Escuela Técnica Superior de Ingeniería de Telecomunicación

Ingeniería de Telecomunicación

PROYECTO FIN DE CARRERA

Diseño e Implementación de Librerías para la Adaptación del Wiimote como Pizarra Digital Interactiva

Autor: Ángel Torrado Carvajal

Tutor: Israel Herraiz Tabernero

Co-tutor: Gregorio Robles Martínez

Curso Académico 2009/2010

Proyecto Fin de Carrera

Diseño e Implementación de Librerías para la Adaptación del Wiimote como Pizarra Digital Interactiva

Autor
Ángel Torrado Carvajal

Tutor
Israel Herraiz Tabernero

Co-tutor
Gregorio Robles Martínez

La defensa del presente Proyecto Fin de Carrera se realizó el día de de , siendo calificada pro el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de .

Agradecimientos

Llego al fin de una etapa de mi vida. Una etapa marcada por numerosos momentos de alegrías y decepciones que han ido forjándome hasta llegar a este punto y, como tal, me gustaría hacer mención de aquellos que han compartido conmigo estos años en la Escuela.

En primer lugar me gustaría mencionar a mis padres. Sé que las palabras no son suficientes, ya que he de agradecerles muchas cosas por soportar tantas horas de estudio que han tenido sus más y sus menos (con sus éxitos y sus derrotas) y, más aun, porque son quienes me han ofrecido la oportunidad de convertirme en lo que hoy soy.

También quiero dar las gracias a mi hermano Antonio, por aceptar que me haya alistado con la "competencia" y, sobre todo, por confiar en que tomaría el relevo que en su día me dejó preparado y plasmado en su TFC.

Fuera de la Escuela, me gustaría agradecer su apoyo a Sergio, Dani, Ester, Manu, Emilio, Javi y Norber, porque llegaron a mi vida durante estos años y han conseguido hacer, aportando algo de una manera u otra, que todo sea más llevadero. Además, puedo decir a ciencia cierta, que han llegado para quedarse.

No puedo olvidar a los compañeros de clase, en especial Hugo, Fer, Jorge y Luismi, que han estado conmigo hasta el final. Hemos pasado mucho tiempo juntos, y espero que no pasen a formar parte del recuerdo universitario, quedando muchos momentos por vivir.

Tampoco podría olvidar a mis otros compañeros. No os nombraré a todos, pero sabréis quienes sois si afirmo que supisteis acogerme a pesar de la barrera que los cursos ponían.

Agradecer también a mis compañeros del IEEE, por ayudarme a cumplir un sueño que parecía inalcanzable hacía solo unos años.

Evidentemente, no puedo olvidar a mis tutores Isra y Grex, quienes apostaron por mí para la realización de este proyecto y han tratado de ayudarme en todo lo posible a lo largo del mismo.

Ahora, se abre un nuevo camino ante mi, pero sé que sin todas estas personas mi día de hoy no sería posible...

A todos vosotros,

GRACIAS

Resumen

En este proyecto se desea investigar y documentar la forma de funcionamiento del Wiimote, así como la implementación de una biblioteca que habilite su uso como ratón inalámbrico para manejar presentaciones y como puntero láser virtual.

En primer lugar, tendremos que familiarizarnos con la manera de funcionar del control remoto de Nintendo. Para ello, tendremos que realizar un estudio de ingeniería inversa que nos proporcione los conocimientos necesarios que seguidamente nos permitan abordar el problema y desarrollar nuestra propia aplicación.

De esta primera etapa, puramente de investigación, surgirá la documentación necesaria para comprender aquello de lo que disponemos y afianzar el camino a seguir.

Para desarrollar nuestra aplicación, nos serviremos de una serie de herramientas como son la tecnología Bluetooth, el lenguaje de programación Python o las librerías Xautomation.

Python nos ofrecerá una API capaz de permitirnos la comunicación Bluetooth con el Wiimote, mientras que Xautomatión proporcionará la capacidad de ofrecernos interacción con los eventos del ordenador.

Finalmente, se comparará nuestro producto final con las opciones de Pizarra Digital Interactiva (PDI) actuales en el mercado (opciones que se están implantando en algunos centros educativos españoles) proporcionando una alternativa de bajo coste.

A veces el recuerdo que deja un libro
es más importante que el libro en sí.

Índice general

Índice de figuras

Índice de cuadros

CAPÍTULO 1

Introducción

Los seres humanos nos tenemos que desenvolver en situaciones donde se requiere comunicación y para ello es necesario establecer medios en que la información se pueda procesar y que permitan que esto se pueda realizar.

Ante dicha creciente demanda de tecnologías de la información, donde las variables son equipos cada vez más rápidos, pequeños, eficientes y poderosos, la integración electrónica hace esto posible. Para lograrlo se reduce el tamaño de los circuitos, se introducen más elementos y la distancia entre los transistores es más pequeña, con lo que se logra aumentar la velocidad. Solo hay un gran problema, la integración se acerca al tamaño del átomo.

Cuando Gordon Moore en 1965 afirma que el número de transistores que contienen los chips se duplica cada año, establece la conocida Ley de Moore. Poco más tarde sufre una modificación en su relación con el tiempo: el número de transistores en un chip se duplica cada 18 meses. Esto quiere decir que cada 18 meses, por el mismo dinero, se puede comprar un microprocesador con el doble de potencia.

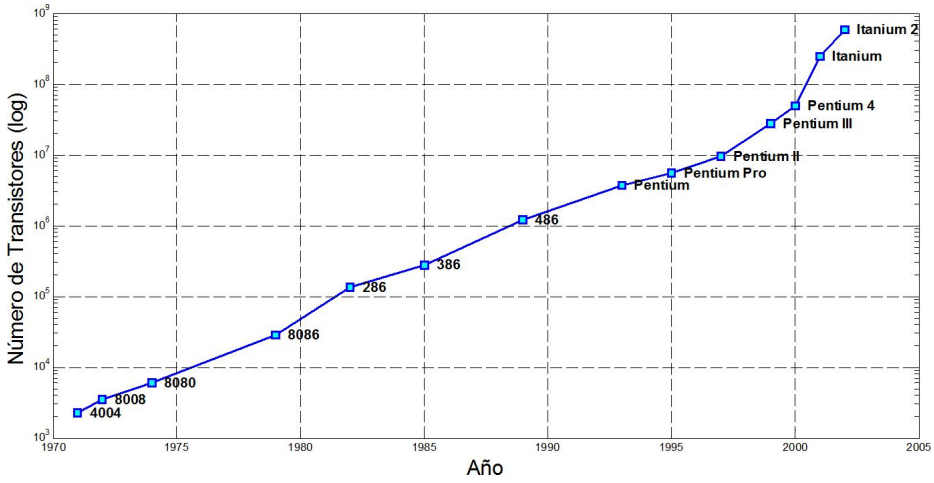


Figura 1.1: Ley de Moore

Esta ley resulta ser muy exacta desde entonces, pero lamentablemente va perdiendo vigencia. Intel, líder mundial en fabricación de procesadores, ya ha confirmado que probablemente la tendencia exponencial de la Ley de Gordon Moore no podrá ser mantenida en esta década.

Además, son varias las voces que advierten que la ley no puede seguir a ese ritmo o, por lo menos, no con esa forma de fabricación de circuitos integrados. De seguir la tendencia en la reducción del tamaño de los componentes resulta inevitable la confrontación con las leyes de la mecánica cuántica, surgiendo así la necesidad de crear nuevos algoritmos y nuevas clases de hardware.

Los expertos calculan que para dentro de 15 ó 20 años se habrá alcanzado este límite, y habrá que usar otros sistemas para la fabricación de los circuitos integrados y por lo tanto de las computadoras.

Por otra parte, el avance de las comunicaciones por un medio no guiado mediante ondas electromagnéticas (especialmente las comunicaciones por radiofrecuencia frente al típico modelo de comunicaciones cableadas) es imparable desde la liberación de la banda de 2,4 GHz.

Anteriormente, los cables que conectaban a los dispositivos o equipos eran difíciles de colocar, resultaban antiestéticos o limitaban la movilidad de los mismos. Actualmente, con las comunicaciones inalámbricas, todo esto se ha vuelto mucho más sencillo y barato.

El uso de las comunicaciones inalámbricas ha crecido y se ha extendido, por tanto, por todas partes del mundo de forma notoria. Todo esto debido a que son eficaces ya que permiten:

- Movilidad de los dispositivos,
- Facilidad de instalación,
- Flexibilidad,
- Reducción de costos,
- Escalabilidad...

Las comunicaciones inalámbricas han ido ganando adeptos rápidamente, debido a que se ha convertido en una tecnología madura y fiable; en gran parte debido a que permite resolver los inconvenientes derivados de la propia naturaleza del cable, como medio físico del enlace en las comunicaciones.

Haciendo análisis de lo comentado, esto nos sitúa en un presente donde capacidad de cómputo, tecnología y comunicaciones, se hallan en un nivel de desarrollo tal que hacen posible gran cantidad de aplicaciones que parecían impensables hace tan solo una década.

En este contexto, la compañía de videojuegos japonesa Nintendo [Nin] lanza en Octubre de 2006 la videoconsola Wii, que ha revolucionado el mercado de los videojuegos. Podría tratarse de una videoconsola más, pero sin embargo posee algo que la hace única y diferente: su mando de control, el Wii Remote o Wiimote.

Ésta es la gran apuesta de Nintendo en el mercado de las videoconsolas de sobremesa, con la que pretendía conseguir enganchar a todo el mundo a los videojuegos, no sólo a los más "jugones". Su intención era que la gente jugase en familia y que los padres y hermanos pequeños disfrutaran también con la consola. La fórmula para atraer a todos fue el mando de control que posibilita jugar juegos muy diversos y de una forma mucho más dinámica.

Se trata de una videoconsola muy revolucionaria que rompe con el concepto del videojuego clásico, sentado y con un único mando que se coge con las dos manos; la Wii introduce el movimiento del jugador como un elemento clave en el desarrollo del juego.

A pesar de no contar unas especificaciones técnicas muy complejas consigue aportar una experiencia de juego muy gratificante y novedosa.

Figura 1.2: Nintendo Wii

La Wii abandona el tradicional mando que se coge únicamente con las dos manos, y aparece el Wiimote. Este mando se puede agarrar con una sola mano y está dotado de unos sensores que detectan el movimiento. Capta también la profundidad y la posición o dirección al moverlo en el aire y está conectado al equipo mediante Bluetooth. Además, es ligero y manejable. Nintendo diseñó el mando de Wii para que fuera el dispositivo de juego más versátil de la historia.

Y es que, si examinamos el Wiimote a fondo, observamos que nos encontramos ante un dispositivo electrónico conformado por la unión de varios sensores y un buen sistema de comunicaciones inalámbrico. Todo esto nos ofrece la libertad asociada al mismo y que además, se encuentra al alcance de todos, convirtiéndolo en un aparato muy interesante desde el punto de vista de desarrolladores.

De esta manera, podemos observar como en los dos años que lleva en el mercado han surgido gran cantidad de proyectos paralelos de personas que supieron apreciar estas cualidades y han mostrado al mundo aplicaciones de lo más variadas, aunque centrándose mayoritariamente en el área de robótica.

En definitiva, el Wiimote es un mando bastante barato, teniendo en cuenta las posibilidades que ofrece, y con un gran potencial. De ahí que haya mucha gente fascinada desarrollando alrededor del mismo.

Sin embargo, independientemente de las posibilidades que puede ofrecer este dispositivo en otros campos más novedosos, nosotros nos centraremos en la aplicación más primitiva, es decir, la funcionalidad básica para la cual fue diseñado el Wiimote. De este modo intentaremos obtener resultados similares a los ofrecidos en la navegación entre canales en el menú de Wii.

CAPÍTULO 2

Objetivos

En este capítulo se explica la línea de trabajo a seguir durante el desarrollo del proyecto. Se describe el problema que se quiere abordar, los requisitos que deben cumplir las soluciones que se adopten y qué métodos de trabajo se han seguido.

El Wiimote incorpora sensores que detectan el movimiento, pudiendo captar también la profundidad y la posición o dirección del mismo al moverlo en el aire. Esto, unido a su conexión Bluetooth, hacen de él un interesante y potente accesorio de bajo coste.

Presentadas las características del Wiimote y conociendo a grandes rasgos sus capacidades, consideramos que es un dispositivo con cualidades interesantes para poder ampliarlas a otros campos, y ese será el objetivo del presente proyecto.

2.1 Datos fundamentales del proyecto

Por los motivos descritos en el párrafo anterior, se ha decidido que Linux cuente con un módulo capaz de realizar funciones básicas pero similares a las que realiza la Wii de Nintendo con los datos que le envía el Wiimote.

En esta línea, los objetivos a cumplir con este proyecto, son principalmente dos:

1. Usar el Wiimote como ratón inalámbrico Bluetooth para manejar presentaciones.

   En esta primera parte se llevará a cabo la implementación de un programa que se encargue de las comunicaciones pertinentes con el mando inalámbrico Bluetooth.

   En primer lugar, se creará una librería que sea capaz de leer los paquetes que envía el Wiimote con la información de los botones y los sensores del mismo.

   Por otra parte, la información de los botones y los sensores acelerómetros, se traducirán en el ordenador a eventos de ratón o teclado empleando "Xte".

2. Usar el Wiimote como puntero láser virtual.

   En esta segunda parte se llevará a cabo la obtención de los datos proporcionados por la cámara de infrarrojos que posee el mando.

   Para ello, realizaremos la calibración del mismo teniendo en cuenta la resolución de pantalla del ordenador en el cual se está empleando, triangulando y/o interpolando la posición del cursor llegando a emplear esta información para posicionar y mover el cursor del ratón por la pantalla.

Resumiendo, los objetivos principales del proyecto que nos ocupa son la creación de una serie de librerías que nos permitan realizar comunicaciones Bluetooth con el Wiimote, traduciéndolos a eventos que nos interesen en el ordenador, como pueden ser el posicionamiento del cursor del ratón o eventos de pulsación de botones o teclas.

2.2 Estimación inicial a nivel producto - Requisitos

Ahora que se han definido los objetivos que se pretenden alcanzar con la realización de este proyecto, hay que describir exactamente qué requisitos mínimos tienen que cumplir las aplicaciones realizadas.

En primer lugar hemos de darnos cuenta de que nos encontramos ante un sistema de tiempo real, puesto que las comunicaciones y el procesamiento de los datos se deberán llevar a cabo en el momento y las decisiones y acciones deberán ser tomadas en un período de tiempo que no sea contraproducente. Una vez conocido dicho detalle, podremos definir como requisitos los siguientes:

- El componente que se encargue de realizar la identificación de los datos correspondientes a los diferentes botones y sensores deberá trabajar continuamente teniendo en cuenta que no existe un patrón fijo de funcionamiento o periódico y procesando los datos según le vayan siendo suministrados.

- El componente que se encargue de realizar la triangulación y posicionamiento del cursor en la pantalla, deberá funcionar de tal manera que se proporcionen los resultados a los datos proporcionados a la vez que los está recibiendo.

- Los datos proporcionados por los módulos han de ser fiables, es decir, que finalmente nuestro programa presente datos con un índice de error muy bajo.

- Todos los módulos implementados deben consumir la menor capacidad de cómputo posible para que el rendimiento de la máquina no se vea penalizado por nuestro software.

2.3 Estimación inicial a nivel proyecto - Metodología

Para el desarrollo del proyecto, se ha seguido un plan de trabajo basado en el modelo de desarrollo en cascada. Se ha elegido este sistema porque se tiene todo bien organizado y no se mezclan las fases, siendo un modelo de desarrollo perfecto para proyectos en los que las especificaciones finales están cerradas, y además donde se especifiquen muy bien los requerimientos y se conozcan las herramientas a utilizar a lo largo del desarrollo del mismo.

En Ingeniería de software el desarrollo en cascada, también llamado modelo en cascada, es el enfoque metodológico que ordena rigurosamente las etapas del ciclo de vida del software, de forma tal que el inicio de cada etapa debe esperar a la finalización de la inmediatamente anterior.

Un ejemplo de una metodología de desarrollo en cascada es:

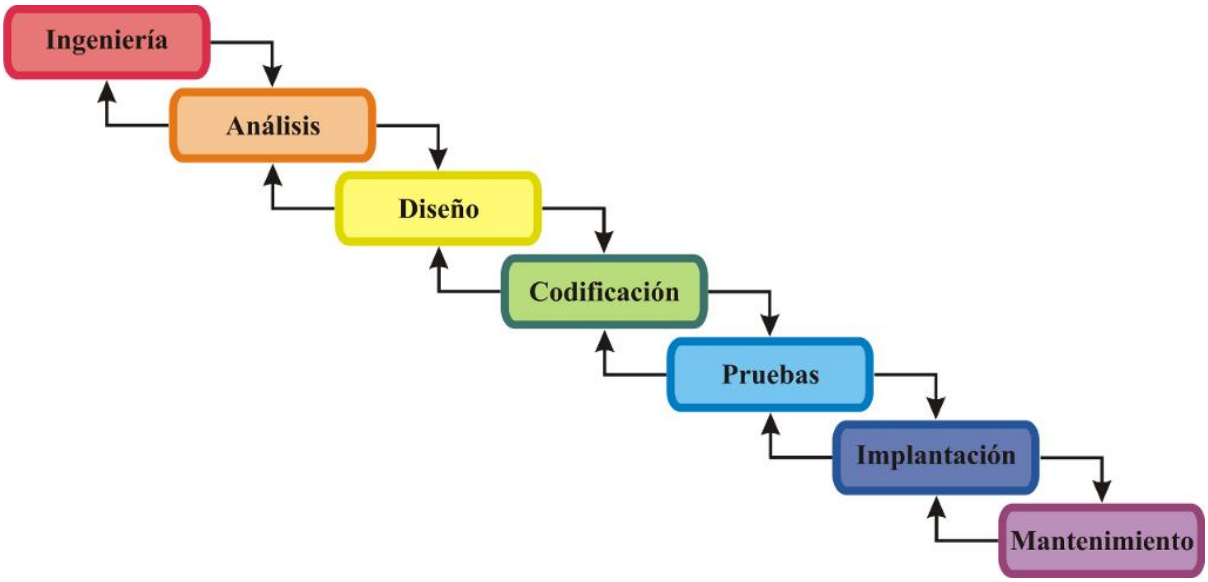


Figura 2.1: Desarrollo en Cascada

De esta forma, cualquier error de diseño detectado en la etapa de prueba conduce necesariamente al rediseño y nueva programación del código afectado, aumentando los costes del desarrollo.

La palabra cascada sugiere por tanto, mediante la metáfora de la fuerza de la gravedad, el gran esfuerzo que es necesario para introducir un cambio en las fases más avanzadas de un proyecto.

Si bien ha sido ampliamente criticado desde el ámbito académico y la industria, que defienden de manera significativa el desarrollo en espiral, sigue siendo un paradigma bastante seguido a día de hoy.

Veamos detalladamente cuales son las líneas generales de actuación en cada una de las fases del modelo:

- Ingeniería y Análisis del Problema. A través de las especificaciones del problema planteado, se hace un estudio a grandes rasgos sobre las posibles soluciones y su análisis de viabilidad.

- Análisis de requisitos. Se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (Documento de Especificación de Requisitos), que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos.

  Es importante señalar que en esta etapa se deben consensuar todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software.

- Diseño. En esta etapa se realizan los planes y concepciones originales que finalmente darán como resultado la producción del producto. Para ello tendremos que tener en cuenta el diseño desde dos perspectivas diferentes.

  Diseño de Alto Nivel. Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el SDD (Documento de Diseño del Software), que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras.

  Diseño de Bajo Nivel o Detallado. Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber que herramientas usar en la etapa de Codificación.

- Codificación. Es la fase de programación o implementación propiamente dicha. Aquí se implementa el código fuente.

  Dependiendo del lenguaje de programación y su versión se crean las librerías y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

- Pruebas. Se realizan pruebas unitarias, donde se evalúa cada módulo como caja negra observando su comportamiento, y de integración, donde los módulos se ensamblan para componer el sistema y se comprueba que funciona correctamente antes de ser puesto en explotación.

- Implantación o Instalación. El software obtenido se pone en producción. Se implantan los niveles software y hardware que componen el proyecto. La implantación es la fase con más duración y con más cambios en el ciclo de elaboración de un proyecto. Es una de las fases finales del proyecto.

  Durante la explotación del sistema software pueden surgir cambios, bien para corregir errores o bien para introducir mejoras. Todo ello se recoge en los Documentos de Cambios.

- Mantenimiento. En caso de ser un producto comercial o de un software utilizable, se procederá a llevar a cabo las operaciones pertinentes para el cuidado necesario que garantice el correcto funcionamiento del mismo.

2.3.1 Planificación

Como un proyecto es un proceso complejo, necesitamos llevar a cabo la estipulación de los hitos necesarios con el fin de aumentar las probabilidades de alcanzar la correcta finalización del mismo. Dichos hitos vendrán dados fundamentalmente por los distintos bloques que se consideran en el modelo de desarrollo en cascada. A continuación pasaremos a explicar la actividad que se piensa realizar en cada una de las diferentes etapas consideradas:

- Ingeniería y Análisis del Problema. Dada la idea principal del proyecto que nos permite conocer el problema y remitiéndonos a líneas realizadas por otros desarrolladores, estimamos la posibilidad de darle solución al problema planteado y la factibilidad de la misma. Dicho estudio, a priori, no debería prolongarse mucho tiempo.

- Análisis de requisitos. Analizamos más profundamente los requisitos finales del proyecto consensuando con los tutores para así poder plantear de manera más clara los objetivos del mismo y que han sido plasmados en el punto 2.1, pero aun sin entrar en profundidad en la manera en que se abordarán los problemas planteados para darles solución. Dichas especificaciones son consecuencia directa de la idea del proyecto inicial, donde nuestros requisitos finales son claramente cerrados, puesto que cada bloque plantea una funcionalidad determinada.

  En este sentido, deberemos recabar información sobre el funcionamiento del dispositivo que vamos a emplear, el Wiimote, que al ser un producto comercial, del cual no se suministran hojas de características o especificaciones de funcionamiento del mismo, nos dirigirá a un proceso de ingeniería inversa para conocer su funcionamiento.

Al mismo tiempo, deberemos emplear la tecnología Bluetooth conociendo su funcionamiento y que nos permitirá las comunicaciones con el mando, mientras nos formamos en Python que será nuestro lenguaje de programación, además de aprender el funcionamiento de Xte y técnicas de triangulación.

La fase de análisis es una de las más largas, ya que requiere bastante tiempo para recopilar información.

- Diseño. Para esta fase, donde se plantearán las soluciones al problema, deberemos tener en primer lugar conocimiento de las tecnologías y productos con los que estamos trabajando (obtenidos en la fase de análisis) y una vez conocidos estos condicionantes pasaremos a la fase de diseño propiamente dicho.

  Una vez finalizado el diseño del sistema pasaremos al diseño del programa, donde deberemos desarrollar o elegir los algoritmos que se adecuen a nuestras especificaciones y sean más eficientes para nuestro fin.

  Hemos de proponer diseños válidos y consistentes, procurando que sean lo suficientemente buenos para el desarrollo de la siguiente fase sin tener que volver atrás a repetir la etapa de diseño.

- Codificación. En esta fase es donde implementaremos el software y las librerías necesarias para hacer funcionar el sistema descrito por los diseños del apartado anterior de la manera que se especificó en el análisis de requisitos.

  Para alcanzar este fin se dedicará gran cantidad de tiempo, puesto que tendremos que dar forma al programa y las librerías que se encargarán del funcionamiento correcto, empleando para ello versiones de pruebas para localizar errores en el código implementado para los diferentes módulos. Paralelamente será necesario construir una barra de infrarrojos autónoma que supla a la barra original.

  Al ser una fase de desarrollo, y teniendo en cuenta lo que ello implica, podríamos encontrarnos en ella durante un período extenso de tiempo.

- Pruebas. Posteriormente, deberemos ensamblar los módulos implementados en la etapa anterior y verificar su correcto funcionamiento de forma exhaustiva en presencia de las dependencias que incluyen unos sobre otros.

- Implantación. Tras la superación de las pruebas pertinentes se estudiará la posibilidad de introducir alguna mejora en el software.

- Mantenimiento. Como no se supone un producto comercial, se obviará la etapa de mantenimiento.

