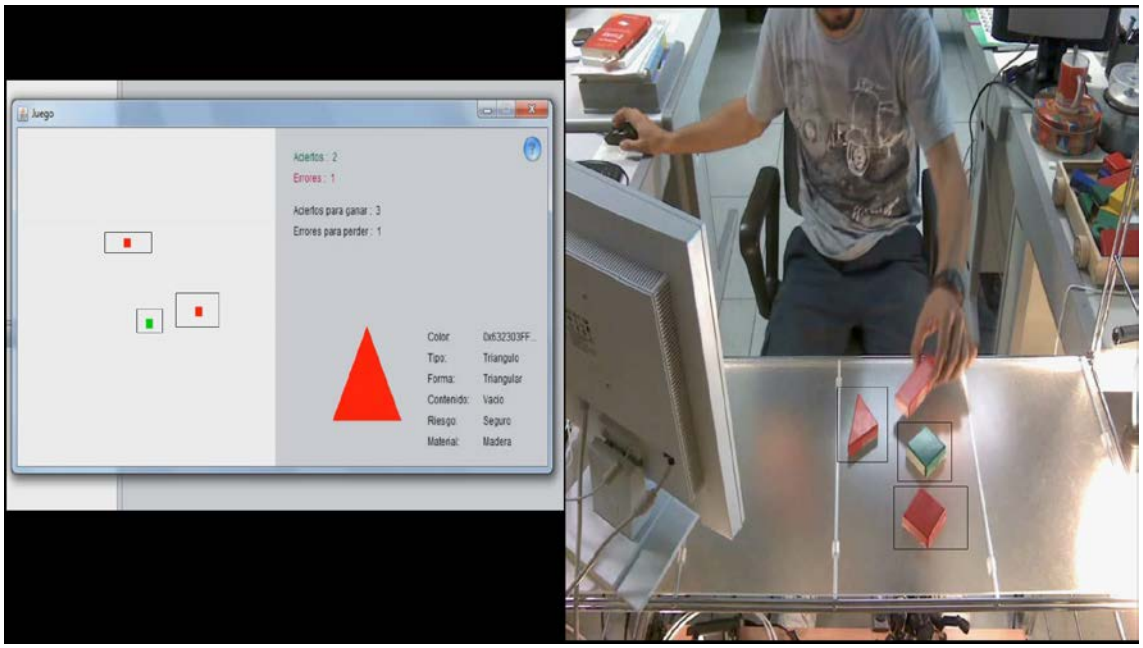


Ilustración 53. Prueba Múltiples Objetos.

5.2 Análisis de resultados

Para evaluar la aplicación construida se realizará una trazabilidad de los objetivos del sistema con las pruebas realizadas. También se realizará una trazabilidad de requisitos funcionales con pruebas de evaluación y viceversa.

Análisis de los objetivos	
La representación e identificación de la naturaleza física de los objetos tales como el color, forma, material del que están hechos.	CP-01, CP-02, CP-03, CP-04, CP-05, CP-07, CP-08
La identificación de las propiedades de los objetos relacionadas con su uso y utilidad tales como el riesgo, tipo de contenido etc...	CP-01, CP-02, CP-03, CP-04, CP-06, CP-09,
Poder utilizar más de un objeto físico al mismo tiempo permitiendo la resolución cooperativa del ECG y el aumento de las capacidades interactivas del mismo.	CP-11
La definición de un modelo de datos que permita representar todas las propiedades comentadas en los apartados anteriores.	CP-01, CP-02, CP-03, CP-04, CP-05, CP-06, CP-07, CP-08, CP-09

Tabla 42. Análisis de los objetivos.

Trazabilidad Pruebas de Evaluación – Requisitos Funcionales	
CP-01	RE-US-01, RE-US-07
CP-02	RE-US-03, RE-US-07
CP-03	RE-US-03, RE-US-07
CP-04	RE-US-02, RE-US-07
CP-05	RE-US-04, RE-US-07
CP-06	RE-US-04, RE-US-07
CP-07	RE-US-04, RE-US-07
CP-08	RE-US-04, RE-US-07
CP-09	RE-US-04, RE-US-07
CP-10	RE-US-05, RE-US-07
CP-11	RE-US-06, RE-US-07

Tabla 43. Matriz de trazabilidad Pruebas-Requisitos Funcionales.