Conocidas las distintas etapas a través de las cuales se desarrollará nuestro proyecto, podemos estipular que la duración del mismo será de aproximadamente siete meses y medio, lo que viene a ser treinta semanas, suponiendo que cada semana se realizará un trabajo de aproximadamente veinte horas. En el siguiente diagrama, podemos ver una estimación de la distribución del trabajo a realizar:

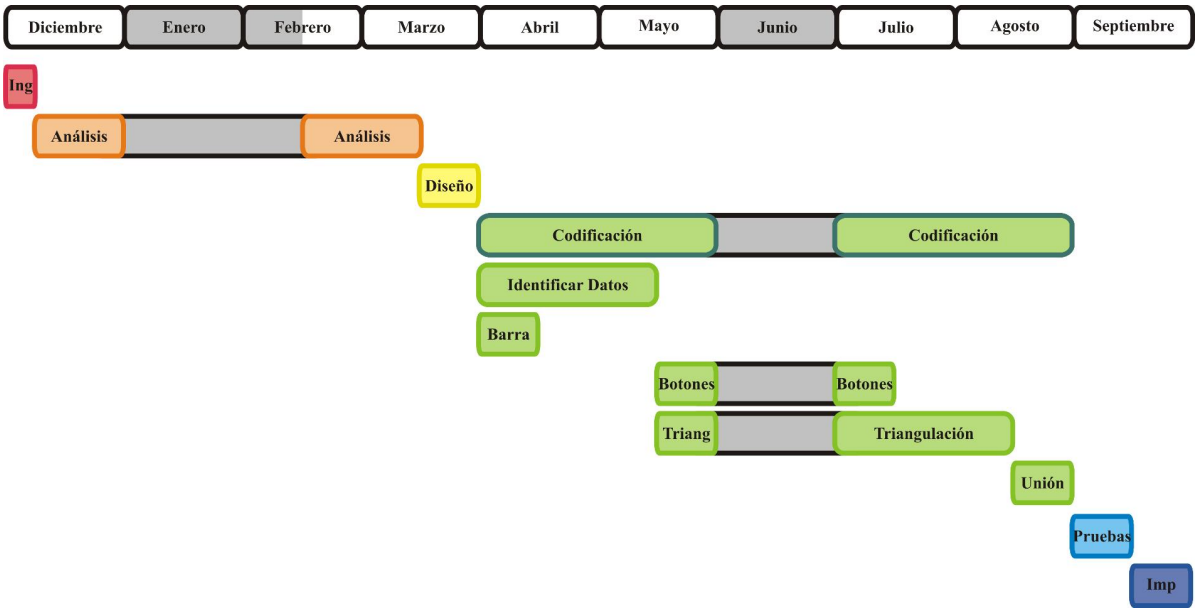


Figura 2.2: Diagrama de Gantt

2.3.2 Recursos

A continuación pasaremos a formalizar los recursos necesarios en el desarrollo de nuestro proyecto, incluyendo por una parte los recursos materiales y por otra los recursos humanos acorde con nuestra planificación.

- En primer lugar, al tratarse de un proyecto de desarrollo software principalmente, necesitaremos encarecidamente disponer de una máquina con la que trabajar. Como las especificaciones de dicha máquina no han de ser extremadamente buenas, podemos plantear el uso de un ordenador de sobremesa de gama media.

- El entorno de desarrollo se apoyará en un sistema operativo GNU/Linux, con lo cual nos evitaremos los costes derivados del pago de licencias al emplear software libre.

- Como material adicional para esta máquina, necesitaremos de manera imprescindible tener a nuestra disposición un dispositivo Bluetooth USB que nos permita realizar las conexiones pertinentes con el mando.

- Por otra parte, necesitaremos material electrónico para la fabricación de la barra de infrarrojos: placas perforadas, LEDs infrarrojos, cable de cobre, pilas recargables, soporte para las mismas, etc.

- Además, el dispositivo central e imprescindible para el desarrollo de este proyecto es el Wiimote de Nintendo.

- También, y aunque no suela tenerse en cuenta por parecer demasiado evidente, supondremos un coste adicional proveniente de otros materiales, como pueda ser el correspondiente al material de oficina.

- Así mismo se tendrán en cuenta los recursos humanos, donde computarán el número de horas dedicadas por el personal dedicado al proyecto, en este caso el autor de este proyecto.

CAPÍTULO 3

Materiales y Métodos

En este capítulo nos centraremos en explicar las principales herramientas con las cuales vamos a trabajar para desarrollar nuestro proyecto, ya que, una vez conocidas éstas, podremos centrarnos en los aspectos importantes a aplicar.

Para ello, vamos a explicar brevemente la tecnología que empleamos para las comunicaciones con el Wiimote, así como el lenguaje de programación elegido para el desarrollo del proyecto y las llamadas al sistema necesarias para poder completar el mismo.

3.1 Bluetooth

La tecnología inalámbrica Bluetooth [CC98] o IEEE 802.15.1 [oEE05], que aparece en 1998, es una especificación que define redes de área personal inalámbricas (Wireless Personal Area Network, WPAN, redes inalámbricas para la conexión de los dispositivos próximos a una persona). Esta tecnología soporta hasta 8 ó 10 dispositivos, dependiendo de la versión, en una piconet o WPAN.

Está desarrollada por Bluetooth Special Interest Group (Bluetooth SIG [SIG]), un grupo de empresas interesadas en la creación de dicha tecnología y que, hoy en día, recoge a más de 11.000 compañías como miembros del grupo desarrollando, implementando y vendiendo cientos de productos con Bluetooth en el mundo.

A partir de su versión 1.1, sus niveles más bajos (en concreto, el nivel físico y el control de acceso al medio) se formalizan también en el estándar IEEE 802.15.1. Bluetooth opera en la banda de 2,4 GHz y ofrece hasta 1 Mbps, que se reducen a 434 Kbps al descontar la sobrecarga de los protocolos. En 2004 se definió la norma Bluetooth v2.0 con transmisión de datos mejorada (EDR), con una velocidad de transmisión de 3Mbps (1.307 Kbps para datos). La versión más reciente es la 2.1, publicada en julio de 2007.

En las siguientes tablas se recogen algunos de los datos más representativos y descriptivos de las distintas clases y versiones de Bluetooth:

	Clase 1	Clase 2	Clase 3
Potencia máxima permitida (mW)	100	2.5	1
Potencia máxima permitida (dB)	20	4	0
Rango (aproximado)	100 m	20 m	1 m
Aplicaciones	Sector industrial	Dispositivos portátiles	

Cuadro 3.1: Especificaciones Bluetooth

Versión	1.2	2.0 + EDR	UWB
Ancho de Banda	1 Mbps	3 Mbps	200 Mbps

Cuadro 3.2: Anchos de Banda Bluetooth

3.1.1 Visión general

La especificación principal de Bluetooth (denominada core) define el nivel físico (PHY) y el control de acceso al medio (MAC) de una red inalámbrica de área personal. Este tipo de redes tienen por cometido la transferencia de información en distancias cortas entre un grupo privado de dispositivos. A diferencia de las LAN inalámbricas, están diseñadas para no requerir infraestructura alguna, o muy poca. Aún más, su comunicación no debería trascender más allá de los límites de la red privada.

El objetivo es lograr redes ad hoc simples de bajo coste y consumo. Para ello, Bluetooth define un espacio de operación personal (personal operating space) omnidireccional en el seno del cual se permite la movilidad de los dispositivos. Se definen tres tipos de dispositivos con diferentes rangos de acción: las clases 1 (cien metros), 2 (diez) y 3 (uno).

El estándar realiza la formalización de estas ideas y se concibe como una solución para evitar el uso de cableado en las comunicaciones. La especificación principal define el sistema básico, pero su diseño potencia la flexibilidad. Por ello, hay multitud de opciones, definidas por los perfiles Bluetooth en especificaciones complementarias.

3.1.2 Comparación técnica

El número de productos con tecnología de radiofrecuencia aumenta a gran velocidad día a día. Adoptar dichas tecnologías, cosa que hace 15 años parecía impensable, nunca ha sido tan fácil. Por este motivo es fundamental que los fabricantes de productos, y en especial con tecnología Bluetooth que es la que nos ocupa, tengan presentes ciertas directrices básicas durante el desarrollo de sus productos procurando cumplir los aspectos definidos en la especificación.

La tecnología inalámbrica Bluetooth cuenta con las siguientes ventajas:

- Es una tecnología inalámbrica orientada a aplicaciones de voz y datos.
- Su funcionamiento se enmarca en la banda de frecuencia de 2,4 GHz, banda libre, con lo cual no precisa de ninguna licencia.
- Su radio de acción, dependiendo de la clase de dispositivo Bluetooth de la que estemos hablando, es de 10 a 100 metros, con velocidades de transmisión de hasta 3 Mbps.
- Los objetos sólidos no suponen ningún obstáculo, más allá de la atenuación correspondiente, para la tecnología inalámbrica Bluetooth.
- Al ser una tecnología de radiofrecuencia, no es necesario que los dispositivos Bluetooth se encuentren situados en la misma línea de visión, es decir, orientados uno frente a otro ya que su diagrama de radiación depende directamente de la antena empleada, y éste suele ser omnidireccional.
- Una de las prioridades en el desarrollo de la tecnología Bluetooth ha sido siempre la seguridad, ofreciendo la especificación tres modos diferentes de la misma.
- El coste de producción de chips Bluetooth es inferior a tres dólares estadounidenses.

Esto nos hace pensar en la tecnología inalámbrica Bluetooth como una excelente solución frente al empleo de cables, pero siempre dependiendo de la aplicación para la cual vayamos a utilizarlo. En la siguiente tabla, proporcionada por Bluetooth SIG, podemos ver una comparativa entre varias tecnologías de radiofrecuencia según factores determinantes a la hora de elegir una u otra para cierta aplicación:

		ZigBee	Bluetooth	802.11				UWB
				b	g	a	n	
Throughtput	Mbps	0.03	1-3	11	54	54	200	200
Max Range	ft	75	30	200	200	150	150	30
Sweet Spot	Mbps-ft	.03@75	1-3@10	2@200	2@200	36@100	100@100	200@10
Service	bps-ft2	530	314M	251G	251G	1.13T	3.14T	62G
Power	mW	30	100	750	1000	1500	2000	400
BW	MHz	0.6	1	22	20	20	40	500
Spectral efficiency	b/Hz	0.05	1	0.5	2.7	2.7	5	0.4
Power efficiency1	mW/Mbps	1000	100	68	19	27	10	2
Power efficiency2	mAh/GB	2211	67	46	12	18	7	1.3
TTGB	Time	3.1 day	2.2 hr	12 min	2.5 min	2.5 min	40 sec	40 sec
Precio	US$	2	3	5	9	12	20	7

Cuadro 3.3: Comparación Técnica

3.1.3 Pila de protocolos Bluetooth

El núcleo del sistema Bluetooth se estructura en cuatro capas inferiores con protocolos asociados definidos por las especificaciones o perfiles Bluetooth. También incluye un protocolo de comunicación entre capas a nivel de servicios: el Protocolo de Descubrimiento de Servicios (SDP), que determina los servicios Bluetooth disponibles, y un Perfil de Acceso Genérico (GAP), que especifica los requisitos generales de los perfiles.

Una aplicación Bluetooth completa precisa varios servicios adicionales y protocolos de capas superiores, que se definen en la especificación Bluetooth.

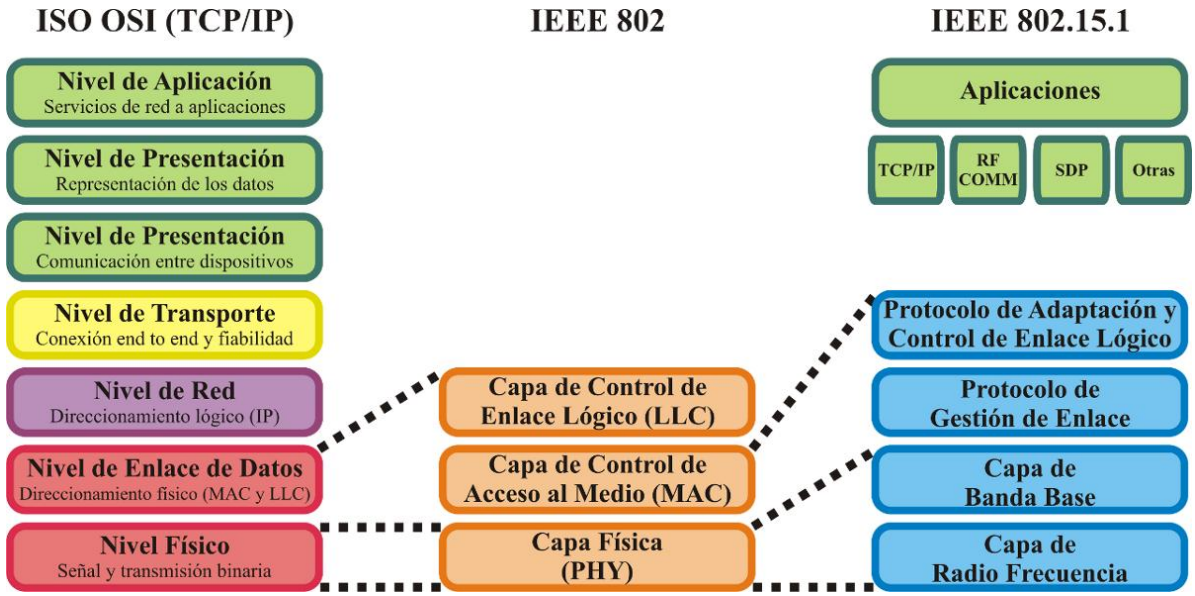


Figura 3.1: Comparación de Modelo de Capas OSI y 802.15.1

Los niveles inferiores de la pila de protocolos constituyen el controlador Bluetooth, que contiene los bloques fundamentales de la tecnología, sobre los cuales se apoyan los niveles superiores y los protocolos de aplicación. Este componente está estandarizado y puede interactuar con otros sistemas Bluetooth de más alto nivel, aunque la separación entre ambas entidades no es obligatoria.

El resto de niveles de base y los protocolos de aplicación residen en el anfitrión Bluetooth (también denominado host), que se comunica con el controlador utilizando un interfaz estándar (HCI). Ambas entidades pueden integrarse para su uso conjunto en sistemas empotrados, o se pueden utilizar de forma intercambiable.

Para que un dispositivo pueda utilizar la tecnología inalámbrica Bluetooth, debe saber interpretar los perfiles Bluetooth, que describen las distintas aplicaciones posibles. Estos perfiles son guías que indican los procedimientos por los que los dispositivos equipados con tecnología Bluetooth se comunican entre sí.

3.2 Python

Python [vR91] es un poderoso lenguaje de programación dinámico que se utiliza en una amplia variedad de dominios de aplicación. Python es a menudo comparado con Tcl, Perl, Ruby, Scheme o Java. Algunos de sus principales características distintivas son:

- sintaxis muy clara y legible,
- capacidad fuerte de introspección,
- orientación a objetos intuitiva,
- expresión natural del código de procedimiento,
- modularidad completa, soportando paquetes jerárquicos,
- control de errores basado en excepciones,
- tipos de datos dinámicos de muy alto nivel,
- librería estándar extensa y módulos de terceros para la labor de prácticamente cualquier tarea,
- extensiones y módulos fácilmente escritos en C, C++ (o Java para Jython, o .NET para IronPython),
- empotrable dentro de las aplicaciones como una interfaz de secuencias de comandos.

Python es un lenguaje de programación que permite trabajar más rápido e integrar sistemas de manera más eficaz. Se puede aprender a usar Python y ver casi de inmediato las ganancias de la productividad y reducir los costos de mantenimiento.

3.2.1 Python es poderoso... y rápido

Los fans de Python utilizan la frase "incluye las pilas" para describir la librería estándar [Pytc], que abarca todo, desde el procesamiento asíncrono a los ficheros zip. El lenguaje en sí es un gran aliado que puede manejar prácticamente cualquier problema. Se puede construir un servidor web en tres líneas de código.

Python permite escribir el código que se necesita rápidamente y, gracias a un compilador altamente optimizado y librerías de soporte, el código Python ejecuta más rápido de lo necesario para la mayoría de las aplicaciones.

3.2.2 Python juega bien con otros

Python se puede integrar con objetos COM, .NET y CORBA. Para las librerías Java, se puede usar Jython, una implementación de Python para la Máquina Virtual de Java. Para .NET, se puede usar IronPython, la nueva implementación de Microsoft de Python para .NET o Python para .NET. Python es también compatible con el motor de las comunicaciones de Internet (ICE) y muchas otras tecnologías de integración.

Si existe algo que Python no puede hacer, o si se necesita la ventaja de rendimiento del código de bajo nivel, se pueden escribir módulos de extensión en C o C++, o añadir código existente con SWIG o Boost.Python. Los módulos añadidos aparecen en el programa exactamente igual que el código Python nativo. También se puede ir en el camino opuesto e integrar Python en tu propia aplicación.

3.2.3 Python ejecuta en todas partes

Python está disponible para todos los principales sistemas operativos: Windows, Linux/Unix, OS/2, Mac, Amiga, entre otros. Hay incluso versiones que se ejecutan en .NET, la máquina virtual de Java, y teléfonos móviles Nokia Serie 60. El mismo código fuente funcionará sin cambios en todas las implementaciones.

Incluso puede que otros sistemas operativos no mencionados todavía soporten Python si existe un compilador de C para ellos. Se puede pedir a la Fundación o simplemente intentar compilarlo por ti mismo.

3.2.4 Python es amigable... y fácil de aprender

El grupo de noticias de Python es conocido como uno de los más amigables. El promotor y la comunidad de usuarios mantienen un wiki, organizan conferencias internacionales y locales, hacen carreras de desarrollo, y contribuyen a repositorios de código online.

Python también incluye documentación completa [Pytb]. Los tutoriales en línea se dirigen tanto al programador experimentado como al recién llegado. Todos están diseñados para hacerte productivo con rapidez. La disponibilidad de libros de alta calidad completan el paquete de aprendizaje.

3.2.5 Python es Abierto

La implementación de Python está bajo una licencia de código abierto que lo hace libremente utilizable y distribuible, incluso para uso comercial. La licencia de Python está administrada por la "Python Software Foundation".

3.3 Xautomation

Xautomation [SJ] es un conjunto de programas de línea de mandatos para el control de las X y hacer "visual scraping"para encontrar cosas en la pantalla.

La interfaz de control permite movimiento del ratón, hacer clic, pulsar teclas, etc, y usa la extensión XTest para no tener los molestos problemas que se tienen con Xse cuando las aplicaciones ignoran ciertos eventos enviados.

El programa *visgrep* encuentra imágenes dentro de imágenes e informa de sus coordenadas, lo que permite a los programas encontrar botones, etc, en la pantalla para hacer clic en ellos.

Xautomation consiste en los siguientes programas:

- **pat2ppm:** convierte una imagen de formato PAT a formato PPM.
- **patextract:** extrae una parte de una imagen en formato PNG.
- **png2pat:** convierte una imagen de formato PNG a formato PAT.
- **rgb2pat:** convierte una imagen en RGB (24 bits) a formato PAT.
- **visgrep:** busca imágenes en otra imagen.
- **xte:** genera entradas falsas empleando la extensión XTest.

3.3.1 Xte

xte [Sla] es una utilidad para emular eventos del servidor X en tiempo de ejecución. Se puede utilizar con permisos de usuario normal, es decir, no requiere permisos de root.

Entre otras cosas permite emular eventos de ratón y de teclado. El manual de Linux nos proporciona información más detallada acerca de esta utilidad.

- **Nombre**

  xte - genera entradas falsas empleando la extensión XTest.

- **Sinopsis**

  xte [opciones] mandatos ...

- **Descripción**

  xte es un programa que genera entradas falsas utilizando la extensión XTest, más fiable que xse.

- **Opciones**

  A continuación se incluye un resumen de las opciones.

  - **-x display**

    Envía mandatos al servidor X remoto. Tenga en cuenta que algunos mandatos pueden no funcionar correctamente a no ser que la pantalla esté en la consola, por ejemplo, la pantalla está actualmente controlada por el teclado y el ratón y no en el fondo. Esto parece ser una limitación de la extensión XTest.

  - **–help, -h**

    Mostrar resumen de las opciones.

- **Mandatos**

  - **key *k***

    Presiona y libera la tecla *k*.

  - **keydown *k***

    Presiona la tecla *k*.

  - **keyup *k***

    Libera la tecla *k*.

  - **str *string***

    Realiza una serie de eventos **key *k*** para cada caracter en el string.

  - **mouseclick *i***

    Hace click con el botón *i* del ratón.

  - **mousemove *x y***

    Mueve el ratón a la posición (x,y).

  - **mousermove *x y***

    Mueve el ratón (x,y) respecto a la posición relativa actual.

  - **mousedown *i***

    Presiona el botón *i* del ratón.

  - **mouseup *i***

    Libera el botón *i* del ratón.

  - **sleep *x***

    Duerme x segundos.

- **usleep *x***

  Duerme x microsegundos.

- **Algunas teclas útiles**

  Estas teclas son sensibles a mayúsculas y minúsculas.

  - Home
  - Left
  - Up
  - Right
  - Down
  - Page_Up
  - Page_Down
  - End
  - Return
  - BackSpace
  - Tab
  - Escape
  - Delete
  - Shift_L
  - Shift_R
  - Control_L
  - Control_R
  - Meta_L
  - Meta_R
  - Alt_L
  - Alt_R

CAPÍTULO 4

Wiimote

El diseño del Wii Remote [Wiib] no se basa en los tradicionales mandos para los videojuegos. A diferencia de ellos, es similar a un control remoto de televisión, creado para ser utilizado con una sola mano y de la manera más intuitiva posible.

En su cara frontal, el Wiimote presenta los botones "A", "1", "2", "+", "-", "HOME", "POWER" y la cruceta de direcciones. En la parte anterior sólo presenta el botón "B", en un formato similar a un gatillo.

Figura 4.1: Wiimote

Adicionalmente, en su parte frontal incluye un altavoz y cuatro indicadores luminosos numerados que indican el número de jugador al que corresponde el mando y permiten ver el tiempo de vida de la batería.

4.1 Comunicación Bluetooth

El Wiimote se comunica con la Wii a través de un enlace Bluetooth permitido gracias a un controlador Broadcom BCM2042 que está diseñado para ser usado con dispositivos de interfaz humana (HID).

El BCM2042 es un gran avance en el diseño de ratones y teclados Bluetooth de bajo coste. Es un chip único que integra todos los perfiles, las aplicaciones y la pila de protocolos de Bluetooth y es totalmente compatible con la especificación de Bluetooth SIG para dispositivos de interfaz humana.

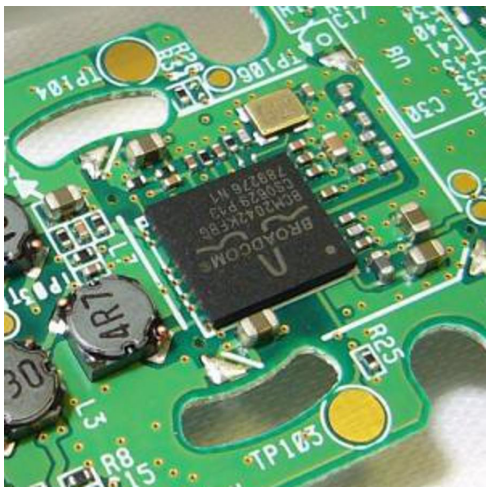


Figura 4.2: BCM2042

El BCM2042 es totalmente compatible con la versión 2.0 de la especificación Bluetooth, incluyendo salto de frecuencia adaptable y conexión de alta velocidad, que son esenciales para el ratón y el teclado de las aplicaciones en los ordenadores personales.

La integración es clave para lograr los objetivos de costes de los fabricantes. Mediante la integración de todos los componentes en el BCM2042, se reducen los costes del sistema y se asemejan a los precios de los dispositivos cableados.

- Características:
  - chip Bluetooth único con dispositivo de interfaz humana (HID) plenamente integrado y pila Bluetooth completa,
  - procesador 8051 y memoria RAM/ROM "on-board" (integrada),
  - solución optimizada en costes para aplicaciones de ratón, teclado y similares, que logra el menor coste posible a través de la integración de todos los componentes externos,
  - reemplaza dispositivos existentes añadiendo funcionalidad Bluetooth.