Trazabilidad Requisitos Funcionales - Pruebas de Evaluación	
RE-US-01	CP-01
RE-US-02	CP-04
RE-US-03	CP-02, CP-03
RE-US-04	CP-05, CP-06, CP-07, CP-08, CP-09
RE-US-05	CP-10
RE-US-06	CP-11
RE-US-07	CP-01, CP-02, CP-03, CP-04, CP-05, CP-06, CP-07, CP-08, CP-09, CP-10, CP-11

Tabla 44. Matriz trazabilidad Requisitos Funcionales-Pruebas de Evaluación.

En la Tabla 42 se puede observar que los objetivos del sistema son cubiertos con todos los casos de pruebas

Para asegurar que el sistema construido implementa todas las funcionalidades que se definieron en el diseño del sistema, se tiene la Tabla 43 donde se puede ver que todas las pruebas tienen el propósito de demostrar al menos un requisito funcional. La Tabla 44 sirve para poder confirmar que todos los requisitos funcionales han sido implementados.

Se puede garantizar que todos los requisitos han sido implementados, y por consiguiente se ha conseguido un sistema que cumple con todo lo que se definió en la

6 Conclusiones y Trabajos Futuros

El Diseño Centrado en el Usuario es una tendencia de diseño de sistemas interactivos que pone el acento en el destinatario del sistema. Es importante tener en cuenta a qué tipo de personas va dirigida la tecnología desarrollada ya que dependiendo de sus características la tecnología puede ser realmente útil o un estorbo, lo cual, la llevaría al fracaso. Muchas veces a la hora de diseñar tecnología no se tiene en cuenta a los usuarios inexpertos, desde el punto de vista tecnológico, lo cual puede dificultar la incorporación de los mismos a la tecnología. De ahí la razón de este proyecto, proponer un sistema que ofrezca una interacción con la que reducir los inconvenientes que tienen dichos usuarios a la hora de aprender a utilizar la tecnología. Para intentar resolver esta problemática el proyecto trata de disminuir el periodo de aprendizaje del uso de la tecnología, al mínimo, ganando en efectividad y diversión, y disminuyendo una posible frustración del usuario al hacer uso de la tecnología.

El entorno ofrece una interacción intuitiva e innata respondiendo a acciones tales como soltar o coger un objeto en la superficie de reconocimiento. Para la identificación de las propiedades de los objetos físicos se ha utilizado una técnica de etiquetado y reconocimiento de dichas etiquetas. Una vez obtenida la información de la etiqueta esta se transmite a la aplicación por medio de una canal de comunicación.

6.1 Aportaciones

El entorno creado para la solución se le ha dotado de un estilo de interacción tangible de tal forma que las acciones reconocidas son soltar un objeto, recogerlo y deslizarlo por la superficie de reconocimiento. Estos movimientos son naturales e innatos a las personas, y no requieren de aprendizaje por lo que se consigue uno de los objetivos principales que es ofrecer una interfaz sencilla que permita salvar las barreras que puedan existir entre un usuario inexperto y la tecnología. Gracias a esta aportación el usuario se puede centrar sólo en qué es lo que tiene que aprender y no en el cómo lo tiene que aprender.

Otra de las funcionalidades que se ofrece con el entorno es la identificación de las propiedades de los objetos físicos que se proporcionan junto con el entorno. Algunas de esas propiedades son físicas y otras de las propiedades están relacionadas con su uso y utilidad por lo que también se cumplen los objetivos secundarios relacionados con la identificación de propiedades. Además también se pueden utilizar varios objetos simultáneamente dotando al entorno de aprendizaje cooperativo, es decir, permitir de manera cooperativa completar las tareas para la realización de los juegos.

Por último se ha diseñado un modelo de datos e información a través de la cual se puede enviar la información identificada de los objetos físicos a diferentes sistemas externos para que hagan uso de dicha información, integrando un protocolo de comunicación a través del cual se puede conectar el entorno a cualquier otro sistema que integre el protocolo de comunicación.