- Aplicaciones:
  - teclados y ratones,
  - dispositivos de control remoto.

Al realizar un escaneo de la zona para ver los dispositivos Bluetooth a nuestro alcance, podemos observar lo siguiente:

```
angel@laptop:~$ hcitool scan
Scanning ...
    00:19:1D:69:5C:55 Nintendo RVL-CNT-01
```

Cuadro 4.1: hcitool scan

El Wiimote es capaz de enviar mensajes con una frecuencia máxima de 100Hz.

4.1.1 Service Discovery Protocol

El protocolo de descubrimiento de servicios permite descubrir que servicios tienen disponibles los dispositivos Bluetooth vecinos. El protocolo no define como se tiene que conectar al servicio, solo informa de los servicios disponibles.

Trabaja con un esquema de petición-respuesta. Se envía una Protocol Data Unit (PDU) y se recibe otra PDU. Durante este proceso, cada protocolo de capa intercambia información entre capas iguales. Cada capa de comunicación en el origen, se comunica con su igual en el destino.

Cuando se le consulta con el SDP, el Wiimote reporta una gran cantidad de información.

```
angel@laptop:~$ sdptool browse 00:19:1D:69:5C:55
Browsing 00:19:1D:69:5C:55 ...
Service RecHandle: 0x0
Service Class ID List:
    "SDP Server" (0x1000)
Protocol Descriptor List:
    "L2CAP" (0x0100)
        PSM: 1
    "SDP" (0x0001)
Language Base Attr List:
    code_ISO639: 0x656e
    encoding: 0x6a
    base_offset: 0x100
Profile Descriptor List:
    "" (0x0100)
        Version: 0x0100
```

```
Service Name: Nintendo RVL-CNT-01
Service Description: Nintendo RVL-CNT-01
Service Provider: Nintendo
Service RecHandle: 0x10000
Service Class ID List:
    "Human Interface Device" (0x1124)
Protocol Descriptor List:
    "L2CAP"(0x0100)
        PSM: 17
    "HIDP" (0x0011)
Language Base Attr List:
    code_ISO639: 0x656e
    encoding: 0x6a
    base_offset: 0x100
Profile Descriptor List:
    "Human Interface Device" (0x1124)
        Version: 0x0100
Service RecHandle: 0x10001
Service Class ID List:
    "PnP Information" (0x1200)
Protocol Descriptor List:
    "L2CAP" (0x0100)
        PSM: 1
    "SDP" (0x0001)
Profile Descriptor List:
    "PnP Information" (0x1200)
        Version: 0x0100
```

Cuadro 4.2: sdptool browse

4.1.2 Human Interface Device

El dispositivo de interfaz humana (HID) da soporte a dispositivos tales como ratones, joysticks y teclados. También puede utilizarse para indicadores luminosos o botones en otro tipos de dispositivos. Se ha diseñado para ofrecer un enlace de baja latencia manteniendo bajo el consumo.

HID es una adecuación del protocolo original definido para USB. Su uso simplifica la implementación del anfitrión (en concreto, el soporte de USB es reutilizable para Bluetooth en sistemas operativos).

El Wiimote no requiere de la autenticación o el cifrado característicos de la norma Bluetooth. Con el fin de interactuar con él, primero hay que poner el controlador en modo "detectable" ya sea presionando los botones "1" y "2" al mismo tiempo, o bien pulsando el botón de sincronización rojo que se encuentra debajo de la cubierta de la batería.

Una vez en este modo, el Wiimote puede ser consultado por el controlador HID de Bluetooth del host. Si el controlador del host no se conecta al Wiimote en 20 segundos, el Wiimote se apagará de manera automática. Manteniendo pulsado los botones 1 y 2 continuamente, se forzará al Wiimote a permanecer en modo "detectable" sin que se apague de manera automática. Sin embargo, esto no funciona con el botón de sincronización.

Cuando el Wiimote se encuentra en modo "detectable", los LEDs del jugador parpadearán. El número de LEDs que parpadea en el Wiimote corresponderá a la duración de la batería restante, más concretamente con la carga actual de la batería, similar a la del medidor en el menú principal de Wii en el que el número de barras es igual al número de luces parpadeando.

El estándar HID permite que los dispositivos se describan a sí mismos, utilizando un bloque descriptor HID. Este bloque incluye una enumeración de las instrucciones que el dispositivo comprende.

Un canal puede considerarse similar a un puerto de red asignado a un servicio en particular. Sin embargo, los canales son unidireccionales, y los descriptores HID listan para cada puerto la dirección (entrada o salida) y el tamaño de la carga útil para cada puerto. Como todos los dispositivos Bluetooth HID, el Wiimote informa de su bloque descriptor HID cuando se le pregunta utilizando el protocolo SDP (como hemos visto anteriormente).

Una versión legible para humanos del bloque descriptor se resume en las siguientes tablas:

Canal	**Carga útil (bytes)**	**Funciones**
0x20	6	Puerto de Expansión
0x21	21	Lectura de datos
0x22	4	Escritura de datos
0x30	2	Botones
0x31	5	Botones - Sensor de movimiento
0x32	16	Botones - Puerto de Expansión - IR
0x33	17	Botones - Sensor de movimiento - IR
0x34	21	Botones - Puerto de Expansión - IR
0x35	21	Botones - Sensor de movimiento - Puerto de Expansión
0x36	21	Botones - Puerto de Expansión - IR
0x37	21	Botones - Sensor de movimiento - Puerto de Expansión
0x38	21	Botones - Sensor de movimiento - IR
0x3D	21	Botones - Puerto de Expansión - IR
0x3E	21	Botones - Sensor de movimiento - IR
0x3F	21	Botones - Sensor de movimiento - IR

Cuadro 4.3: Entradas (0xA1)

Canal	Carga útil (bytes)	Funciones
0x11	1	LEDs, Vibración
0x12	2	Informe/ID
0x13	1	Habilitar Sensor IR
0x14	1	Habilitar altavoz
0x15	1	Estado del controlador
0x16	21	Escritura de datos
0x17	6	Lectura de datos
0x18	21	Datos altavoz
0x19	1	Silenciar altavoz
0x1A	1	Habilitar Sensor IR 2

Cuadro 4.4: Salidas (0x52)

Nótese que "Entradas" se refiere a los paquetes que son enviados desde el Wiimote al servidor, mientras que "Salidas" se refiere a los paquetes que son enviados desde el servidor al Wiimote. Para más claridad, la convención que adoptaremos en este documento será mostrar los paquetes con la cabecera Bluetooth (sentido de la comunicación), el canal (funciones) y la carga útil. Cada byte se encuentra escrito en hexadecimal, por ejemplo:

0xA1 0x30 0x00 0x00

Cuadro 4.5: Mensaje de ejemplo

es un paquete de entrada de datos (0xA1), en el canal 0x30 (botones) con ambos bytes de carga útil a 0x00.

4.2 Entradas

Como ya hemos mencionado anteriormente, con "Entradas" nos referiremos a los paquetes que son enviados por el Wiimote al servidor, es decir, la Wii o en nuestro caso el portátil.

4.2.1 Botones

Como hemos mencionado, el Wiimote presenta 12 botones, cuatro de los cuales se corresponden con la cruceta de direcciones y el resto se encuentran esparcidos por todo el mando de control: "A", "B", "1", "2", "+", "-", "HOME" y "POWER".

Además, existe un decimotercer botón tras la tapa de las baterías que, al ser presionado, hace que el Wiimote corte cualquier conexión y se mantenga en modo "detectable" durante 20 segundos exactamente, sin permitir que se mantenga en este modo de manera indefinida.

Físicamente, el hardware de los botones varía entre interruptores de membrana y pulsadores. La siguiente tabla describe el hardware físico para cada entrada:

Botón	**Tipo**
2	Membrana
1	Membrana
B	Membrana
A	Membrana
-	Pulsador
Home	Pulsador
←	Membrana
→	Membrana
↓	Membrana
↑	Membrana
+	Pulsador
Power	Pulsador
Sync	Pulsador

Cuadro 4.6: Valor de los botones

De manera predeterminada, al realizar una nueva conexión Bluetooth con el Wiimote, siempre que se presione un botón o sea liberado, se envía un paquete de datos al servidor a través de la entrada HID (0xA1) por el canal 0x30 con una carga útil de dos bytes en la cual se comunica el estado actual de todos los botones.

De las pruebas realizadas, se puede generalizar que dicho mapeo de los botones se encuentra en todos los dos primeros bytes de la carga útil en cualquier canal de entrada. A pesar de esto, algunos de los bits de esta pareja de bytes no parecen estar directamente relacionados con la pulsación de los botones.

Por ejemplo, en una nueva conexión, cuando el botón "A" es presionado, se recibe en el servidor un paquete HID como el siguiente:

0xA1 0x30 0x00 0x08

Cuadro 4.7: Botón "A" presionado

que es un paquete de entrada de datos (0xA1), en el canal 0x30 (botones) con el segundo byte de carga útil a 0x08, lo cual indica que el botón "A" ha sido presionado.

Del mismo modo, cuando el botón es liberado, se recibe otro paquete como este:

0xA1 0x30 0x00 0x00

Cuadro 4.8: Botón "A" liberado

que es un paquete de entrada de datos (0xA1), en el canal 0x30 (botones) con ambos bytes de carga útil a 0x00, lo cual indica que ningún botón se encuentra presionado.

De esta manera podemos asignar cada botón a un bit, de tal modo que la máscara queda resumida en la siguiente tabla:

Botón	**Valor (hexadecimal)**
2	0x00 0x01
1	0x00 0x02
B	0x00 0x04
A	0x00 0x08
-	0x00 0x10
Home	0x00 0x80
←	0x01 0x00
→	0x02 0x00
↓	0x04 0x00
↑	0x08 0x00
+	0x10 0x00

Cuadro 4.9: Valor de los botones

Cabe destacar que el botón "POWER" funciona de una manera inusual, ya que en vez de enviar un paquete HID que informe de que ha sido presionado, envía una petición de desconexión Bluetooth al servidor.

4.2.2 Sensor de movimiento

El movimiento del Wiimote es detectado por un acelerómetro lineal de 3 ejes que se encuentra ligeramente a la izquierda del botón "A". El circuito integrado que se lo permite es el ADXL330, fabricado por Analog Devices.

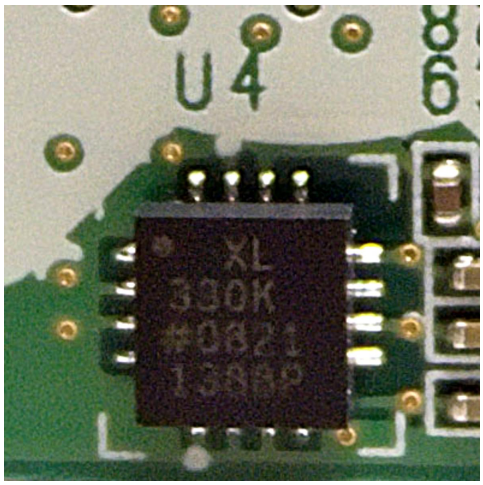


Figura 4.3: ADXL330

El ADXL330 es un pequeño y completo acelerómetro de tres ejes de bajo coste y consumo, con acondicionamiento de señal en las salidas de voltaje, montado en un circuito integrado monolítico.

Es capaz de medir la aceleración en un rango máximo de ±3g con un 10 % de sensibilidad; puede medir la aceleración estática de la gravedad para aplicaciones en las que sea necesario medir la inclinación, así como la aceleración dinámica resultante del movimiento, choques o vibraciones.

El usuario selecciona el ancho de banda del acelerómetro usando una serie de condensadores C_X, C_Y y C_Z en los pines X_{OUT}, Y_{OUT} y Z_{OUT}. Los anchos de banda pueden ser seleccionados para satisfacer la aplicación, con un rango desde 0.5 Hz hasta 1600 Hz para los ejes X e Y, y un rango desde 0.5 Hz hasta 550 Hz para el eje Z.

El ADXL330 está disponible en un integrado pequeño de bajo perfil, de 4 mm x 4 mm x 1.45 mm, 16 patillas, y encapsulado de plástico (LFCSP_LQ).

- Características:
  - sensibilidad en tres ejes,
  - pequeño tamaño,
  - bajo consumo,
    - 180 μA a V_S = 1.8 V (típica)
  - tensión de alimentación única,
    - 1.8 V a 3.6 V
  - supervivencia a choques de 10000g,
  - excelente estabilidad con la temperatura,
  - ajuste del ancho de banda con un único condensador por eje.
- Aplicaciones:
  - dispositivos móviles,
  - sistema de videojuegos,
  - protección de unidades de disco,
  - estabilización de imágenes,
  - dispositivos para deportes y salud.

Dentro del chip se encuentra una pequeña estructura mecánica que se apoya en resortes fabricados en silicio. Así, mediciones de diferencias en la capacitancia permiten convertir el desplazamiento neto de la pequeña masa convirtiéndose en una tensión, que seguidamente es digitalizada.

Es importante señalar que el sensor no mide la aceleración del Wiimote, sino más bien la fuerza ejercida por la masa de prueba en sus resortes de apoyo. Debido a la convención de signos utilizada, esta cantidad es proporcional a la fuerza neta ejercida por la mano del jugador en el Wiimote cuando lo sostiene. Así, en reposo sobre una mesa plana, el acelerómetro informa de la fuerza vertical de +g normalizada, y al dejarlo caer informa de una fuerza cercana a cero.

El sensor utiliza un sistema de coordenadas diestro, con el eje x positivo hacia la izquierda, y el eje z positivo apuntando hacia arriba, cuando el mando está en posición horizontal, como se muestra en la figura.

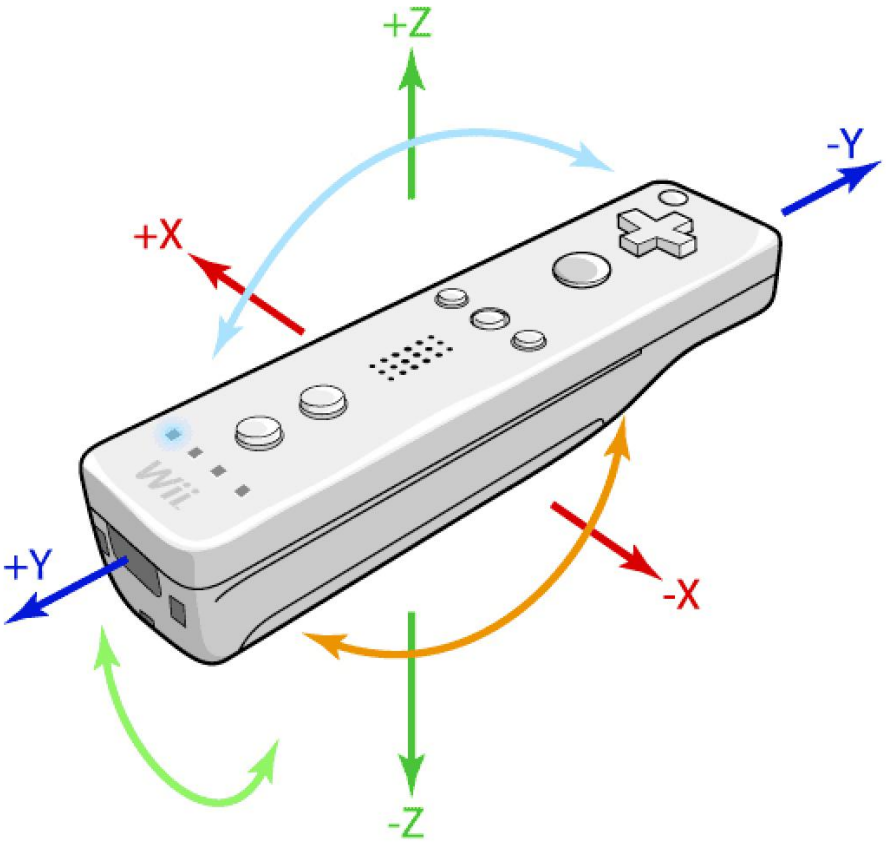


Figura 4.4: Sistema de coordenadas del Wiimote

Las fuerzas en cada eje están digitalizadas a 8 bits sin signo, con la escala de cero establecida en 0x80. Por supuesto, defectos de fabricación y de calibración hacen que existan algunos desplazamientos intrínsecos del cero.

El Wiimote no suele informar lecturas del sensor de movimiento al servidor, pero pueden ser solicitadas enviando la siguiente petición al canal 0x12:

0x52 0x12 0x00 0x31

Cuadro 4.10: Inicialización del sensor de movimiento

El tercer byte es una máscara. Por ejemplo, 0x01 habilita el vibrador y 0x04 habilita salida contínua. Si no se envía 0x04, solo se envían paquetes al servidor cuando los valores cambian (casi siempre cuando el sensor de movimiento está habilitado). Si se envía 0x04, se envían valores continuamente.

En modo continuo, el dispositivo enviará datos a una velocidad de 100 paquetes por segundo. Es decir, con el tercer byte a 0x04, se informa de los valores de los botones múltiples veces por segundo, sin embargo, con el tercer byte a 0x00, solamente se informa de dichos valores cuando se presiona o libera un botón.

El cuarto byte especifica que canal HID debe ser seleccionado y por tanto qué información de sensores estamos pidiendo. Tras recibir este mensaje una vez, el Wiimote enviará continuamente paquetes de entrada en el canal seleccionado (0x31 en el ejemplo de arriba), en los cuales para los canales 0x31, 0x33, 0x35 y 0x37 los bytes quinto, sexto y séptimo contienen las lecturas X, Y y Z del acelerómetro.

Un paquete de prueba cuando el Wiimote se encuentra en reposo sobre una mesa plana es el siguiente:

0xA1 0x31 0x40 0x20 0x86 0x8A 0xA5

Cuadro 4.11: Paquete del sensor de movimiento

donde 0x86 es la lectura del eje X, 0x8A es la lectura del eje Y y 0xA5 es la lectura del eje Z. Los dos primeros bytes de la carga útil, 0x40 y 0x20, se siguen correspondiendo con el valor de los botones sin cambio alguno, como se explicó en la sección correspondiente.

Como explicamos en la sección de HID, la longitud de la carga útil depende del canal. De esta manera, canales que tengan cargas útiles mayores que la necesitada para informar del estado del sensor de movimiento, devolverá el resto de bytes rellenos a 0xFF. Por ejemplo, para el canal 0x33:

0xA1 0x33 0x40 0x00 0x86 0x8A 0xA5 0xFF [...] 0xFF

Cuadro 4.12: Paquete largo del sensor de movimiento

Los informes del sensor de movimiento se pueden parar en cualquier momento enviando una petición del canal 0x30 (solo botones):

0x52 0x12 0x00 0x30

Cuadro 4.13: Detener el sensor de movimiento

4.2.3 Sensor de infrarrojos

El Wiimote también cuenta con un sensor óptico PixArt, lo que le permite determinar el lugar al que el Wiimote está apuntando. Dicho sensor se diferencia del resto en que fue fabricado exclusivamente para esta aplicación, y sus características se han mantenido en secreto sin haberse hecho público ningún datasheet por parte del fabricante.

A pesar de esto, sabemos que es una cámara monócroma de 128x96 con procesamiento de imagen incluído. Este procesamiento incluye análisis de subpixels (8x) para proporcionar una imagen de 1024x768. El procesamiento que incluye es capaz de posicionar hasta cuatro objetos moviéndose en su campo de visión proporcionando sus coordenadas, siendo éstas los únicos valores que puede conocer el servidor. De esta manera el servidor no podrá acceder a los datos originales, y por tanto, no se puede emplear la cámara para tomar una imagen convencional.

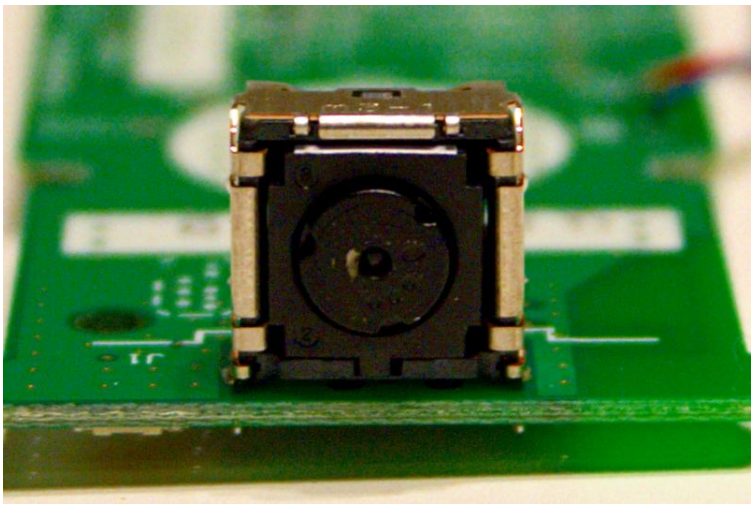


Figura 4.5: Cámara PixArt

La cámara recoge lo que le llega a través de un filtro infrarrojo en la cubierta del Wiimote con una apertura efectiva de unos 35º en horizontal y 25º grados en vertical. Con el filtro intacto, las fuentes de 940nm son detectadas con, aproximadamente, el doble de la intensidad equivalente para fuentes de 850nm; si este filtro le es retirado, será capaz de reconocer cualquier objeto brillante.

El uso de un sensor infrarrojo para detectar posición puede causar algunos problemas cuando otras fuentes de infrarrojos se encuentran alrededor, como bombillas incandescentes o velas. Esto puede ser fácilmente mitigado por el uso de luces fluorescentes alrededor de la Wii, ya que emiten poca o ninguna luz infrarroja.

Barra emisora de infrarrojos

La barra de infrarrojos original está alimentada por la Wii y contiene 2 grupos de LEDs infrarrojos espaciados 20 centímetros de centro a centro. Cada grupo se compone de 5 LEDs que operan en una longitud de onda de 940 nm.

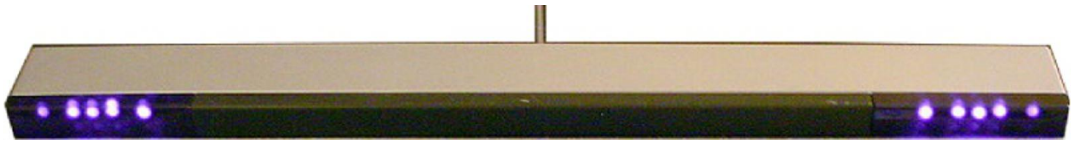


Figura 4.6: Barra emisora de infrarrojos

Los dos LEDs exteriores se encuentran ligeramente inclinados hacia los extremos y los dos LEDs interiores se encuentran ligeramente inclinados hacia el centro de la barra, mientras que los tres LEDs centrales apuntan directamente hacia adelante agrupados. Esta posición de los LEDs mejora el margen de la barra emisora de infrarrojos. Los LEDs encendidos pueden verse a través de algunas cámaras y otros dispositivos con un mayor espectro visible que el ojo humano.

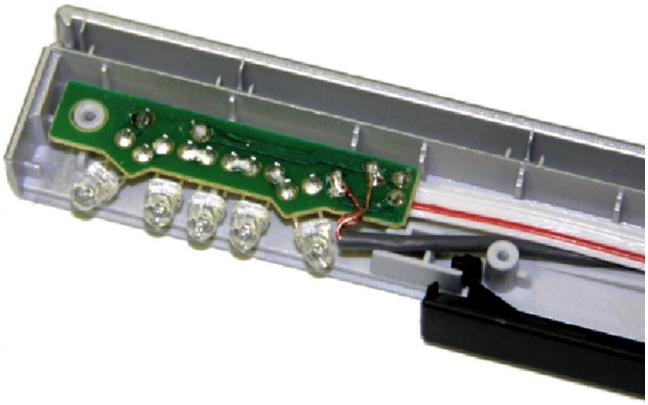


Figura 4.7: LEDs infrarrojos

A diferencia de lo que se pudiera pensar intuitivamente, el cable de la barra emisora de infrarrojos solamente transporta energía para alimentarla sin que se efectúe ningún traspaso de información o se module la intensidad de alguna manera. La salida de la Wii hacia la barra de infrarrojos compone una alimentación de 12 V de corriente continua, pero se ha comprobado que debería funcionar con una batería de 9 V.

El Wiimote detecta las fuentes de luz infrarroja de la barra emisora que ocuparán una posición fija y conocida, siendo colocada en la parte superior o inferior de la pantalla de manera que quede centrada, lo que permite el uso de dichas fuentes sin necesidad de indicarle el tamaño o el tipo de esta.

Si está colocada por encima, el sensor debe estar alineado con la parte delantera de la pantalla, y si está colocada en la parte inferior, debe alinearse con la parte delantera de la superficie sobre la que se coloca. No es necesario señalar directamente a la barra emisora de infrarrojos, pero apuntar significativamente fuera de ésta perturbará la capacidad de detección debido al limitado ángulo de visión del Wiimote.

Funcionamiento

La barra emisora de infrarrojos permite al Wiimote ser empleado como un dispositivo de señalamiento preciso hasta los 5 metros de distancia de la barra. Como hemos explicado, el sensor óptico identifica los puntos de luz de cada grupo de LEDs y los localiza en su campo de visión. De esta manera, conociendo la separación de los puntos proporcionados por el sensor óptico y la separación real entre las dos fuentes emisoras de infrarrojos, la Wii es capaz de calcular la distancia entre el Wiimote y la barra emisora de infrarrojos utilizando una sencilla triangulación.

La barra emisora de infrarrojos es necesaria cuando el Wiimote está controlando movimientos arriba-abajo, izquierda-derecha de un cursor en la pantalla. Además, la rotación del Wiimote con respecto al suelo también puede ser calculada a partir del ángulo relativo de los dos puntos de luz en el sensor de imagen.

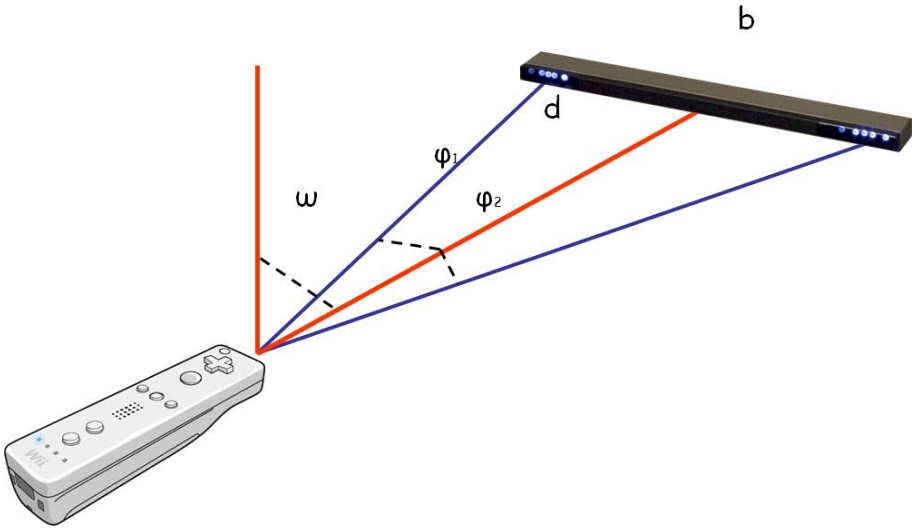


Figura 4.8: Esquema de triangulación

Debido a que la Barra de sensores también permite calcular la distancia entre el mando y la sensibilidad de la Barra, el Wiimote también puede controlar el movimiento lento adelante-atrás. El movimiento adelante-atrás está controlado por los sensores de aceleración. Usando estos sensores de aceleración (que actúan como sensores de inclinación), el Wiimote también puede controlar la rotación de un cursor u otros objetos.

Se ha comprobado que se pueden emplear otras fuentes de luz infrarroja, como bombillas incandescentes o un par de mecheros. La posición y seguimiento del movimiento del Wiimote permite al usuario imitar acciones reales, en lugar del clásico mando en el que solamente se podía interactuar pulsando los botones.

Formato de datos

Cuando el sensor óptico reconoce un objeto, lo asigna al primer objeto disponible y así sucesivamente, de tal manera que si el primero de ellos sale del margen de visión, ese objeto se marca como vacío (0xFF) y los demás mantienen sus posiciones en la carga útil del paquete. Por ejemplo, si se reconocen dos objetos y uno de ellos se sale de la visión de la cámara, un paquete podría ser:

Objeto 1	0xFF	0xFF	0xFF
Objeto 2	0xA1	0x80	0x10
Objeto 3	0xFF	0xFF	0xFF
Objeto 4	0xFF	0xFF	0xFF

Cuadro 4.14: Información de objetos IR

Con más de cuatro objetos en el campo de visión, la cámara será incapaz de mantenerse en cuatro e irá saltando entre ellos.

La cámara infrarroja puede devolver diferentes conjuntos de valores de datos describiendo los objetos que reconoce. El canal seleccionado para obtener la información del Wiimote deberá tener una carga útil que coincida con el número de bytes que nos va a devolver (ni más ni menos), dependiendo del modo de informe que hayamos elegido. Los modos son:

Modo	Valor
Básico	1
Extendido	3
Completo	5

Cuadro 4.15: Modos de funcionamiento del sensor óptico

Modo básico

En modo básico, la cámara devuelve 10 bytes de datos correspondientes a las coordenadas X e Y de cada uno de los cuatro objetos. Cada localización se encuentra codificada en 10 bits con un rango de 0-1023 para la dimensión X y 0-767 para la dimensión Y.

Cada par de objetos se empaqueta en 5 bytes, transmitiendo dos de estos pares para sumar un total de cuatro puntos con 10 bytes.

El formato de paquete para un par de objetos es el siguiente:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	X_1<7:0>							
1	Y_1<7:0>							
2	Y_1<9:8>		X_1<9:8>		Y_2<9:8>		X_2<9:8>	
3	X_2<7:0>							
4	Y_2<7:0>							

Cuadro 4.16: Paquete de modo básico

Modo extendido

En modo extendido, la cámara devuelve 12 bytes de datos correspondientes a las coordenadas X e Y de cada uno de los cuatro objetos, tal y como hacía el modo básico, con la peculiaridad de incluir un tercer valor de "tamaño" con un rango de 0-15.

Ahora cada localización se envía de manera independiente a la de los otros objetos, siendo codificada toda la información referente a cada objeto en 3 bytes que se concatenarán con los del resto de objetos para formar el paquete de 12 bytes.

El formato de paquete para un objeto es el siguiente:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	X<7:0>							
1	Y<7:0>							
2	Y<9:8>		X<9:8>		S<3:0>			

Cuadro 4.17: Paquete de modo extendido

Modo completo

En modo completo, la cámara devuelve 36 bytes de datos que son eviados en dos paquetes consecutivos de 18 bytes cada uno. Los tres primeros bytes se corresponden con los del modo extendido, donde se informaba de las coordenadas X e Y de los objetos y de la sensibilidad.

Estos bytes son ahora seguidos por otros informando de los valores que delimitan una máscara rectangular en la cual se encuentra contenido el pixel de las coordenadas X e Y. Además, se incluye un valor de intensidad con que la cámara reconoce a ese objeto.

Como vemos, la información referente a cada objeto se codifica en 9 bytes a pesar de dejar espacios vacíos, puesto que de esta manera se pueden conformar los dos paquetes de 18 bytes de forma sencilla.

El formato de paquete para un objeto es el siguiente:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	X<7:0>							
1	Y<7:0>							
2	Y<9:8>		X<9:8>		S<3:0>			
3	0	X_{min}<6:0>						
4	0	Y_{min}<6:0>						
5	0	X_{max}<6:0>						
6	0	Y_{max}<6:0>						
7	0							
8	Intensidad<7:0>							

Cuadro 4.18: Paquete de modo completo

4.3 Salidas

Como ya hemos comentado con anterioridad, con "Salidas" nos referiremos a los paquetes que son enviados por el servidor al Wiimote, es decir, desde la Wii o en nuestro caso el portátil.

4.3.1 LEDs

La parte inferior del Wiimote contiene cuatros LEDs azules. Estos LEDs se usan durante su uso para indicar que el mando se encuentra en modo "detectable" (indicando el estado de carga de las baterías mientras parpadean), o para indicar el número de jugador de ese mando (un único LED fijo).

Figura 4.9: LEDs del Wiimote

Los LEDs son controlables de forma independiente mediante el envío de un paquete de datos al canal 0x11, con un tamaño de carga útil de 1 byte. Los cuatro bits más significativos controlan cada LED, con el bit menos significativo correspondiéndose con el LED del jugador 1. Así, las máscaras quedan resumidas en la siguiente tabla:

Bit	Valor (hexadecimal)	LEDs
4	0x10	
5	0x20	
6	0x40	
7	0x80	

Cuadro 4.19: Valor de los LEDs

El canal 0x11 también puede controlar la función de vibración, así que es mejor mantener los 4 bits menos significativos a cero, a fin de evitar la vibración del controlador. Por ejemplo, este paquete enciende el LEd del jugador 1 solamente:

```
0x52 0x11 0x10
```

Cuadro 4.20: Encender el LED del jugador 1

Ya que cada LED tiene su propio bit, cualquier combinación de LEDs puede ser iluminada. Actualizando la máscara de LEDs rápidamente (varias veces por segundo) hará que los cuatro testigos parpadeen, como si el Wiimote se encontrase en modo de emparejamiento. Una vez que se produzca un pequeño retraso desde la última actualización de la máscara de LEDs, la máscara más reciente se volverá visible, y el parpadeo se detendrá.

4.3.2 Vibración

La vibración es producida por un motor que tiene unido a su eje un peso no balanceado de tal manera que, al activar el motor, se produce la vibración.

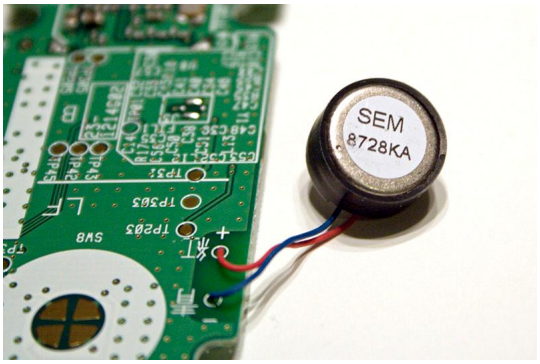


Figura 4.10: Motor del Wiimote

El despiece de distintos Wiimotes demuestra que se han utilizado motores de vibración de distintos modelos. Un ejemplo es el mostrado en la imagen, el SEM 8728KA. Otro que hemos podido observar es el SEM 8728DA. El Wiimote proporciona al motor una alimentación de 3.3 V y 35 mA. Es razonable pensar que se podría eliminar este componente y ser reemplazado por otro dispositivo que requiera las mismas especificaciones eléctricas.

Se puede activar el motor enviando un paquete al mismo canal con el que se controlan los LEDs con la siguiente carga útil:

0x52 0x11 0x01

Cuadro 4.21: Encender la vibración

Y se puede apagar, poniendo ese bit a cero:

0x52 0x11 0x00

Cuadro 4.22: Apagar la vibración

El motor puede ser activado, enviando ese mismo paquete a los canales 0x11, 0x13, 0x14, 0x15, 0x19 o 0x1A. Parece que todos los canales son equivalentes, pero no se recomienda emplear el canal 0x11 dado que controla los LEDs de los jugadores y podríamos obtener efectos indeseados. En lugar de eso, es recomendable emplear algún canal que no estemos empleando, por ejemplo:

0x52 0x13 0x01

Cuadro 4.23: Encender la vibración

4.4 Memoria EEPROM

Como la memoria es un componente de retención o almacenamiento de datos, y el sentido de la comunicación puede ser de "entrada" o de "salida" dependiendo de las necesidades en cada momento, la trataremos de manera independiente al resto de componentes ya vistos sin enmarcarla en uno de estos contextos.

La memoria no volátil retendrá la información almacenada incluso si no recibe corriente eléctrica constantemente, como es el caso de la memoria ROM. En nuestro caso, es necesaria una memoria no volátil dado que es indispensable disponer de algún lugar en el que almacenar cierta información que no queremos perder entre una conexión y la siguiente.

El Wiimote incluye una memoria interna para almacenar ciertos datos de usuario, así como la configuración de funcionamiento del mando en la conexión Bluetooth actual.

La memoria incorporada en el Wiimote es una EEPROM M24128-BWP fabricada por ST Microelectronics.

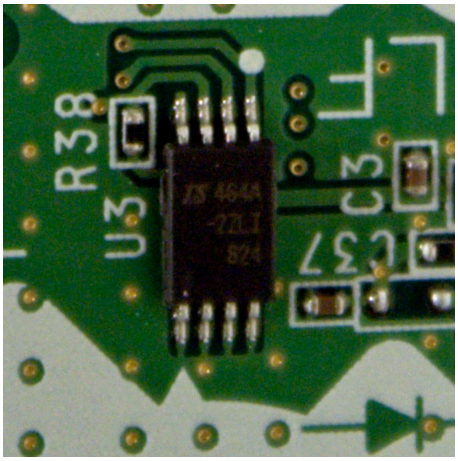


Figura 4.11: Memoria flash del Wiimote

La M24128-BWP es una memoria ROM de 128 kbit (16kB) eléctricamente programable que cumple con I^2C en el bus de comunicaciones serie.

- Características:
  - Soporta el modo de bus I^2C:
    - 400 kHz modo rápido
    - 100 kHz modo estándar
  - Tensión de alimentación única:
    - 2.5 V to 5.5 V (M24C32-W)
    - 1.8 V to 5.5 V (M24C32-R)
    - 1.7 V to 5.5 V (M24C32-F)
  - entrada de control de escritura,
  - escritura a nivel de byte y página,
  - modos de lectura secuencial y aleatorio,
  - ciclo de programación autoprogramado,
  - incremento automático de direcciones,
  - protección mejorada contra estática,
  - duración de más de 1 millón de ciclos de escritura,
  - persistencia de los datos de más de 40 años.

La parte de memoria de usuario se emplea para almacenar constantes de calibración, así como los datos de los Mii. Además, como veremos más adelante, muchos periféricos del Wiimote disponen de registros que son accesibles a través de una porción del espacio de direcciones.

Tanto la memoria incorporada como los registros de los periféricos son accesibles usando los mismos informes, simplemente marcando un flag para seleccionar entre cada una de ellas.

Si en un Wiimote virgen (vendido por separado para que no haya realizado ninguna conexión con una Wii) realizamos un volcado de memoria, la estructura que veremos será la siguiente:

0000:	A1	AA	8B	99	AE	9E	78	30	A7	74	D3	A1	AA	8B	99	AE
0010:	9E	78	30	A7	74	D3	82	82	82	15	9C	9C	9E	38	40	3E
0020:	82	82	82	15	9C	9C	9E	38	40	3E	00	00	00	00	00	00
0030:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Cuadro 4.24: Primeros bytes de la memoria flash

que son básicamente dos secuencias, cada una repetida dos veces:

0000:	A1	AA	8B	99	AE	9E	78	30	A7	74	D3
000B:	A1	AA	8B	99	AE	9E	78	30	A7	74	D3
0016:	82	82	82	15	9C	9C	9E	38	40	3E	
0020:	82	82	82	15	9C	9C	9E	38	40	3E	

Cuadro 4.25: Secuencias iniciales de la memoria

Estos datos se corresponden con secuencias de calibrado del Wiimote, y suponemos que están repetidos para poder hacer cambios en la primera secuencia y conservar el valor de fábrica en caso de querer resetear el Wiimote.

Los cuatro primeros bytes de la segunda secuencia, 0x82 0x82 0x82 0x15 ,contienen la calibración de los offsets para los acelerómetros (recordemos que cada eje es descrito con 10 bits). Los tres siguientes, 0x9C 0x9C 0x9E, contienen la fuerza de la gravedad para cada uno de los ejes.

El resto de bytes de estas secuencias tienen una función desconocida.

Igualmente, existe otra porción de memoria que no se encuentra a cero y cuya función es desconocida:

16D0:	00	00	00	FF	11	EE	00	00	33	CC	44	BB	00	00	66	99
16E0:	77	88	00	00	2B	01	E8	13	00	00	00	00	00	00	00	00
16F0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Cuadro 4.26: Dtos desconocidos

Por tanto, la memoria puede ser dividida en las siguientes secciones:

Comienzo	Fin	Longitud	Valor inicial	Uso
0x0000	0x0029	0x002A	Ver arriba	Valores de calibración
0x002A	0x0FC9	0x0FA0	Vacía	Datos de usuario
0x0FCA	0x12B9	0x02f0	Vacía	Datos Mii 1
0x12BA	0x15A9	0x02f0	Vacía	Datos Mii 2
0x15AA	0x16CF	0x0126	Vacía	Desconocido
0x16D0	0x16FF	0x0030	Ver arriba	Desconocido

Cuadro 4.27: Rangos de memoria

Las direcciones de memoria de usuario son de tan solo dos bytes, de tal manera que si intentamos acceder a la posición 0x01 0x00 0x00, el Wiimote comprenderá que queremos acceder a 0x00 0x00.

4.4.1 Registros de control

Como hemos comentado, el Wiimote posee una serie de registros en memoria. Estos registros se corresponden con los diferentes periféricos que contiene el mando o que pueden ser conectados a él, incluyendo el altavoz, el puerto de expansión, Wii Motion Plus y la cámara de infrarrojos.

Para acceder a estas posiciones es necesario, como vimos en el apartado anterior, habilitar un flag en el primer byte de la carga útil del paquete que envía el servidor (0x04), ya que de otra manera, accederemos a la memoria de datos sobreescribiendo el contenido de ésta.

La manera de acceder a los registros es bastante sencilla. Ahora las direcciones no serán de 2 bytes como las de la memoria de usuario, sino de 3 bytes.

Dada una dirección de registro de control, el primer byte indicará a qué periférico queremos acceder, mientras que los dos bytes siguientes indican la posición del registro a la que queremos acceder dentro de ese periférico.

Por tanto, los registros pueden ser divididos en las siguientes secciones:

Start	End	Use
0xA20000	0xA20009	Speaker settings
0xA40000	0xA400FF	Extension Controller settings and data
0xA60000	0xA600FF	Wii Motion Plus settings and data
0xB00000	0xB00033	IR Camera settings

Cuadro 4.28: Registros de control

4.4.2 Lectura

Para leer datos, es necesario enviar una petición al canal 0x17:

0x52 0x17 0xMM 0xFF 0xFF 0xFF 0xSS 0xSS

Cuadro 4.29: Petición de lectura

Como vemos, la carga útil es de 6 bytes, donde MM seleccionará el espacio de direcciones, 0x00 para memoria de usuario ó 0x04 para registros; 0xFF 0xFF 0xFF será la dirección a la que queremos acceder; y 0xSS 0xSS el tamaño en bytes que queremos leer.

Los datos leídos se retornan a través de un paquete por el canal 0x21:

0xA1 0x21 0xBB 0xBB 0xSE 0xFF 0xFF 0xDD [...] 0xDD

Cuadro 4.30: Datos leídos

En este canal la carga útil es de 21 bytes, donde 0xBB 0xBB es el estado de los botones en el Wiimote; 0xSE contiene el tamaño S en bytes menos uno para el paquete actual y el flag de error E, cuyo valor será: 0 si no hay error, 7 si se intenta leer de un área de solo escritura y 8 si se intenta leer de direcciones inexistentes; 0xFF 0xFF es la dirección de memoria absoluta, ya que sabremos si hemos accedido a memoria de usuarios o registro y a qué registro al haber efectuado la petición, para el primer byte de datos que se nos devuelve; y 0xDD [...] 0xDD serán los datos, rellenados con ceros si no ocupan todas las posiciones.

Si se piden más de 16 bytes de datos, se recibirán en múltiples paquetes en los que se irá incrementando la dirección 0xFF 0xFF en 16 bytes, correspondientes a los datos proporcionados en el paquete anterior.

4.4.3 Escritura

Para escribir datos, es necesario enviar una petición al canal 0x16:

0x52 0x16 0xMM 0xFF 0xFF 0xFF 0xSS 0xDD [...] 0xDD

Cuadro 4.31: Petición de escritura

En este canal la carga útil es de 21 bytes, donde el significado de los bytes es el mismo que en la petición de lectura, exceptuando que el tamaño no podrá exceder de los 16 bytes de datos que adjuntamos en el mismo paquete, 0xDD [...] 0xDD, y que deberán ser rellenados con ceros al final en caso de ser menos de los mencionados 16 bytes.

CAPÍTULO 5

Wiinux

Una vez realizado el estudio de ingeniería inversa para conocer en profundidad el funcionamiento del Wiimote, podemos proceder a desarrollar nuestra librería para controlar este periférico Bluetooth.

En una primera instancia, el proyecto comenzó con la implementación de una biblioteca del Wiimote para Python, pero una vez que se vio el potencial que poseía, y las posibilidades que esto abría, se optó por ampliar dicha biblioteca y hacerla más accesible a cualquier tipo de público. Esta ampliación ha alargado el tiempo de desarrollo.

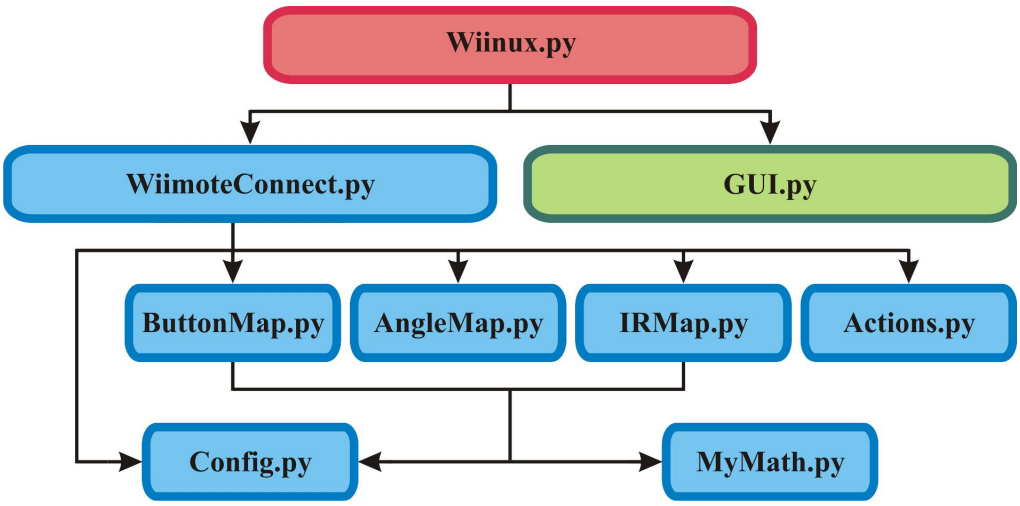


Figura 5.1: Esquema de la biblioteca

Para nuestra aplicación necesitaremos por tanto tres hilos, uno que se encargue de la conexión con el Wiimote y de mantener las comunicaciones pertinentes con el mismo; otro con la interfaz gráfica de usuario; y otro que haga las tareas de control entre ambos hilos para establecer una serie de criterios y ayudar en la comunicación entre ambos. Además, fabricaremos una barra de infrarrojos propia.

5.1 Librerías

Las librerías que se han implementado contienen todas las clases y métodos necesarios para conseguir una conexión fiable con el Wiimote y la interacción de una persona que lo controle con el equipo.

El diseño de la biblioteca se realizó de forma modular, es decir, cada clase realiza una función muy específica de manera que es bastante sencillo explicar el funcionamiento de la misma con un diagrama de bloques, y por tanto, encontrar cada parte de forma meramente intuitiva. Esto conlleva una serie de ventajas:

- facilita la tarea del programador,
- permite controlar el acoplamiento,
- facilita la reutilización del código,
- permite el desarrollo independiente y paralelo de aplicaciones,
- permite el desarrollo de versiones mejoradas de la biblioteca sin afectar al código que las utiliza,
- permite distribuir implementaciones empaquetadas.

5.1.1 WiimoteConnect.py

Esta clase proporciona la capacidad de connectar y acceder a un Wiimote que se encuentre en las proximidades. Para ello, se ayuda de los siguientes módulos de Python:

```
import bluetooth
import time
from threading import Thread
```

PyBluez [PyBa] es un módulo de extensión de Python que ofrece a los desarrolladores usar los dispositivos Bluetooth de manera sencilla mediante una API [PyBb] que contiene funciones de uso común.