6.2 Trabajos Futuros

El proyecto sólo se enfoca en algunos aspectos de la problemática del HCI. Para ello se ha propuesto un prototipo y como tal tiene carencias y cosas que mejorar:

- La primera y más importante es trabajar en un entorno que sea menos sensible a los cambios ambientales de luminosidad. Para ello hay dos campos en los cuales se puede mejorar el entorno. El primero es la técnica de identificación de las propiedades de los objetos físicos. Se puede mejorar tanto el tipo de etiquetado como el software de reconocimiento para que sea menos sensible a cambios en el entorno. El otro campo en el cual se puede mejorar el entorno es en su robustez cerrando la sección del entorno en la cual se encuentra la iluminación y la cámara de reconocimiento.
- Otro aspecto a mejorar del prototipo es aumentar la portabilidad del entorno. Para ello se podría reducir el tamaño del mismo y aumentar la robustez para posibles traslados. La estructura ideal sería una caja con todos los elementos, cámara, luces y superficie de reconocimiento, fijos a dicha estructura, de tal forma que no sea sensibles a cambios en la luminosidad del entorno y a traslados.

6.3 Opiniones Personales

La realización de este proyecto me ha permitido introducirme en el mundo de la interacción entre las personas y la tecnología. Es importante tener en cuenta a qué tipo de personas va dirigida la tecnología, ya que dependiendo de sus características la tecnología puede ser útil o un estorbo adicional. Utilizando el estilo de interacción tangible con objetos físicos y limitando las acciones a un conjunto cercano a los conocimientos del usuario y sin necesidad de tener que aprender nuevas acciones para realizar las tareas con el entorno. Pero este tipo de entornos tiene limitaciones. La utilización del reconocimiento de etiquetas a través de una cámara implica la necesidad de un entorno estable, en condiciones de luz y disposición. Para ello, previamente, debe estar debidamente calibrado, haciéndolo esto sensible a cualquier desplazamiento o cambio en la intensidad de luminosidad en el entorno lo cual, a tenor de lo experimentado en las pruebas de concepto realizadas para el proyecto, hace que la efectividad de uso disminuya y en ocasiones pueda aumentar la frustración al usarlo. Otro problema, bastante relacionado con el anterior, detectado en las pruebas de concepto realizadas al entorno y que puede producir la misma sensación en el usuario, es que la tasa de reconocimiento de los objetos no es del 100% pudiendo dificultar la interacción del usuario con el entorno.

Anexo A. Control de versiones

Tabla de estado del documento			
Nombre	Desarrollo de una interfaz tangible para el aprendizaje basado en juegos de forma natural e innata		
Fecha	26/04/2010		
Código	TUIECG_PFC		
Versión	1.10		
Estado	Pendiente de aprobación		
Revisado por	David Díez Cebollero	**Fecha**	03/10/2011
Aprobado por	No consta	**Fecha**	--/--/----

Tabla de modificaciones relevantes		
Versión	**Fecha**	**Avance y modificaciones relevantes**
1.0	26/04/2011	Primera versión. ▪ Diseño del Documento ▪ Introducción ▪ Contexto del problema
1.1	3/05/2011	▪ Diseño del Documento ▪ Introducción ▪ Planteamiento del Problema ▪ Contexto del problema
1.2	10/05/2011	▪ Introducción ▪ Planteamiento del Problema ▪ Contexto del problema
1.3	17/05/2011	▪ Introducción ▪ Planteamiento del Problema ▪ Contexto del problema
1.4	08/06/2011	▪ Introducción ▪ Contexto del problema
1.5	01/07/2011	▪ Estado de la Cuestión
1.6	2/09/2011	▪ Requisitos ▪ Gestión de Proyecto ▪ Diseño ▪ Validación ▪ Conclusiones
1.7	16/09/2011	▪ Resumen. ▪ Reestructuración del estado de la cuestión. ▪ Mejora de la gestión del proyecto. ▪ Casos de Prueba. ▪ Anexos.