Time es el módulo que proporciona los métodos que permiten manipular valores de tiempo. Nos será útil para introducir pequeñas esperas entre líneas de código que permitan una comunicación coherente entre los dispositivos Bluetooth.

Threading usa el paquete Thread para proveer al programador de interfaces de alto nivel. Contiene mecanismos para prevenir el interbloqueo de las secciones críticas, y por supuesto, también contiene métodos wait y notify, semáforos, etc. Esto nos permitirá la ejecución y convivencia de los tres hilos (conexión, GUI y control) bajo el mismo proceso.

Además, obtiene soporte de las siguientes clases que definiremos posteriormente en esta misma sección:

```
from Config import *
from ButtonMap import *
from AngleMap import *
from IRMap import *
from Actions import *
```

Los métodos de los que dispone son los siguientes:

def __init__(self, parent_obj, geometry):

El objeto Thread nunca se utiliza directamente, sino sobrescribiendo el método __init__() o la función run(), a través del interfaz threading.Thread.

En nuestro caso sobreescribimos este método para valernos de un constructor que guarde un puntero a la instancia padre (parent_obj, el hilo de control), guarde el valor de la resolución de pantalla actual (geometry) e inicialice a "False" una variable con el estado de finalización de este hilo.

Finalmente, y dado que sobreescribimos el método __init__(), debemos llamar a Thread.__init__()

def run(self):

El método run() hace que la instancia WiimoteConnect() comience a ejecutar.

En nuestro caso simplemente llama al método connect().

def connect(self):

El método connect() es el que contiene el código que ejecutará realmente y de manera contínua en este hilo.

En primer lugar, hace una búsqueda de los dispositivos bluetooth cercanos, para posteriormente discriminar por el nombre objetivo "Nintendo RVL-CNT-01", correspondiente con el "Service Name" que devuelve cualquier Wiimote.

```
target_name = "Nintendo RVL-CNT-01"
target_address = None
nearby_devices = bluetooth.discover_devices()
for bdaddr in nearby_devices:
    if target_name == bluetooth.lookup_name( bdaddr ):
        target_address = bdaddr
        break
```

Si no se encuentra un dispositivo con estas características se eleva una notificación al hilo de control mediante *notify_no_wiimote()*; si se encuentra un Wiimote, se procede a la conexión Bluetooth.

Para crear esta conexión y el correspondiente vínculo con el Wiimote, se crean los sockets Bluetooth correspondientes a los canales lógico y de datos:

```
wiimote_control_chanel = wiimote['control_chanel']
wiimote_data_chanel = wiimote['data_chanel']
control_socket = bluetooth.BluetoothSocket(bluetooth.L2CAP)
data_socket = bluetooth.BluetoothSocket(bluetooth.L2CAP)
control_socket.connect((target_address, wiimote_control_chanel))
data_socket.connect((target_address, wiimote_data_chanel))
packet_size = 1024
```

Para conectar con el Wiimote es necesario seguir los siguientes pasos:

- notificar que queremos activar la cámara,
- activar el reloj interno de 24 MHz de la cámara,
- activar la cámara,
- escribir en los registros la activación de la cámara,
- escribir en los registros el Sensitivity Block 1 (configuración de sensibilidad),
- escribir en los registros el Sensitivity Block 2 (configuración de sensibilidad),
- escribir en los registros el Mode Number (modo de operación),
- reescribir en los registros la activación de la cámara.

Esto en código se traduce como:

```
control_socket.send(control['act_IR']);      time.sleep(0.01)
control_socket.send(control['act_CLK']);     time.sleep(0.01)
control_socket.send(control['act_CAM']);     time.sleep(0.01)
control_socket.send(control['act_REG']);     time.sleep(0.01)
control_socket.send(control['act_SB1']);     time.sleep(0.01)
control_socket.send(control['act_SB2']);     time.sleep(0.01)
control_socket.send(control['act_MN']);      time.sleep(0.01)
control_socket.send(control['act_REG']);     time.sleep(0.01)
```

En el código real, hemos intercalado paquetes de activación de los LEDs 1 y 2 para que parpadeen los cuatro mientras se realiza la configuración del Wiimote, además de notificaciones de progreso *notify_progress()* de la conexión al hilo de control.

Una vez finalizada la configuración del Wiimote, se le hace vibrar para advertir al usuario que se ha finalizado con éxito y se notifica mediante *notify_end_conection()* al hilo padre para que también sea conocedor del éxito de la conexión.

Llegados a este punto, entramos en un bucle que solamente se romperá si no se reciben paquetes del Wiimote o si se ha notificado en la interfaz gráfica finalizar la conexión. Las tareas del bucle son:

- recibir paquetes en los que, dada su posición, si son:
  - 3 o 4 llamará a *button_map()*
  - 5, 6 o 7 llamará a *angle_map()*
  - mayor que 7, llamará a *ir_map()*
- llamar a *pressing()* para realizar la acción de los botones,
- cada 2 paquetes (para evitar retrasos en el movimiento):
  - llamar a *positioning(width, height)* para posicionar el puntero en el lugar correcto,
- cada 50 paquetes (para evitar conflictos con las X):
  - llamar a *moving()* para traducir los movimientos del Wiimote en caso de haberlos,
  - informar al hilo padre del valor de los acelerómetros y la cámara infrarroja mediante *notify_sensors()*

Además, es capaz de distinguir distintas excepciones en la ejecución del código y notificarlo al hilo padre. Las excepciones capturadas y sus notificaciones correspondientes son:

- RuntimeError, e:
  - *notify_wiimote_interruption()*
- bluetooth.btcommon.BluetoothError, e:
  - *notify_bluetooth_error()*

def stop(self):

El método stop() hace que la instancia WiimoteConnect() pare su ejecución. Para ello pone a "True" la variable con el estado de finalización de este hilo, con lo que la ejecución saldrá del bucle principal y por tanto, el hilo terminará su existencia al llegar al final del código.

5.1.2 MyMath.py

Esta clase proporciona las operaciones matemáticas que no están incluídas en la librería estándar y que son necesarias para el funcionamiento de nuestro programa.

def Denary2Binary(n):

El método Denary2Binary() convierte un número "n" en base 10 a base 2. Para ello divide el número inicial en base 10 de manera sucesiva por 2 hasta obtener un cociente menor que 2, concatenando el último cociente y los restos en forma ascendente, de manera que se obtiene el número en base 2.

$8_{10} =?_2$

$8 \,\%2 = 4 + \mathbf{0}$
$4 \,\%2 = 2 + \mathbf{0}$
$2 \,\%2 = \mathbf{1} + \mathbf{0}$

$8_{10} = 1000_2$

Cuadro 5.1: Conversión de base 10 a base 2

def CompleteOctet(bin):

El método CompleteOctet() se encarga de rellenar el binario "bin" con tantos ceros a la izquierda como sea necesario para completar un byte. Para el ejemplo anterior:

CompleteOctet(1000) → 00001000

Cuadro 5.2: Completado del octeto

5.1.3 Config.py

Esta clase contiene la información correspondiente a todas las variables internas del programa relacionadas con el Wiimote, organizadas en una serie de cómodos diccionarios:

wiimote

El diccionario *wiimote* contiene la información correspondiente a los canales Bluetooth HID, es decir, el número de canal de multiplexación para los canales lógico (L2CAP PSM:17) y de datos (PSM:19)

mapped

El diccionario *mapped* contiene el valor correspondiente, de ratón o de teclado, que se introdujo en la interfaz gráfica para cada botón del Wiimote.

control

El diccionario *control* contiene el valor en hexadecimal de los bytes de los distintos paquetes de datos que son necesarios en la comunicación con el Wiimote.

A continuación trataremos los distintos paquetes para examinar su contenido. Son los siguientes, todos ellos con sentido de salida (0x52):

- led1:
  paquete para activar el LED del jugador 1.

Sentido	Canal	Carga útil
0x52	0x11	0x10

Cuadro 5.3: Encender el LED del jugador 1

  Se envía al canal 0x11 la carga útil correspondiente a la máscara del jugador 1 que, como recordaremos del capítulo anterior, es 0x10.

- led2:
  paquete para activar el LED del jugador 2.

Sentido	Canal	Carga útil
0x52	0x11	0x20

Cuadro 5.4: Encender el LED del jugador 2

  Como en el caso anterior, ahora se envía al canal 0x11 la carga útil correspondiente a la máscara del jugador 2 que es 0x20.

- rumble:
  paquete para encender el vibrador del Wiimote.

Sentido	Canal	Carga útil
0x52	0x11	0x21

Cuadro 5.5: Activar el vibrador

  Recordaremos del capítulo anterior que se podía encender el vibrador enviando la carga útil 0x01 a diferentes canales, lo cual era aconsejable para evitar conflictos con los LEDs, pero nosotros enviaremos al canal 0x11 la carga útil correspondiente a la máscara del jugador 2 y del vibrador que es 0x21.

  Hemos optado por este canal (0x11) ya que la que queremos realizar es una operación bastante sencilla y así nos evitamos el envío de dos paquetes diferentes.

- act_IR:
  paquete para la notificación de inicio de la secuencia de activación de la cámara infrarroja:

Sentido	Canal	Carga útil
0x52	0x12	0x00 0x33

Cuadro 5.6: Notificación de activación IR

  Se envía al canal 0x12 (notificaciones) la carga útil correspondiente a la activación de la cámara infrarroja en modo extendido, es decir, 0x00 0x33.

  Hemos elegido este modo de operación porque el manejo de la información contendia en los paquetes es más sencillo con respecto al modo básico. Además, nos proporciona información suficiente para nuestra aplicación, ofreciendo el modo completo demasiada información que deberíamos desechar.

- act_CLK:
  paquete para activar el reloj interno que controla la cámara infrarroja. Como ya hemos comentado, éste es un reloj de 24 MHz.

Sentido	Canal	Carga útil
0x52	0x13	0x04

Cuadro 5.7: Activar el reloj interno

  Se envía al canal 0x13 la carga útil correspondiente a una máscara específica que habilita el funcionamiento del reloj interno de la cámara.

  Como podemos observar, solamente se envía un bit a modo de "flag"; de este hecho podemos suponer que probablemente ese bit se corresponda con un "enable" de la electrónica.

- act_CAM:
  paquete para activar la cámara infrarroja en si.

Sentido	Canal	Carga útil
0x52	0x1a	0x04

Cuadro 5.8: Activar la cámara

  Se envía al canal 0x1a la carga útil correspondiente a una máscara específica que habilita la cámara infrarroja.

  Como ocurría en el caso anterior, es muy probable que también sea un "enable" de la electrónica.

- act_REG:
  paquete para escribir en los registros la orden de activación de la cámara

Sentido	Canal	Memoria	Dirección	Tamaño	Datos (16 bytes)
0x52	0x16	0x04	0xb0 0x00 0x30	0x01	0x08 0x00 [...] 0x00

Cuadro 5.9: Orden de activación IR

Para habilitar el uso de la cámara infrarroja, es necesario escribir en los registros dicha activación. De esta manera habremos de enviar al canal 0x16 (escritura de datos), seleccionando los registros de control con 0x04, la dirección en la que queremos escribir que será 0xb0 0x00 0x30, el tamaño que queremos escribir 0x01 y el dato que queremos escribir 0x08. El resto del paquete se rellena con 0x00.

- act_SB1:
  paquete para escribir en los registros la configuración 1 de sensibilidad.

Sentido	Canal	Memoria	Dirección	Tamaño	Datos (16 bytes)
0x52	0x16	0x04	0xb0 0x00 0x00	0x09	0x00 [...] 0x90 0x00 0xC0 [...] 0x00

Cuadro 5.10: Sensitivity Block 1

El primer bloque de sensibilidad tiene la misma estructura que act_REG, en la que enviamos al canal 0x16, seleccionando los registros de control con 0x04 la dirección en la que queremos escribir, en este caso en 0xb0 0x00 0x00 los datos de tamaño 0x09 y contenido 0x00 0x00 0x00 0x00 0x00 0x00 0x90 0x00 0xC0.

Para el primer bloque de sensibilidad, el tamaño máximo permitido es 0x09.

- act_SB2:
  paquete para escribir en los registros la configuración 2 de sensibilidad.

Sentido	Canal	Memoria	Dirección	Tamaño	Datos (16 bytes)
0x52	0x16	0x04	0xb0 0x00 0x1a	0x01	0x40 0x00 [...] 0x00

Cuadro 5.11: Sensitivity Block 2

El segundo bloque de sensibilidad también tiene la misma estructura, en la que enviamos al canal 0x16, seleccionando los registros de control con 0x04 la dirección en la que queremos escribir, en este caso en 0xb0 0x00 0x1a los datos de tamaño 0x01 y contenido 0x04.

Para el segundo bloque de sensibilidad, el tamaño máximo permitido es 0x02.

Los datos a escribir en los bloques de sensibilidad son propuestas de desarrolladores que están trabajando en proyectos similares. En los foros de desarrollo de aplicaciones para la Wii podemos encontrar diferentes configuraciones para lograr distintas sensibilidades del Wiimote.

- act_MN:
  paquete para escribir en los registros el modo de funcionamiento de la cámara infrarroja.

Sentido	Canal	Memoria	Dirección	Tamaño	Datos (16 bytes)
0x52	0x16	0x04	0xb0 0x00 0x33	0x01	0x03 0x00 [...] 0x00

Cuadro 5.12: Orden de modo de funcionamiento

Se envía al canal de datos 0x16, seleccionando los registros de control con 0x04 la dirección correspondiente al modo de funcionamiento 0xb0 0x00 0x33, y en ella escribimos el que emplearemos, en nuestro caso el 0x03.

Las razones de emplear este modo de funcionamiento han sido descritas anteriormente.

buttons

El diccionario *buttons* contiene el valor actual y anterior de los botones, es decir, si se encuentran o encontraban presionados o liberados. Estas condiciones son precisas para poder detectar cambios, comparando el estado actual con el que le precedía.

angles

El diccionario *angles* contiene el valor actual del ángulo con respecto al eje 'a' (que puede ser X, Y o Z) y el incremento de dicho ángulo 'Aa' respecto al estado anterior. Estos valores son necesarios para conocer la posición del Wiimote en el espacio y la velocidad de desplazamiento en éste, respectivamente.

moves

El diccionario *moves* contiene la información acerca de los movimientos bruscos del Wiimote. Se comportan como un booleano que indica si ha habido o no movimiento brusco a izquierda o derecha.

ir

El diccionario *ir* contiene las coordenadas X e Y de los dos primeros puntos detectados por el Wiimote (de los cuatro posibles), además de la sensibilidad de cada uno de ellos. También contiene los valores XT e YT, que se corresponden con el punto imaginario calculado a partir de los reales para posicionar el puntero en la pantalla.

5.1.4 ButtonMap.py

Esta clase proporciona la capacidad de mapear el estado de los botones (presionado o liberado) a partir de un byte que se le haya proporcionado, indicándole la posición de éste dentro del paquete Bluetooth. Para ello obtiene soporte de las siguientes clases:

```
from config import buttons
from MyMath import *
```

Como vemos, solo necesita acceder al diccionario *buttons* y obtener apoyo matemático. El método del que dispone es:

def button_map(pos_byte, byte):

El método button_map() convierte a byte en binario el valor de "byte", completando el octeto. Una vez en binario, guarda en "buttons['a']['state']" el valor anterior del botón "a" contenido en "buttons['a']['pressed']" y traduce la información del byte, bit a bit, para obtener la información actual de los botones, que sobreescribirá en "buttons['a']['pressed']".

Para realizar la traducción de bit a botón sigue la siguiente relación:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
3				+	↑	↓	→	←
4	Home			-	A	B	1	2

Cuadro 5.13: Traducción de los botones

Nótese que, para un humano, la lectura de un número binario se realiza desde la derecha, pero si accedemos a ella en un programa, las posiciones aumentan hacia la derecha. Es decir, si queremos leer el bit 2 habremos de acceder a la posición 5 de nuestro array.

5.1.5 AngleMap.py

Esta clase proporciona la capacidad de mapear el estado de los acelerómetros y calcular el ángulo que forma con cada eje a partir de un byte que se le haya pasado, indicándole la posición de éste dentro del paquete Bluetooth. Para ello obtiene soporte de las siguientes clases:

```
from Config import angles
from Config import moves
```

Como vemos, solamente necesita acceder a los diccionarios *angles* y *moves*.

El método del que dispone es:

def angle_map(pos_byte, byte):

El método angle_map() convierte a byte en binario el valor de "byte", y calcula el ángulo que forma la posición del Wiimote respecto a ese eje guardándolo en "angles['a']", donde "a" puede ser X, Y o Z. También calcula el incremento del ángulo en cada momento, lo cual se corresponderá realmente con una fuerza ejercida sobre el Wiimote.

Para realizar la traducción de byte a ángulo tiene en cuenta lo siguiente:

Byte	**Eje**
5	X
6	Y
7	Z

Cuadro 5.14: Correspondencia entre byte y eje

El Wiimote devuelve, para un eje, valores en el intervalo [100, 150], que se corresponderían con los asociados a los ángulos comprendidos en el intervalo [-90º, 90º]. Así pues, hemos de aplicar una sencilla operación que relaciona el valor proporcionado por el sensor y el ángulo real. Matemáticamente esto es:

$$X^{\circ} = 3{,}6(valor - 125)$$

Lo que en código se traduce como:

```
angles['X'] = (ord(byte)-125)*3.6
```

Además, para el eje X comprueba si el movimiento ha sido brusco, por si ha de incrementar las variables "moves['RM']" o "moves['LM']" que informan de estos hechos. Para ello, simplemente calcula la diferencia entre el valor del ángulo contenido en la variable "angles['a']" y el valor actual, para después compararlo con un valor umbral, determinado por la sensibilidad humana.

```
angles['Ax'] = angles['X'] - (ord(byte)-125)*3.6
if angles['Ax'] ¡-60:
    moves['RM'] = moves['RM'] + 1
if angles['Ax'] ¿60:
    moves['LM'] = moves['LM'] + 1
```

Tras una serie de pruebas, hemos planteado el umbral 60, que parece responder adecuadamente tras una fase de aprendizaje, ya que cuesta un poco hacerse a este movimiento por las fuerzas de acción y reacción.

5.1.6 IRMap.py

Esta clase proporciona la capacidad de mapear los puntos vistos por la cámara infrarroja y calcular un punto medio imaginario a partir de un byte que se le haya pasado, indicándole la posición de éste dentro del paquete Bluetooth. Para ello obtiene soporte de las siguientes clases:

```
from Config import ir
from MyMath import *
```

Como vemos, solo necesita acceder al diccionario *ir* y obtener apoyo matemático.

El método del que dispone es:

def ir_map(pos_byte, byte):

El método ir_map() traduce la información de los dos primeros puntos vistos por la cámara infrarroja. Para ello, obtiene los bits correspondientes de cada byte y reconstruye el valor de cada coordenada formada por 20 bits (10 para el eje X y 10 para el eje Y) además del valor de tamaño.

Para realizar la traducción de bits a coordenadas, sigue la siguiente relación:

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
8	X_1<7:0>							
9	Y_1<7:0>							
10	Y_1<9:8>		X_1<9:8>		S_1<3:0>			
11	X_2<7:0>							
12	Y_2<7:0>							
13	Y_2<9:8>		X_2<9:8>		S_2<3:0>			

Cuadro 5.15: Traducción de las coordenadas

Para obtener el valor de una coordenada, traduce a entero el valor del byte completo de ese eje para, a continuación, sumarle el valor entero (como si estuvieran seguidos de ceros) de los bits más significativos. Esto en código, para X_1, es:

```
if pos_byte == 8:
    ir['X1'] = ord(byte)
if pos_byte == 10:
    octet = CompleteOctet(Denary2Binary(ord(byte)))
    ir['X1'] = 512*int(octet[2])+256*int(octet[3])+ir['X1']
```

Recordemos que aunque nosotros definamos las posiciones de los bits de derecha a izquierda, en nuestro programa la posición del array aumenta hacia la derecha.

Una vez que tenemos las coordenadas de los dos puntos, podemos pasar a calcular el punto medio imaginario, que en nuestro caso es simplemente la media aritmética:

```
ir['XT'] = (ir['X1']+ir['X2'])/2
ir['YT'] = (ir['Y1']+ir['Y2'])/2
```

De esta manera, queda determinado el punto imaginario que nos será útil para posicionar el puntero del ratón.

5.1.7 Actions.py

Esta clase proporciona la capacidad de traducir los datos que hemos mapeado del Wiimote en acciones concretas. Para ello, se ayuda del siguiente módulo de Python:

```
import os
```

Os es el módulo que proporciona el acceso a routinas del Sistema Operativo para MAC, NT o Posix, dependiendo del sistema en el que estemos ejecutando. En concreto, en esta clase, nos interesará *os.system()* que ejecuta el mandato especificado como argumento en una subshell.

Además, obtiene soporte de las siguientes clases:

```
from Config import buttons
from Config import moves
from Config import mapped
from Config import ir
```

Como podemos observar, necesita acceder a todas las variables correspondientes al estado de botones y sensores para llevar a cabo las distintas acciones.

Los métodos de los que dispone son:

def moving():

El método moving() traduce la información del diccionario *moves* en pulsaciones de teclas. Para ello, realiza una comparación entre el valor entero contenido en moves['RM'] y moves['LM'] para determinar en qué sentido se ha realizado la sacudida del Wiimote.

Es intuitivo pensar que, en un movimiento brusco en el que tendremos fuerzas de acción y reacción, la ida sea realizada con más ímpetu que la vuelta, con lo que el valor de la variable asociada al sentido de la ida tenderá a ser mayor que el de la vuelta y en consiguiente se procederá a realizar una acción relacionada con dicho movimiento.

Por ejemplo, si la variable hacia la derecha es mayor que hacia la izquierda, realizaremos la acción contenida en mapped['RM'], que se corresponde con la asociada a la derecha:

```
if moves['RM'] ¿moves['LM']:
    os.system("xte 'key " + mapped['RM'] + " ' ")
    moves['RM'] = 0
    moves['LM'] = 0
```

Como vemos, *os.system()* ejecutará en una subshell el mandato:

```
xte 'key mapped['RM']'
```

que se corresponderá con el presionado y la liberación de la tecla correspondiente al contenido de mapped['RM'].

def pressing():

El método pressing() traduce la información del diccionario *buttons* en pulsaciones de teclas o clicks de ratón. Para ello, realiza una comparación entre el valor anterior y el valor actual del estado de los botones contenido en buttons['a']['state'] y buttons['a']['pressed'] respectivamente (para el botón "a" correspondiente), y así poder determinar si ha ocurrido algún cambio en el estado de los botones del Wiimote.

Para realizar la traducción de estados, e identificar si ha de realizar alguna acción, sigue la siguiente relación:

buttons['a']['pressed']	**buttons['a']['state']**	**Acción**
True	False	Presionado
False	True	Liberado

Cuadro 5.16: Traducción de estados de los botones

En este caso, *os.system()* ejecutará en una subshell el mandato correspondiente con alguna de las siguientes combinaciones:

```
xte 'keydown mapped['a']'
xte 'keyup mapped['a']'

xte 'mousedown mapped['a']'
xte 'mouseup mapped['a']'
```

dependiendo de si la traducción se corresponde con una pulsación de tecla o un click de ratón.

def positioning(width, height):

El método positioning() traduce la información del diccionario *ir* en movimientos del puntero de ratón sobre la pantalla. Anteriormente, hemos comentado que este método se llama cada dos paquetes, ya que si no es de esta manera se hará visible un efecto de "seguimiento" indeseado, como si el ratón siguiera la mano, acumulando latencia en cada paquete.

Si recordamos, la cámara del Wiimote tiene una resolución de 1024x768 pixels por lo que el primer reto que se nos plantea es realizar una correspondencia entre dicha resolución y la resolución real de la pantalla a la que estamos apuntando.

Además, el valor que tenemos mapeado es el de los puntos luminosos sobre unos ejes X e Y que no plantean una correspondencia directa con lo que queremos plasmar sobre la pantalla, sino que nos proporcionan un eje X invertido con respecto a la información que queremos representar.

Por otra parte, los puntos de la barra de infrarrojos se encuentran separados unos 400 pixels, y han de encontrarse siempre dentro del campo de visión de la cámara para evitar efectos indeseados como puedan ser saltos del puntero.

El escenario que tenemos es algo como lo que sigue:

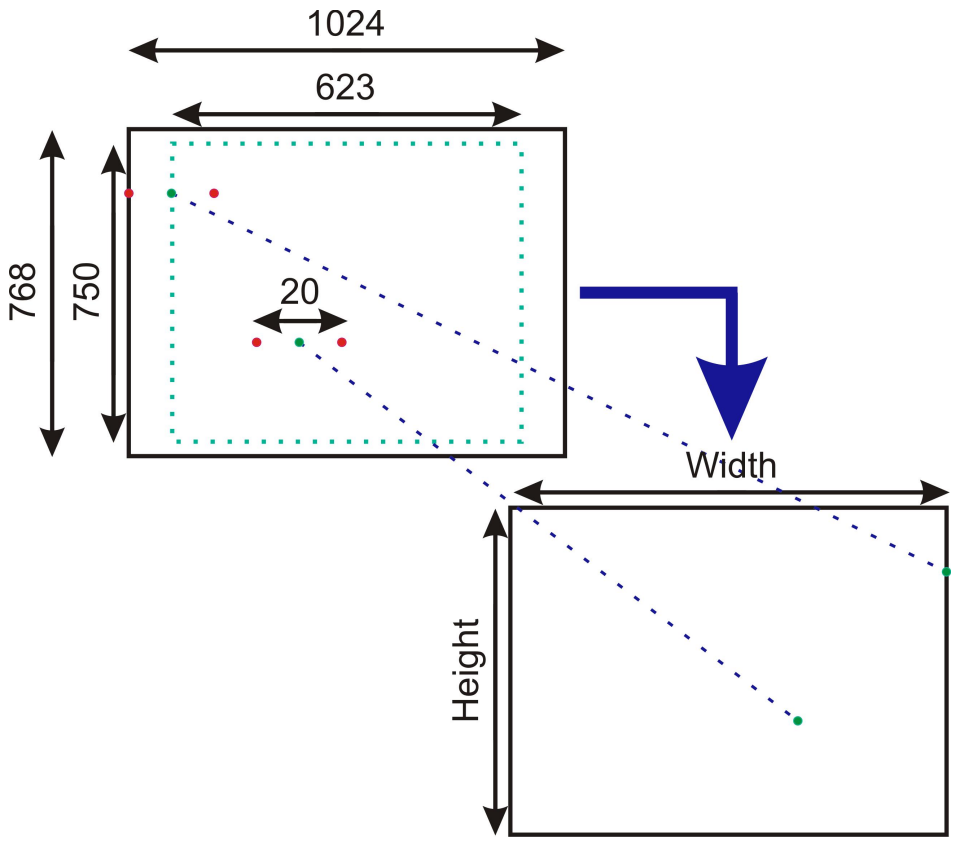


Figura 5.2: Correspondencia entre IR y resolución

Por lo tanto, necesitamos una relación entre los valores "raw" proporcionados por el Wiimote y los valores que queremos representar realmente sobre la pantalla de nuestro ordenador, es decir, unas nuevas coordenadas X e Y que cumplan los requisitos de nuestra resolución y el eje de coordenadas del usuario.

Matemáticamente, podemos plantear las siguientes ecuaciones:

para el eje X:

$$X = width - \left(\frac{width}{623}(ir['XT'] - 200)\right) \tag{5.1}$$

para el eje Y:

$$Y = \frac{height}{750} ir['YT'] \tag{5.2}$$

Como vemos, para el eje X debemos tener en cuenta la separación entre los dos puntos luminosos de los extremos de la barra para que, cuando desaparezca uno por los extremos de la cámara, no aparezcan los efectos indeseados comentados. Así, al evitar que el puntero salte, la experiencia de usuario es mejor, eliminando la sensación de no saber donde apuntar.

Sin embargo, en el eje Y no debemos tener esta precaución, ya que ambos puntos se encontrarán siempre a la misma altura y desaparecerán de la visión del Wiimote de manera simultánea, evitando efectos indeseados por definición.

Este cálculo posee una pequeña debilidad, suponiendo que el Wiimote se encontrará siempre posicionado sobre el plano XY y no permitiendo movimientos que estén relacionados con la rotación del mismo.

Finalmente, *os.system()* ejecutará en una subshell el mandato correspondiente para posicionar el puntero sobre la pantalla:

```
xte 'mousemove X Y'
```

Al encontrarse anidada en un "if", esta llamada al sistema se realiza de manera demasiado lenta respecto a la persistencia visual, y da la impresión de mover el ratón en pequeños intervalos, pero es lo suficientemente buena como para no perder la interacción con el usuario.

En este sentido, es preferible tener la impresión de "saltos" en el movimiento del puntero, a que la respuesta del mismo se encuentre retardada por intentar hacer todas las llamadas por cada paquete de datos.

5.2 GUI.py

Esta clase genera la Interfaz Gráfica de Usuario que permite interaccionar en tiempo real con el programa en ejecución. Para ello se ayuda de los siguientes módulos:

```
import sys
import time

from threading import Thread

import pygtk
pygtk.require('2.0')
import gtk
gtk.gdk.threads_init()

import matplotlib
matplotlib.use('GTK')
from matplotlib.figure import Figure
from matplotlib.axes import Subplot
from matplotlib.backends.backend_gtk import FigureCanvasGTK
```

Algunos de ellos, como time o threading, ya los conocemos de las librerías. Veamos las tareas que desempeñan sys, PyGTK y matplotlib.

Sys es el módulo que proporciona los métodos que permiten el acceso a objetos del intérprete y a funciones que interactúan con dichos objetos.

PyGTK [Fin06] es un módulo de extensión de Python que ofrece a los programadores el desarrollo de interfaces gráficas usando las bibliotecas gráficas multiplataforma GTK mediante una API [PyG09] sencilla. Como vemos, es necesario indicar la versión mínima de PyGTK que queremos emplear.

Además, como es lógico, es necesario importar GTK, ya que al desarrollar interfaces gráficas con PyGTK se hace uso directo de las librerías gráficas. Por otra parte, como nuestra aplicación será multihilo, necesitamos inicializar el motor de threads de GTK, puesto que si no, se pensará que nuestra aplicación es monohilo (aunque se haya importado el módulo threading y la clase herede de Thread).

Matplotlib [Huna] es un módulo de extensión de Python que se apoya en la extensión numérica de Python NumPy y ofrece a los desarrolladores la capacidad de generar gráficos a partir de datos contenidos en listas o arrays. Además, tiene la peculiaridad de estar diseñada para proporcionar una API [Hunb] que recuerde a la de MATLAB.

Como vemos es necesario avisar de que Matplotlib empleará GTK para realizar las gráficas, así como importar las clases concretas que necesitaremos en nuestro diseño: FigureCanvasGTK (área de dibujo GTK), Figure y Subplot.

Los métodos de los que dispone son los siguientes:

def __init__(self, parent_obj):

En nuestro caso, hemos de sobreescribir el método constructor de la interfaz gráfica, donde guardaremos un puntero a la instancia padre (parent_obj, el hilo de control), inicializaremos a "False" una variable que nos servirá para saber cuando estamos dibujando las gráficas de los sensores y obtendremos la resolución de la pantalla, que más adelante nos será de utilidad.

Además, contiene lo más importante, el código que construirá la ventana principal de nuestro programa, que quedará de la manera mostrada en la siguiente figura:

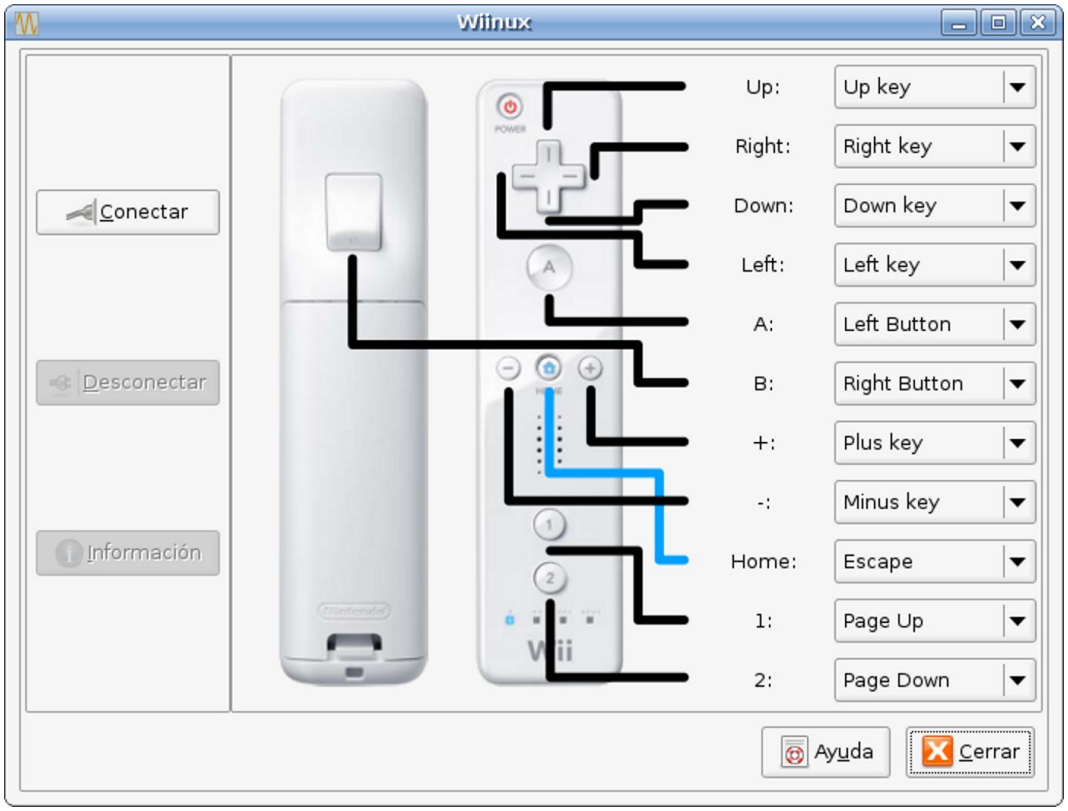


Figura 5.3: Wiinux GUI - Wiimote desconectado

Como vemos, la interfaz gráfica de usuario (GUI) es sencilla e intuitiva. A la izquierda podemos observar los botones destinados a la conexión con el Wiimote (si no hay una conexión activa), a la desconexión del Wiimote y el acceso a la información de los sensores (estos dos habilitados si existe una conexión activa). A la derecha observamos una serie de desplegables que nos permiten configurar los botones del Wiimote a nuestro gusto antes de la conexión. En la parte inferior se encuentran el botón de ayuda y el destinado a cerrar la aplicación.

def open_connect(self, widget, data=None):

El método open_connect() llama al método *open_connect_aux()* para, seguidamente, obtener las opciones seleccionadas en los desplegables con el método *read_options()* y notificar al hilo de control dicha selección y la resolución de pantalla mediante el método del hilo de control *notify_start_connection()*.

def open_connect_aux(self):

El método open_connect_aux() realiza una serie de operaciones con la GUI. En primer lugar deshabilita todos los botones para que no se pueda modificar nada mientras dura la conexión, como se muestra en la siguiente figura:

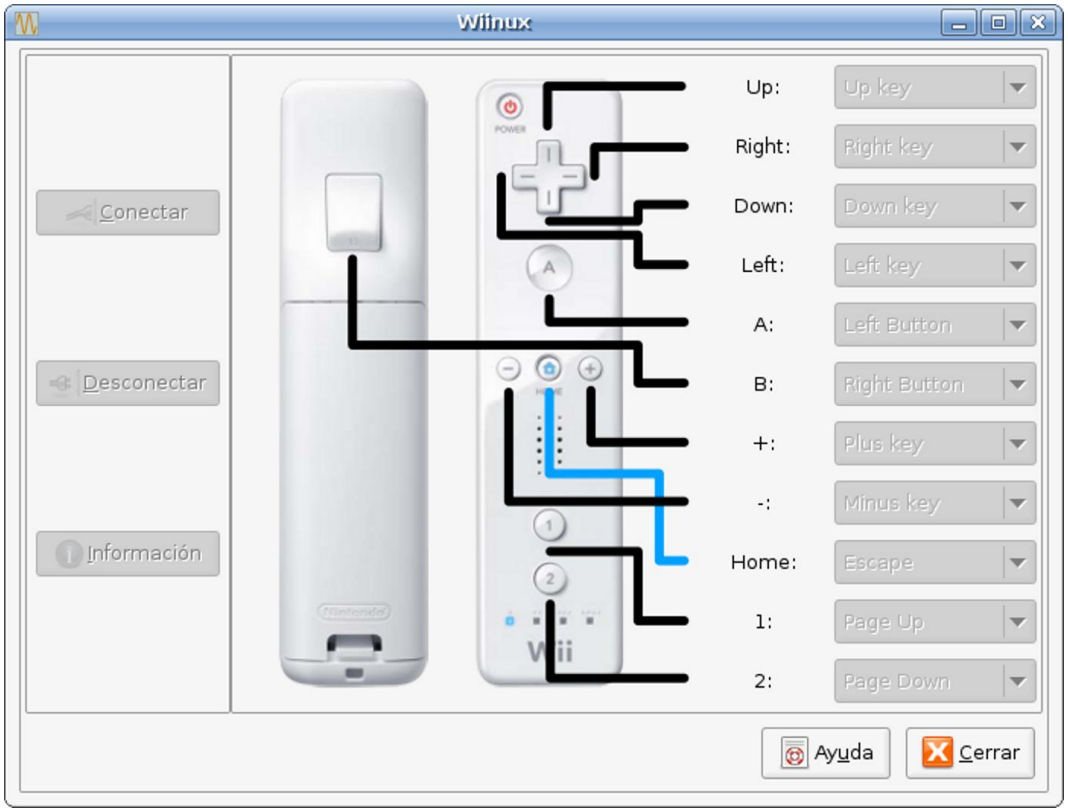


Figura 5.4: Wiinux GUI - Wiimote conectando

Además, crea y muestra una nueva barra de progreso como la siguiente:

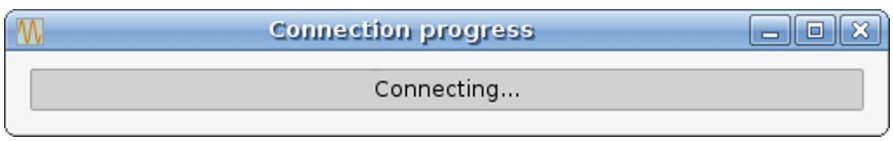


Figura 5.5: Wiinux GUI - Connection progress

def read_options(self):

El método read_options() se encarga de crear una lista con el valor seleccionado en los desplegables de la ventana principal, que leerá gracias al método *get_active_text()*.

def set_progress(self, progress):

El método set_progress() está pensado para que se pueda establecer de manera externa la progresión de la barra de progreso anterior. Para ello, este método habrá de ser llamado con un valor comprendido en el intervalo [0, 1], que determinará la porción de barra rellena.

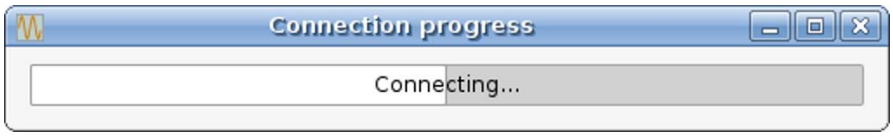


Figura 5.6: Wiinux GUI - Connection progress 0.5

def end_connection_dialog(self):

El método end_connection_dialog() está pensado para que se pueda notificar al hilo de la interfaz gráfica el éxito de la conexión con el Wiimote para, de esta manera, realizar una serie de operaciones que se traducirán en el cierre de la barra de progreso y la habilitación de los botones de desconexión e información. De esta manera, la interfaz gráfica quedará como el la siguiente figura:

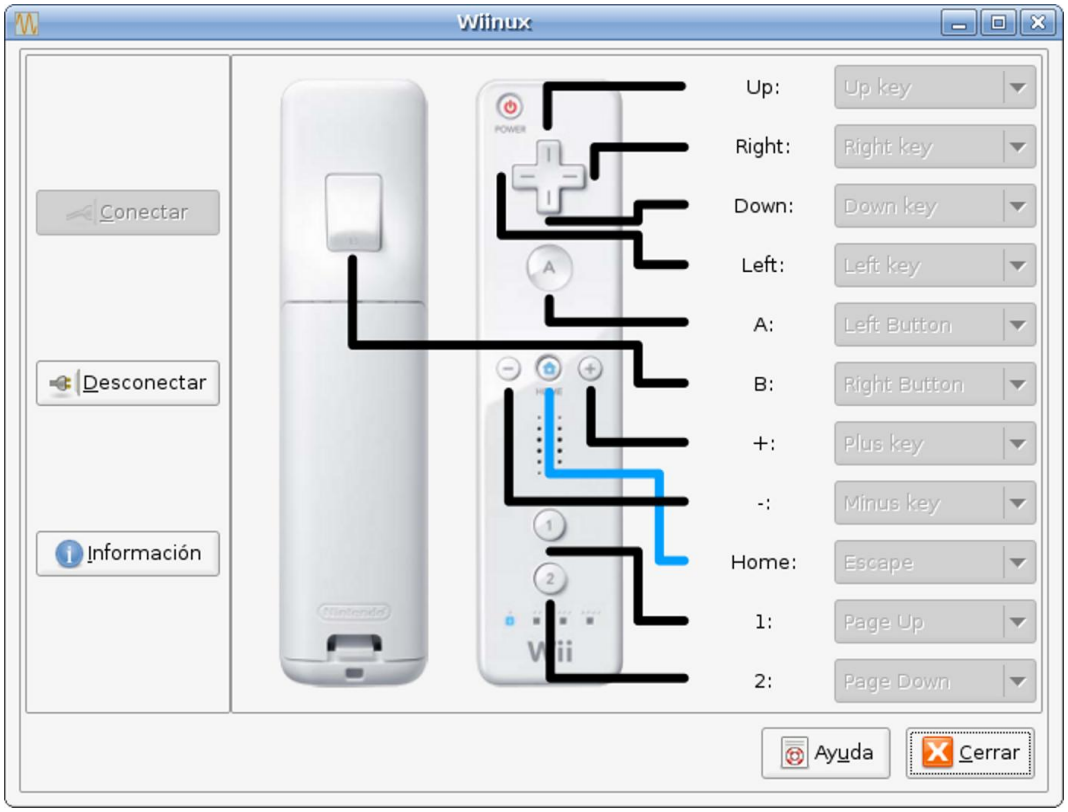


Figura 5.7: Wiinux GUI - Wiimote conectado

Estas operaciones se encuentran separadas debido a que no queremos que se habiliten los botones hasta que se cumpla necesariamente la conexión con el Wiimote.

def open_disconnect(self, widget, data=None):

El método open_disconnect() llama al método *open_disconnect_aux()* para, seguidamente, notificar al hilo de control que se desea interrumpir la conexión con el Wiimote mediante el método *notify_finish_connection()*.

def open_disconnect_aux(self):

El método open_disconnect_aux() realiza una serie de operaciones con la GUI para devolverla a su estado inicial. Para ello, deshabilita los botones de desconexión e información de los sensores y a continuación habilita todos los botones inactivos.

def open_plots(self, widget, data=None):

El método open_plots() crea una nueva ventana a la que añade los elementos necesarios para poder realizar las representaciones gráficas de los sensores. Además, pone a "True"la variable que nos sirve para saber cuando estamos dibujando esta ventana, y que evitará que se llame al método *replot()* si ésta no se encuentra abierta.

Para ello, crea un nuevo objeto "FigureCanvasGTK" al que asociará otro objeto del tipo "Figure"; a su vez, esta figura contendrá cuatro "subplot" que se corresponderán con la información del eje X, del eje Y, del eje Z y del sensor infrarrojo respectivamente.

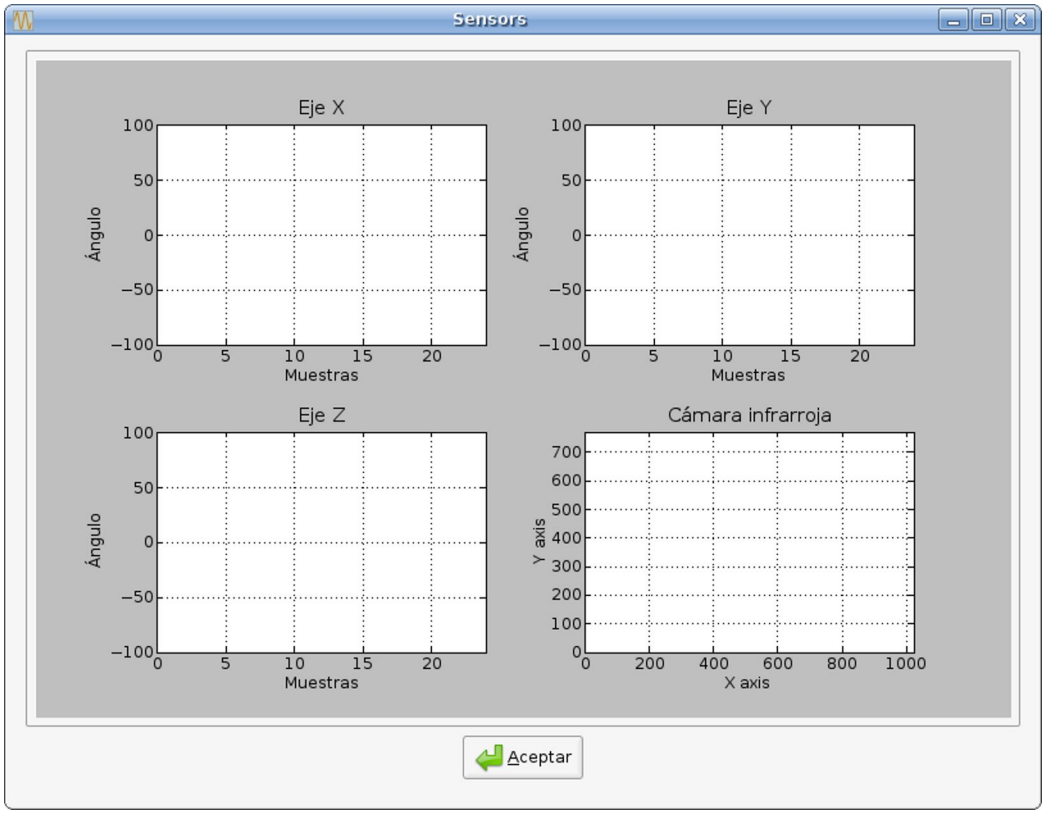


Figura 5.8: Wiinux GUI - Creación Información de los Sensores

Si nos fijamos en los ejes del acelerómetro, podemos observar que el valor mostrado es el correspondiente al ángulo que forman con cada eje. Por esta razón los ejes de estas gráficas se encuentran en un intervalo [-90, 90] para el eje de ordenadas.

Si nos fijamos en la información del sensor infrarrojo, podemos observar que el valor mostrado es el correspondiente al punto visto en la propia cámara del Wiimote. Por esta razón los ejes de estas gráficas se encuentran en un intervalo [0, 1024]x[0, 768] que, si recordamos, es la resolución que ofrece la cámara incorporada.

def replot(self, x_angle, y_angle, z_angle, x_pos, y_pos):

El método replot() se encarga de recoger la información de los argumentos y mantener un buffer con las últimas 25 muestras para cada eje del acelerómetro.