1.8	23/09/2011	▪ Introducción ▪ Estado de la Cuenstión ▪ Implementación
1.9	1/10/2011	▪ Introducción ▪ Estado de la Cuenstión ▪ Resumen. ▪ Conclusiones.
1.10	3/10/2011	▪ Resumen ▪ Conclusiones ▪ Abstract
1.11	27/10/2011	▪ Presupuesto

Anexo B. Propiedades Identificadas

Figura	Código
INDEFINIDO	0
CIRCULO	1
CUADRADO	2
TRIANGULO	3
CILINDRO	4
PIRAMIDE	5
CUBO	6
ESFERA	7

Tabla 45.Figuras Reconocidas

Material	Código
INDEFINIDO	0
PLASTICO	1
METAL	2
MADERA	3

Tabla 46. Material Reconocido

Objeto	Código	Objeto	Código
Básicos		**Dígitos**	
INDEFINIDO	0	DIGITOCERO	15
RECTANGULO	1	DIGITOUNO	16
CIRCULO	2	DIGITODOS	17
CUADRADO	3	DIGITOTRES	18
TRIANGULO	4	DIGITOCUATRO	19
CILINDRO	5	DIGITOCINCO	20
PIRAMIDE	6	DIGITOSEIS	21
CUBO	7	DIGITOSIETE	22
Cocina		DIGITOOCHO	23
BOTELLA	8	DIGITONUEVE	24
CUCHILLO	9	**Operadores Matemáticos**	
CUCHARA	10	SUMA	25
TENEDOR	11	RESTA	26
SALERO	12	MULTIPLICACION	27
FUEGO	13	DIVISION	28
SERVILLETA	14		

Tabla 47. Objetos Reconocidos

Contenido	Código
LIQUIDOINDEFINIDO	0
GASINDEFINIDO	1
SOLIDOINDEFINIDO	2
VACIO	3
AGUA	4
LEJIA	5
ACIDO	6
SAL	7
ACEITE	8

Tabla 48. Contenidos Reconocidos

Riesgo	Código
INDEFINIDO	0
SEGURO	1
ARRIESGADO	2
PELIGROSO	3

Tabla 49. Riesgos Reconocidos

7 Bibliografía

[1] Carson Learning Service. Carson Learning Service. [Online]. http://www.clsllc.com/id/game/index.html

[2] Real Academia Española, *Diccionario de la lengua española*., 2001.

[3] CAN EDUCATIONAL COMPUTER GAMES HELP EUCATORS LEARN ABOUT THE PSYCHOLOGY OF LEARNING MATHEMATICS IN CHILDREN?, "Kamran Sedighian and Andishe Sedighia".

[4] Steven Heim, "The Resonant Interface, HCI FOUNDATIONS FOR INTERACTION DESIGN," 2008.

[5] David Benyon, *Designing Interactive Systems*.: Adison Wesley, 2005.

[6] Minder Chen, Titus D. M. Purdin Jay Nunamaker Jr., "System Development in Information System Research".

[7] Ministerio de política territorial y administración pública. Descripción de Métrica V3 (2010). [Online]. http://administracionelectronica.gob.es

[8] Donald A. Norman, "The Psychology of Everyday Things," 1988.

[9] M.I.T. MIT Media Lab - Tangible Media Group. [Online]. http://tangible.media.mit.edu/project.php?recid=115

[10] MIT. Lusidosc. [Online]. http://lusidosc.sourceforge.net/

[11] MIT. Wiki TrackMate. [Online]. http://sourceforge.net/apps/mediawiki/trackmate/index.php?title=Trackmate_Tracker

[12] MIT. Wiki Track Mate Etiquetas. [Online]. http://sourceforge.net/apps/mediawiki/trackmate/index.php?title=Trackmate_Tagger

[13] Microsoft. Welcome to Surface 2.0. [Online]. http://www.microsoft.com/surface/

[14] Red Eléctrica Española. [Online]. http://www.ree.es/educacion/controla.asp

[15] Carleton College. Teaching Entry Level Geoscience. [Online]. http://serc.carleton.edu/introgeo/games/

[16] Jimmy Wales y Larry Sanger. Wikipedia. [Online]. http://es.wikipedia.org/wiki/Wikipedia:Portada

[17] Education and Culture DG Lifelong learning programer. Engage Learning. [Online]. http://www.engagelearning.eu/

[18] Paula Selvidge, "How Long is Too Long to Wait for a Website to Load?," 1999.

[19] Linda Lim and Andrew Turk, "Individual Differences and Human Computer Interaction".

[20] Alan Dix, Janet Finlay, Gregory Abowd, and Beale Russell, *Human Computer Interaction*.: PRENTICE HALL, 1998.

[21] Shalom M. Fisch, "Making Educational Computer Games "Educational"".

[22] Adam Kumpf. Trackmate: Large-Scale Accessibility of Tangible User Interfaces.