Con estos datos, redibuja cada "subplot" con el contenido del buffer correspondiente para cada eje y con el valor de la posición del punto infrarrojo en este momento, para finalmente refrescar la ventana de tal manera que acabamos obteniendo lo siguiente:

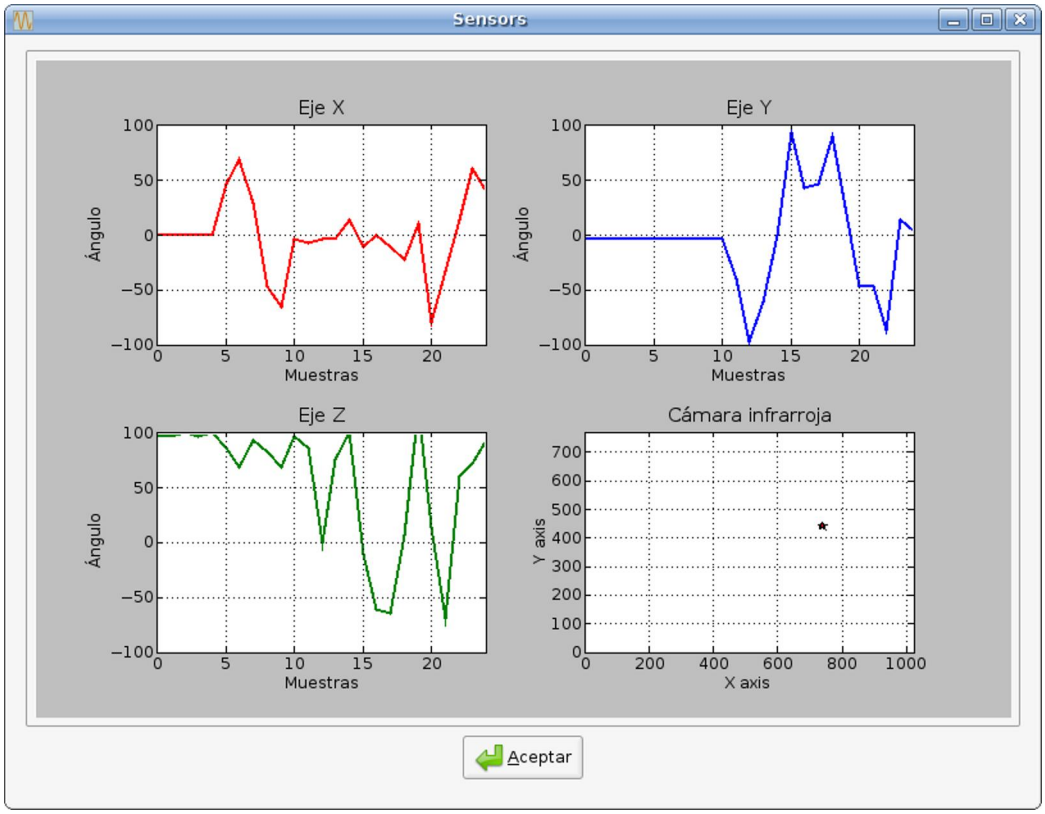


Figura 5.9: Wiinux GUI - Información de los Sensores

En estas gráficas vemos que:

- Para el eje X los movimientos positivos se relacionan con el posicionamiento de la cara superior del Wiimote hacia la derecha; los movimientos negativos se relacionan con el posicionamiento hacia la izquierda.

- Para el eje Y los movimientos positivos se relacionan con el posicionamiento de la cara superior del Wiimote hacia adelante; los movimientos negativos se relacionan con el posicionamiento hacia atrás.
- Para el eje Z los movimientos positivos se relacionan con el posicionamiento de la cara superior del Wiimote hacia abajo; los movimientos negativos se relacionan con el posicionamiento hacia arriba.
- El punto infrarrojo muestra la posición vista en la cámara del Wiimote.

def close_plots(self, widget, data=None):

El método close_plots() se encarga de volver a asignar el valor "False" a la variable que indica la existencia de esta ventana y proceder a su destrucción.

def open_help(self, widget, data=None):

El método open_help crea una nueva ventana con información de interés. Cuando se pulsa el botón de ayuda de la ventana principal aparecerá lo siguiente:

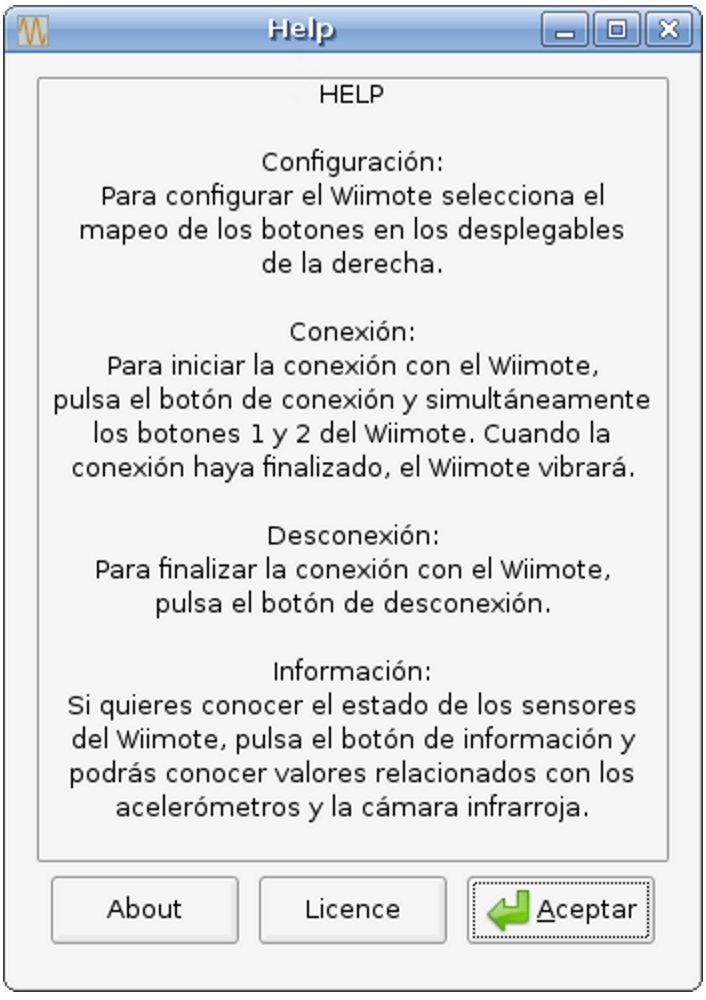


Figura 5.10: Wiinux GUI - Help

Como vemos, ofrece información básica acerca de como configurar el Wiimote y para poner a funcionar el programa en unos sencillos pasos, así como la manera de interactuar con el mismo.

Además, en la parte inferior izquierda y central de esta ventana podemos identificar los botones "About" y "Licence" respectivamente, que nos conducirán a sendas ventanas que explicamos a continuación.

def close_help(self, widget, data=None):

El método close_help() se encarga de cerrar la ventana de ayuda.

def open_about(self, widget, data=None):

El método open_about() crea una nueva ventana con algo de contenido acerca del proyecto. Cuando se pulsa el botón "About" de la ventana de ayuda aparecerá lo siguiente:

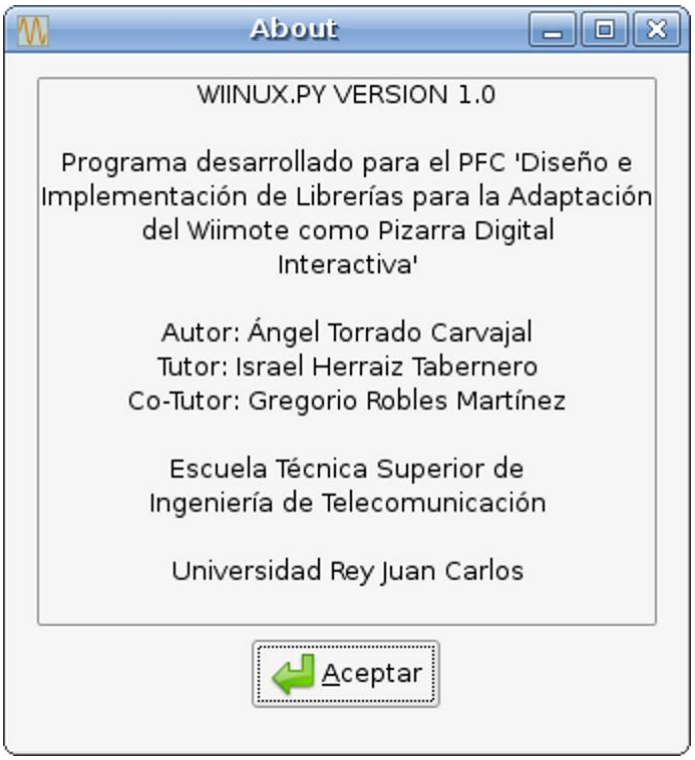


Figura 5.11: Wiinux GUI - About

Como vemos, ofrece información acerca de los términos de registro del Proyecto Fin de Carrera en la Universidad Rey Juan Carlos.

def close_about(self, widget, data=None):

El método close_about() se encarga de cerrar la ventana de about.

def open_licence(self, widget, data=None):

El método open_licence() crea una nueva ventana con información acerca de la licencia. Cuando se pulsa el botón "Licence" de la ventana de ayuda aparecerá lo siguiente:

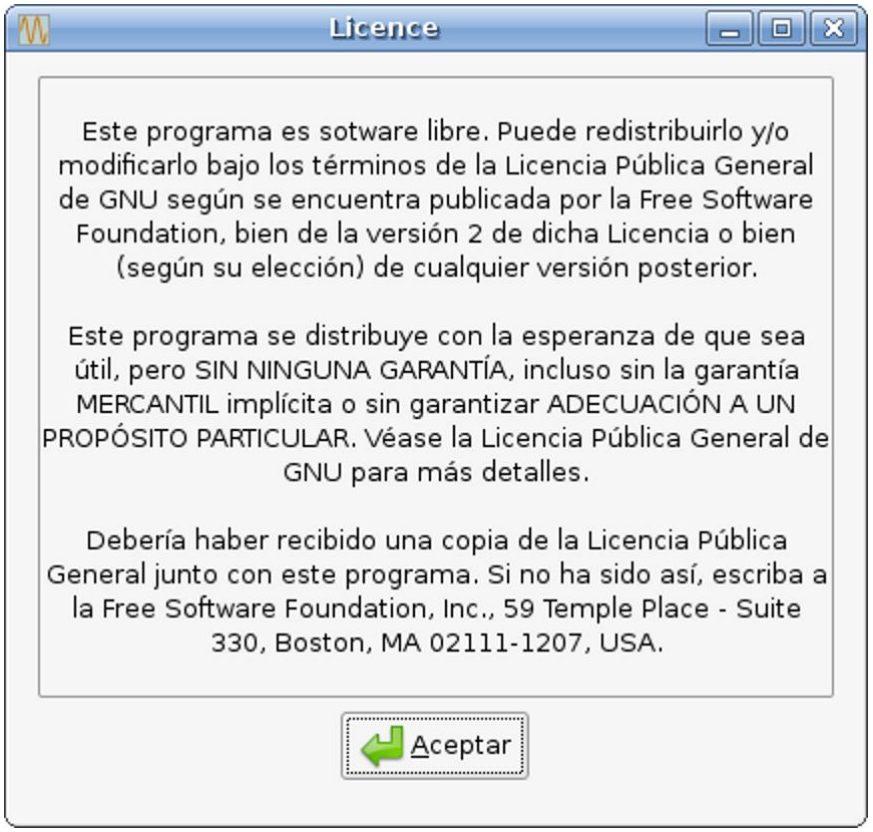


Figura 5.12: Wiinux GUI - Licence

Como vemos, ofrece información acerca de los términos de licencia que explicaremos posteriormente.

def close_licence(self, widget, data=None):

El método close_licence() se encarga de cerrar la ventana de licencia.

def exit_program(self, widget, data=None):

El método exit_program() notificará al hilo de control que finalice cualquier conexión activa, en caso de encontrarse alguna en progreso, esperará a que el hilo de control realice las tareas necesarias y acabará el programa completo mediante *sys.exit()*.

def complete_keys(self, combo):

El método complete_keys() se encargará de rellenar el contenido del desplegable correspondiente con el listado de teclas disponibles para su elección.

def complete_mouse(self, combo):

El método complete_mouse() se encargará de rellenar el contenido del desplegable correspondiente con el listado de botones de ratón disponibles para su elección.

def get_active_text(self, combobox):

El método get_active_text() devuelve el texto activo de un desplegable. Python no posee un método para realizar esta operación, pero si cuenta con otros para operar con desplegables, de esta manera podemos seguir los siguientes pasos:

- obtener el modelo del contenido del desplegable con *get_model()*,
- obtener la posición activa del desplegable con *get_active()*,
- devolver el texto activo con *model[active][0]*.

def no_wiimote_dialog(self):

El método no_wiimote_dialog() crea un nuevo diálogo de error mostrando que no se ha podido encontrar un Wiimote en las proximidades al pulsar el botón "Conectar" de la ventana principal.

Figura 5.13: Wiinux GUI - Could not find Wiimote nearby

def wiimote_interruption_dialog(self):

El método wiimote_interruption_dialog() crea un nuevo diálogo de error mostrando que se ha perdido la conexión con el Wiimote, ya sea porque se ha quedado sin baterías o porque se haya presionado el botón "POWER" de éste.

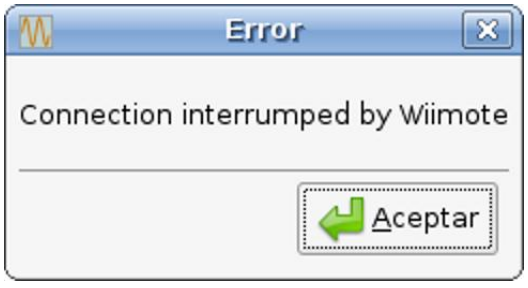


Figura 5.14: Wiinux GUI - Connection interrumped by Wiimote

def bluetooth_error_dialog(self):

El método bluetooth_error_dialog() crea un nuevo diálogo de error mostrando que no se puede acceder al dispositivo Bluetooth, ya sea porque no ha sido conectado o porque su funcionamiento sea incorrecto.

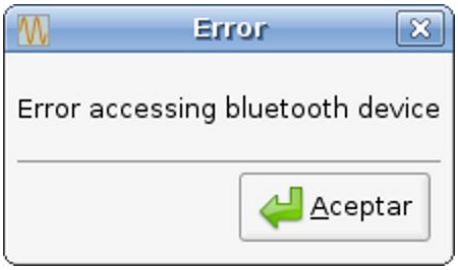


Figura 5.15: Wiinux GUI - Error Accessing Bluetooth Device

def close_dialog(self, widget, data=None):

El método close_dialog() se encarga de cerrar el diálogo de error activo.

5.3 Wiinux.py

Como ya hemos comentado con anterioridad, para conseguir la ejecución concurrente de la GUI y la conexión con el Wiimote, surgía la necesidad de disponer de un hilo de control que manejase el flujo de ejecución y las comunicaciones pertinentes entre ambos hilos.

De esta manera, Wiinux.py existe para llevar a cabo esta tarea obteniendo soporte de las siguientes clases:

```
from GUI import *
from Wiinux.WiimoteConnect import *
from Wiinux.Config import mapped
```

Como podemos observar, necesita acceder a GUI y WiimoteConnect, ya que tendrá que acceder a los constructores y métodos de sus hilos hijos. Además, en uno de sus propios métodos escribirá en el diccionario *mapped* del fichero de configuración Config.py la información leída de la GUI.

Además, por su naturaleza, posee un método especial que se encarga de llamar al constructor de su clase y ejecutarla:

```
if __name__ == '__main__':
app = App()
app.run()
```

Siendo "App()" por tanto la clase principal.

Los métodos de los que dispone son:

def __init__(self):

El método __init__() es el constructor de la clase *App()* y se encarga de crear vacías las variables que contendrán después las instancias correspondientes a la GUI y al hilo de conexión.

def run(self):

El método run() es el encargado de lanzar la GUI, para ello sigue los siguientes pasos:

- se crea una nueva instancia de *GUI(self)*,
- se llama al método start() que heredó de la clase threading.Thread para que se ejecute el código de la GUI,
- se llama a *gtk.main()*, método que proporcionará acceso a la instancia global de GTK.

def notify_start_connection(self, options, geometry):

Cuando *GUI()* llama a este método, se llevan a cabo las siguientes acciones:

- se crea una nueva instancia de *WiimoteConnect(self, geometry)* con el parámetro "geometry" que le ha sido pasado como argumento,
- se llama al método propio *write_options(options)* con el parámetro "options" que le ha sido pasado como argumento,
- se llama al método *start()* del hilo de conexión para que comience su ejecución.

def notify_progress(self, progress):

Cuando *WiimoteConnect()* llama al método notify_progress() informando del estado de progreso de la conexión en curso, se establece un puente con el método *set_progress()* de la GUI, que reflejará este estado en la barra de progreso que verá el usuario.

def notify_end_connection(self):

Cuando *WiimoteConnect()* llama al método notify_end_connection() informando de que la conexión con el Wiimote se ha completado con éxito, se establece un puente con el método *end_connection_dialog()* de la GUI, que cerrará la barra de progeso y habilitará los botones de desconexión e información.

def notify_finish_connection(self):

Cuando *GUI()* llama al método notify_finish_connection() informando de que el usuario desea terminar la conexión con el Wiimote al haber presionado el botón de desconexión de la ventana principal, se para el hilo *WiimoteConnect()* mediante su método *stop()*.

def write_options(self, options):

El método write_options() se encarga de traducir los valores contenidos en la lista *options* a valores de teclas que sea capaz de entender el mandato Xte y los escribe en el diccionario *mapped* del fichero de configuración Config.py para que se mantengan fijos durante esta conexión.

def notify_sensors(self, x_angle, y_angle, z_angle, x_pos, y_pos):

Cuando *WiimoteConnect()* llama al método notify_sensors() informando del estado actual del acelerómetro y de la cámara de infrarrojos del Wiimote, se establece un puente con el método *replot()* de la GUI, que se encargará de representar estos datos de manera conveniente en la ventana de información de sensores si se encuentra abierta.

def notify_no_wiimote(self):

Cuando *WiimoteConnect()* llama al método notify_no_wiimote() como resultado de una excepción al no encontrar un Wiimote en las proximidades, se llevan a cabo una serie de operaciones en la GUI:

- se cierra la barra de progreso con *end_connection_dialog()*,
- se habilitan y deshabilitan los botones correspondientes para devolver la GUI a su estado inicial con *open_disconnect_aux()*,
- se abre el diálogo de error correspondiente con *no_wiimote_dialog()*.

def notify_wiimote_interruption(self):

Cuando *WiimoteConnect()* llama al método notify_wiimote_interruption() como resultado de una excepción al interrumpirse la conexión con el Wiimote, se llevan a cabo una serie de operaciones en la GUI:

- se habilitan y deshabilitan los botones correspondientes para devolver la GUI a su estado inicial con *open_disconnect_aux()*,
- se abre el diálogo de error correspondiente con *wiimote_interruption_dialog()*.

def notify_bluetooth_error(self):

Cuando *WiimoteConnect()* llama al método notify_bluetooth_error() como resultado de una excepción al no encontrar el dispositivo Bluetooth, se llevan a cabo una serie de operaciones en la GUI:

- se cierra la barra de progreso con *end_connection_dialog()*,
- se habilitan y deshabilitan los botones correspondientes para devolver la GUI a su estado inicial con *open_disconnect_aux()*,
- se abre el diálogo de error correspondiente con *bluetooth_error_dialog()*.

5.4 Barra casera

Se ha comprobado que barras "caseras" han resultado efectivas con menos LEDs mientras que la intensidad de emisión sea la suficiente. Para comprobar este hecho, se ha procedido a fabricar una barra de infrarrojos USB.

De esta manera, en primer lugar hemos elegido los LEDs adecuados para el montaje. En nuestro caso hemos conseguido LEDs infrarrojos de alta luminosidad, para así evitar problemas derivados de interferencias con otras fuentes de luz o por la distancia del Wiimote a la barra.

Estos LEDs se polarizan a unos 1.3 V, demandando una corriente del orden de los 100 mA. Dada esta situación, comprobamos que el USB es idóneo para la tarea, ya que proporciona 5 V y una corriente de hasta 500 mA, con lo que podemos proponer el siguiente montaje:

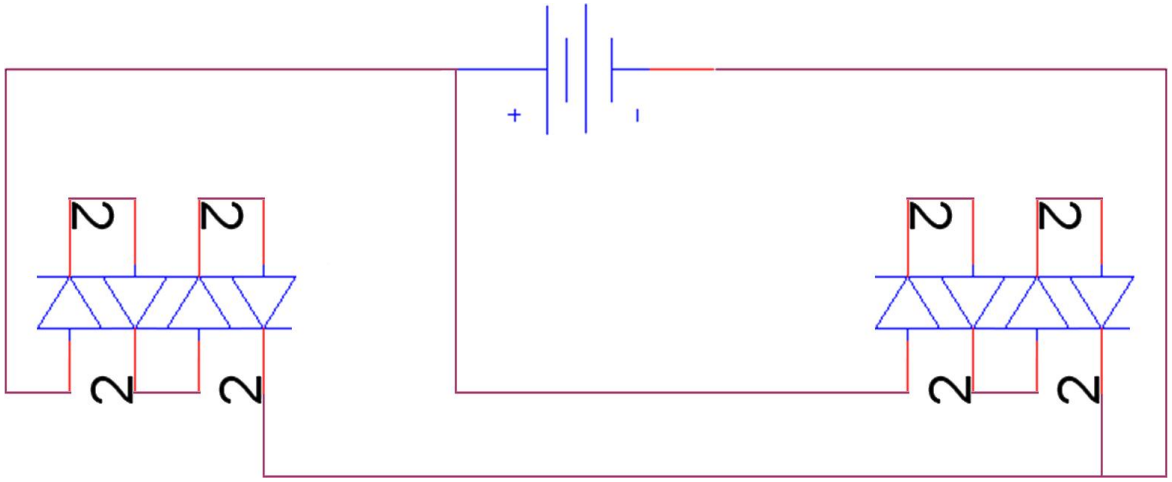


Figura 5.16: Esquemático de la Barra de Infrarrojos Casera

Con esta configuración, cada LED se encontrará polarizado a 1.25 V y le será suministrada la corriente suficiente para que luzcan correctamente.

De la teoría pasamos a la práctica. Ahora debemos realizar el montaje de cada extremo de la barra, donde posicionaremos los cuatro LEDs de manera similar a la barra original:

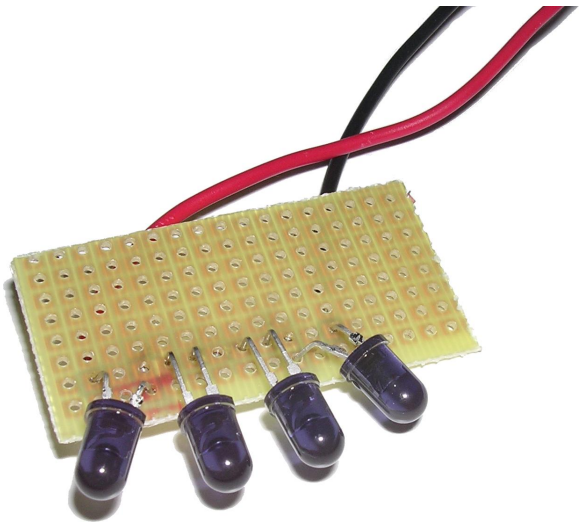


Figura 5.17: Montaje de los Infrarrojos

Teniendo en cuenta que cada LED tiene un ángulo de apertura de 50º, este posicionamiento hará que los puntos de luz infrarroja sean visibles desde un ángulo total de más de 100º, obteniendo un diagrama de radiación similar al siguiente:

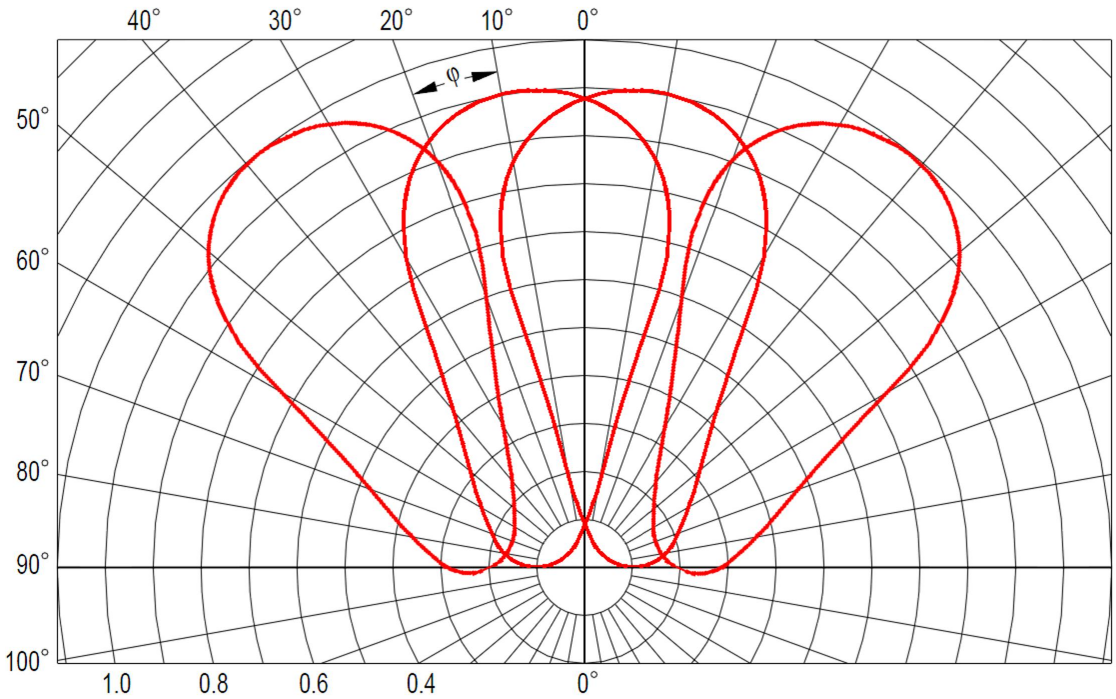


Figura 5.18: Diagrama de Radiación de la Barra Casera

Este hecho permitirá el movimiento libre delante de la pantalla de la persona que esté interactuando con el programa.

Finalmente, uniremos los dos extremos de la barra al cable USB, dejando libres los cables de datos, y fijaremos todo a una base rígida, obteniendo nuestra barra infrarroja casera:

Figura 5.19: Barra de Infrarrojos Casera

Ahora tenemos todo lo necesario para poder usar nuestro proyecto en cualquier ordenador.

5.5 Publicación del código

Siguiendo con la filosofía de código abierto para nuestra aplicación, hemos de realizar dos pasos para completar su "ciclo de vida" y ponerlo a disposición de nuestros "clientes" que serán usuarios y desarrolladores interesados en el tema.

5.5.1 Licencia

En primer lugar, antes de publicarlo, habremos de anexionar una licencia a nuestro código para evitar comportamientos indeseados y pueda seguir su nuevo desarrollo. Para ello hemos elegido una Licencia Pública General GNU.

La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License [Inc] (GNU GPL), es una licencia creada por la Free Software Foundation en 1989 (su primera versión), y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

La licencia GPL, al ser un documento que cede ciertos derechos al usuario, asume la forma de un contrato, por lo que usualmente se la denomina contrato de licencia o acuerdo de licencia. Como contrato, la GPL debe cumplir los requisitos legales de formación contractual en cada jurisdicción.

De esta manera, nuestro programa puede ser redistribuído y/o modificado bajo los términos de la Licencia Pública General de GNU según se encuentra publicada por la Free Software Foundation, bien de la versión 2 de dicha Licencia o bien de cualquier versión posterior. (Ver apéndices para más detalles).

5.5.2 Sitio

Una vez conformado el paquete "código-licencia" habremos de disponernos a pasar al siguiente punto, liberar el código en algún repositorio que permita el acceso al mismo de esos espectros de población que hemos comentado.

Existen varios repositorios donde poder realizar esta acción. Quizá el más conocido sea SourceForge.net [Sof], una central de desarrollos de software que controla y gestiona varios proyectos de software libre y actúa como un repositorio de código fuente. SourceForge.net se hospeda por VA Software y corre una versión del software SourceForge, software de colaboración para la administración de desarrollos.

CAPÍTULO 6

Conclusiones

Una vez descritas las soluciones adoptadas para resolver los problemas planteados al principio de esta memoria, en este último capítulo se quieren recapitular todas las experiencias que se han vivido a lo largo del desarrollo del proyecto, así como estudiar otros aspectos importantes de impacto en su área y proponer mejoras y líneas futuras que se pueden plantear a raíz de las aplicaciones desarrolladas.

6.1 Conclusiones

Ante las especificaciones iniciales del proyecto podría pensarse que se trataría de una tarea rápida y de poca envergadura, sin embargo, la realidad ha demostrado ser todo lo contrario. La obtención de información a partir del producto comercial como caja negra se convirtió en una tarea un tanto ardua que en ocasiones parecía prácticamente imposible.

Tareas como la conexión Bluetooth o el análisis de los botones, resultaron ser más sencillas de lo que pensaba, bastando una serie de pruebas y errores (más unas cuantas lecturas sobre el funcionamiento del protocolo de comunicaciones y la API de Python) para solucionar los problemas que se iban presentando.

Sin embargo, otras tareas como las relacionadas con los sensores acelerómetros y sobre todo con la cámara infrarroja, consiguieron ponerme al límite y hacerme pensar que era una misión imposible. El hecho de no saber cómo obtener valores coherentes que describieran la actividad de los mismos, llegaron a enervarme situando al proyecto en estado de crisis.

Sin embargo, esto me hizo aprender lo que la investigación conlleva en muchas ocasiones, y es que para llegar al éxito es probable tener que fracasar varias veces. No obstante, de cada uno de esos fracasos se obtiene parte de conocimiento que te hace progresar.

Además, en este punto se manifestó otra característica común del ingeniero, y por norma general de cualquier ser humano, y es que bajo grandes presiones se suelen conseguir grandes resultados.

Esto me hizo recordar y tener presentes durante el resto del proyecto ciertas palabras que había leído con anterioridad del célebre físico Albert Einstein:

"No pretendamos que las cosas cambien, si siempre hacemos lo mismo. La crisis es la mejor bendición que puede sucederle a personas y países, porque la crisis trae progresos. La creatividad nace de la angustia, como el día nace de la noche oscura. Es en la crisis que nace la inventiva, los descubrimientos y las grandes estrategias. Quien supera la crisis, se supera a sí mismo sin quedar 'superado'.

Quien atribuye a la crisis sus fracasos y penurias, violenta su propio talento y respeta más a los problemas que a las soluciones. La verdadera crisis, es la crisis de la incompetencia. El inconveniente de las personas y los países es la pereza para encontrar las salidas y soluciones. Sin crisis no hay desafíos, sin desafíos la vida es una rutina, una lenta afonía. Sin crisis no hay méritos. Es en la crisis donde aflora lo mejor de cada uno, porque sin crisis todo viento es caricia. Hablar de crisis es promoverla, y callar en la crisis es exaltar el conformismo. En vez de esto, trabajemos duro. Acabemos de una vez con la única crisis amenazadora, que es la tragedia de no querer luchar por superarla."

Y es que, ¿qué tendría de "divertido" un proyecto que no presentase metas que superar? Así que afronté el problema, y cuando surgieron los siguientes, sobre todo al realizar la interfaz gráfica de usuario, mi perspectiva había cambiado, incluso los problemas parecían menos problemas.

Finalmente, se obtuvo el resultado de carácter esencialmente técnico que se explica en esta memoria, cuyas características cumplen los requisitos que se propusieron en su día al comienzo de este proyecto. Recordándolas:

1. Usar el Wiimote como ratón inalámbrico Bluetooth para manejar presentaciones.

2. Usar el Wiimote como puntero láser virtual.

De esta manera, como se nos exigía, hemos cumplido con el análisis, diseño y desarrollo de una solución parcial al problema planteado inicialmente en nuestro ámbito de estudio; solución que satisface ambas premisas de los requisitos.

Destacar el término parcial del enunciado anterior, porque es un campo que, como expondremos más adelante, puede tener bastante futuro, siendo la base para proponer mejoras a este proyecto o desarrollar multitud de aplicaciones que tengan al mismo como base de su funcionamiento.

6.1.1 Ingeniería inversa

Llegados a este punto, donde finaliza un largo camino de horas y horas de trabajo, me gustaría pasar a tratar un par de reflexiones que me resultan de interés y unifican dos perspectivas muy diferentes.

Por una parte, destacaría el "entrenamiento" que nos ha sido inculcado a lo largo de estos años de carrera. En ellos, la mayoría de los problemas o de las prácticas planteadas se limitaban a seguir un guión a modo de instrucciones de uso, o a lo sumo se proponían tareas que podían ser solucionadas, de manera más o menos difícil, gracias a la documentación a la que se tenía acceso.

Hemos afrontado problemas de diseño e implementación donde partíamos de cero, o problemas de estudio donde lo que nos interesaba era conocer el comportamiento de una caja negra dadas sus entradas y sus salidas sin importar lo que había dentro.

Pero, realmente ¿qué es Ingeniería? Etimológicamente es una palabra derivada de "ingenio", y como tal hace referencia a la facultad que tiene el hombre para discurrir o inventar con creatividad y, en su fin último, de crear.

Con esto no trato de criticar la metodología universitaria, sino destacar que ahora si puedo asegurar que, a lo largo de este proyecto, se ha puesto a prueba el ingenio. Hemos conseguido determinar de qué está hecho el Wiimote, qué lo hace funcionar y como fue fabricado.

Para ello, hemos intentado seguir el camino contrario al que siguieron los ingenieros que lo desarrollaron, afrontando las dificultades que este hecho imponía, además de afrontar la falta de documentación al respecto por ser un producto comercial registrado.

Por otra parte, me gustaría tratar brevemente algo sobre la legitimidad de realizar este tipo de proyectos. Recuerdo que en una asignatura debatimos largo y tendido sobre este tema, ya que existen casos históricos de estudio donde muchos ingenieros creían que la ingeniería inversa, en particular la decodificación, no es ética en esencia.

En general, si el producto o material que fue sometido a la ingeniería inversa fue obtenido de manera apropiada, entonces el proceso es legítimo y legal. De la misma forma, pueden fabricarse y distribuirse legalmente los productos genéricos creados a partir de la información obtenida en el proceso de ingeniería inversa.

De hecho, y dadas las circunstancias, defiendo el desarrollo que siga esta filosofía para temas como el que nosotros estamos tratando, pues pueden suponer grandes ventajas incluso para el propietario primero y, sobre todo siguiendo la filosofía del software libre, la facilitación del acceso a determinados productos.

6.1.2 Pizarras Digitales Interactivas

La Pizarra Digital Interactiva (PDI) es un recurso TIC para utilizar principalmente en el aula, y que es ya una realidad en muchos centros educativos, siendo objeto de investigación por prácticamente todas las comunidades autónomas de nuestro país.

Están compuestas por un ordenador con conexión inalámbrica, un videoproyector y un "tablero interactivo" de manera que nos permiten escribir directamente sobre ellas y controlar los programas informáticos con un puntero (a veces simplemente con los dedos).

Figura 6.1: Pizarra Digital Interactiva

Además, la Pizarra Digital Interactiva permite:

- Escribir directamente sobre la pizarra, subrayados...
- Interactuar desde la pantalla con los programas,
- Disponer de otras utilidades del software asociado a la PDI.

Gracias a esta nueva generación de pizarras se busca conseguir la atención de los alumnos con clases dinámicas y contenidos sofisticados, captando su atención con la interacción en el tablero a la vez que se introducen las mejoras que ofrecen las TIC en este ámbito.

En la web del Plan Avanza, de la Secretaría de Estado de Telecomunicaciones y para la Sociedad de la Información están disponibles una serie de vídeos del evento que tuvo lugar el Día de Internet del año 2007. En ellos se hace presente la iniciativa de las empresas por dar a conocer las prestaciones de las PDIs desde entonces.

Veamos una serie de ejemplos comerciales, estudiando su funcionamiento y su capacidad interactiva para poder compararlos con las prestaciones que ofrece nuestro proyecto.

eBeam Edge for Education

El eBeam Edge for Education es fabricado y comercializado por Luidia Inc. y propone transformar una pizarra blanca del aula en un espacio interactivo de enseñanza. Tiene el tamaño de un borrador de pizarra y es más ligero que una caja de tizas. Proponen eBeam Edge for Education como el producto con más funciones de sistema de PDI móvil en el mercado.

Figura 6.2: eBeam Edge for Education

Si quieres más, puedes seguir ampliando las capacidades de la pizarra con eBeam Focus150 y eBeam Inscribe200e, productos que se integran totalmente con eBeam Edge for Education. Éstos son, respectivamente, una cámara que permite digitalizar material real de manera instantánea y una tableta inalámbrica.

Trabaja con los proyectores estándar, tanto en Windows como en Macintosh. Se integra fácilmente con las aplicaciones existentes y, por una fracción del coste de los sistemas fijos comparables, proporciona todas las características interactivas necesarias en este ámbito.

- Características:
  - USB
  - Compatibilidad
    - Microsoft Windows
    - Mac OS X
  - $899.95 USD

Hitachi StarBoard FX

Diseñada con la Educación en mente, la Pizarra Digital Interactiva FX tiene una superficie en la que los profesores y los estudiantes pueden colaborar en las actividades, explorar los sitios web, o hacer presentaciones de colores. Las pantallas de 63" y 77" se adaptan perfectamente a las aulas de tamaño tradicional, y permiten a los educadores mejorar las experiencias de aprendizaje a través de las herramientas proporcionadas por Starboard Software.

Figura 6.3: Hitachi StarBoard FX

Es fácil de usar, sólo tienes que conectar la StarBoard FX a un ordenador y un proyector LCD, y controlar la pizarra desde el StarBoard Software de manera sencilla en cuestión de minutos. La pizarra sólo responde a la pluma electrónica inalámbrica para garantizar una gran precisión mediante la eliminación de la interacción no deseada por el contacto del cuerpo u otros objetos.

- Características:
  - Bluetooth
  - Compatibilidad
    - Microsoft Windows
    - Mac OS X
  - 1320 €

3M Digital Board 578

La Pizarra Digital Interactiva desarrollada por 3M, está dirigida a educadores que necesitan presentar la información de manera atractiva, comprensible y con un formato de presentación motivador. En ella se pueden proyectar imágenes o ficheros como en el resto de pizarras sobre una superficie de 78", pudiendo resaltar, subrayar o destacar lo que se considere necesario y guardar los apuntes o comentarios escritos en una gran variedad de formatos electrónicos.

Figura 6.4: 3M Digital Board 578

Puede ser usada una y otra vez como una superficie de borrado en seco (con cualquier rotulador base agua) y limpiada fácilmente. Captura notas electrónicamente, para guardar y enviar al instante y permite acceder a Internet o a otras aplicaciones directamente desde la pizarra o navegar por el escritorio de su ordenador utilizando el lápiz de presentación, que se comporta como un ratón.

- Características:
  - USB
  - Compatibilidad
    - Microsoft Windows
    - Mac OS X
  - $1522 USD

Wiinux

Nuestra propuesta ofrece funcionalidades similares a las que dan los productos anteriores, permitiendo la interacción con el ordenador dentro del espacio delimitado por el aula. Sin embargo, el precio que se debe pagar para acceder a ellos es ínfimo sobre todo comparando los grandes pedidos que deberían hacer las entidades educativas, donde la diferencia de precios se haría aun más notable.

Si atendemos a los productos empleados para el desarrollo de nuestro proyecto y teniendo en cuenta que el resto de productos mencionados en esta sección dan por supuesto que el ordenador incorpora Bluetooh y se dispone del mismo así como del proyector, el coste ascendería a:

- Componentes:
  - Wiimote (Nintendo)
    - 37 €
  - Wiimote (3rd Party)
    - 18.90 €
  - Barra de infrarrojos inalámbrica (YS)
    - 4.90 €
  - **Total:**
    - 41.90 €(Nintendo - YS)
    - 23.80 €(3rd Party - YS)

Por tanto podríamos crear el listado siguiente para definir nuestro "producto":

- Características:
  - Bluetooth
  - Compatibilidad
    - Multiplataforma
  - 41.90 €(Nintendo - YS)
  - 23.80 €(3rd Party - YS)

Con todo esto vemos que nuestra alternativa es mucho más económica ofreciendo una gran relación prestaciones/precio y abriendo las puertas a una nueva opción de bajo coste.

6.2 Trabajos futuros

Está claro que esta aplicación facilitaría bastante las técnicas de enseñanza comunes mediante la interactividad profesor-alumno pero, como todo nuevo proyecto software, es sensible a cambios y modificaciones que lo hagan más atractivo y funcional surgiendo nuevas versiones posteriores.

En primer lugar, cabría la posibilidad de hacer crecer el programa haciendo que mejoren algunas experiencias de usuario, como por ejemplo la del movimiento, o añadiendo otras hasta ahora no estudiadas, por proponer algo, estudiar la funcionalidad extra que ofrecería el Wii Motion Plus o algún otro periférico y si podríamos obtener beneficios docentes con su inclusión.

Además, existen otra serie de proyectos cuyo coste tampoco es elevado, y podrían complementarse con este para incluir nuevas funcionalidades a la interacción con el material. Un claro ejemplo son los proyectos de reconocimiento de patrones o movimiento mediante una webcam, que podrían aportar otra serie de técnicas quizá bastante interesantes en este campo.

Pero sobre todo, las "mejoras" que se podrían introducir son las relacionadas con su uso docente. Al mapear directamente eventos de ratón y teclado, el Wiimote sería capaz de manejar multitud de aplicaciones que ya tenemos en nuestro ordenador, pero ahora de manera remota. Sin embrago, como otras muchas cosas nuevas, este proceso podría antojarse lento, ya que los períodos de implantación de las nuevas tecnologías, a veces no son tan cortos como quisiéramos. De esta manera, sería positivo el estudio de viabilidad para la inclusión de este sistema en las nuevas "aulas digitales".

Así mismo, podrían desarrollarse aplicaciones docentes específicas que ayuden a la implantación de este sistema, es decir, pasar de que la pizarra sea un apoyo a la docencia mediante el acceso a internet o la presentación de trabajos en clase por parte de los alumnos, a un nuevo escenario donde los recursos didácticos físicos se complementen con recursos digitales.

En este aspecto podrían tratarse dos pendientes:

- por un lado, la relacionada con la adquisición del material físico y digital de manera conjunta y cuyo desarrollo y costes tendrían que financiar las editoriales, incluyendo este material como anexo a sus publicaciones.

  Esta suposición podría convertirse en algo real a largo plazo, pero actualmente es muy probable que las editoriales sean reacias a desarrollar e incluir material nuevo evitando así sus costes derivados alegando el "hasta ahora nos ha funcionado".

- por otro lado, la que pasa por el desarrollo de nuevo software libre que sea capaz de adaptarse a las necesidades de cada área de conocimiento.

  Esta vía es más adecuada para dar a conocer las capacidades de las TICs en educación, así como para concienciar a las editoriales del camino hacia el que converge la educación en si, sin embargo, como el desarrollo de las aplicaciones se realizaría de manera genérica para garantizar la compatibilidad con todas las publicaciones, no explotarían todo el potencial docente de cada área.

Como podemos observar, existen multitud de trabajos que podrían estar relacionados con este proyecto, y más que podrían ir surgiendo con el uso e implantación de estos nuevos sistemas de PDI.

De manera genérica, y como mejora al sistema educativo tradicional, espero que el desarrollo de estas tecnologías sea posible, facilitando las labores de docentes y alumnos.

Apéndices

LD271H Datasheet

Optoelectrónica → LED → Emisores Infrarrojos
Fabricante → OSRAM Opto Semiconductors

- Especificaciones:
  - Material
    - GaAs
  - Encapsulado
    - Circular 5 mm gris
  - Pico de longitud de onda
    - 950 nm
  - Tipo
    - Emisor infrarrojo
  - V_F típ. a I_FmA
    - 1.3 @ 100
  - V_R máx.
    - 5V
  - Ángulo de visión
    - 50°

IR-Lumineszenzdiode
Infrared Emitter
Lead (Pb) Free Product - RoHS Compliant

LD 271
LD 271 H
LD 271 L
LD 271 LH

Wesentliche Merkmale

- GaAs-LED in 5mm radial-Gehäuse
- Typische Peakwellenlänge 950nm
- Hohe Zuverlässigkeit
- Mit verschiedenen Beinchenlängen lieferbar
- Variante mit “stand-off” lieferbar
- TTW Löten geeignet

Features

- GaAs-LED in 5mm radial package (T 1 $^3/_4$)
- Typical peak wavelength 950nm
- High reliability
- Available with two different lead lengths
- Version with stand-off available
- Suitable for TTW soldering

Anwendungen

- IR-Fernsteuerung von Fernseh- und Rundfunkgeräten, Videorecordern, Lichtdimmern
- Gerätefernsteuerungen für Gleich- und Wechsellichtbetrieb
- Sensorik
- Diskrete Lichtschranken

Applications

- IR remote control of hi-fi and TV-sets, video tape recorders, dimmers
- Remote control for steady and varying intensity
- Sensor technology
- Discrete interrupters

Typ **Type**	**Bestellnummer** **Ordering Code**	**Strahlstärkegruppierung** [1] (I_F = 100mA, t_p = 20 ms) **Radiant intensity grouping** [1] I_e (mW/sr)
LD 271	Q62703Q0148	15 (>10)
LD 271 L	Q62703Q0833	
LD 271 H	Q62703Q0256	>16
LD 271 LH	Q62703Q0838	

[1] gemessen bei einem Raumwinkel Ω = 0.01 sr
measured at a solid angle of Ω = 0.01 sr

Grenzwerte
Maximum Ratings

Bezeichnung **Parameter**	**Symbol** **Symbol**	**Wert** **Value**	**Einheit** **Unit**
Betriebs- und Lagertemperatur Operating and storage temperature range	T_{op}; T_{stg}	– 40 … + 100	°C
Sperrspannung Reverse voltage	V_R	5	V
Durchlaßstrom Forward current	I_F	130	mA
Stoßstrom, t_p = 10 µs, D = 0 Surge current	I_{FSM}	3.5	A
Verlustleistung Power dissipation	P_{tot}	220	mW
Wärmewiderstand Thermal resistance	R_{thJA}	330	K/W

Kennwerte (T_A = 25 °C)
Characteristics

Bezeichnung **Parameter**	**Symbol** **Symbol**	**Wert** **Value**	**Einheit** **Unit**
Wellenlänge der Strahlung Wavelength at peak emission I_F = 100 mA, t_p = 20 ms	λ_{peak}	950	nm
Spektrale Bandbreite bei 50% von I_{max} Spectral bandwidth at 50% of I_{max} I_F = 100 mA	$\Delta\lambda$	55	nm
Abstrahlwinkel Half angle	φ	± 25	Grad deg.
Aktive Chipfläche Active chip area	A	0.25	mm^2
Abmessungen der aktiven Chipfläche Dimensions of the active chip area	$L \times B$ $L \times W$	0.5 × 0.5	mm
Abstand Chipoberfläche bis Linsenscheitel Distance chip front to lens top	H	4.0 … 4.6	mm
Schaltzeiten, I_e von 10% auf 90% und von 90% auf 10%, bei I_F = 100 mA, R_L = 50 Ω Switching times, I_e from 10% to 90% and from 90% to 10%, I_F = 100 mA, R_L = 50 Ω	t_r, t_f	1	µs

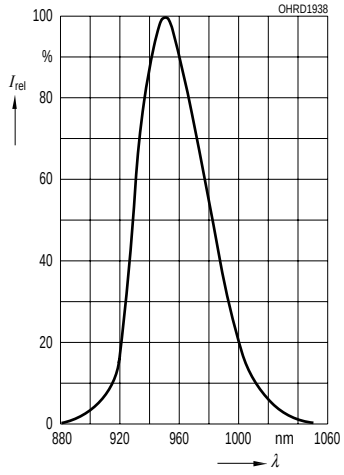
Kennwerte (T_A = 25 °C)
Characteristics (cont'd)

Bezeichnung **Parameter**	**Symbol** **Symbol**	**Wert** **Value**	**Einheit** **Unit**
Kapazität, V_R = 0 V, f = 1 MHz Capacitance	C_o	40	pF
Durchlaßspannung Forward voltage I_F = 100 mA, t_p = 20 ms I_F = 1 A, t_p = 100 µs	 V_F V_F	 1.30 (≤ 1.5) 1.90 (≤ 2.5)	 V V
Sperrstrom, V_R = 5 V Reverse current	I_R	0.01 (≤ 1)	µA
Gesamtstrahlungsfluß Total radiant flux I_F = 100 mA, t_p = 20 ms	Φ_e	18	mW
Temperaturkoeffizient von I_e bzw. Φ_e, I_F = 100 mA Temperature coefficient of I_e or Φ_e, I_F = 100 mA	TC_I	– 0.55	%/K
Temperaturkoeffizient von V_F, I_F = 100 mA Temperature coefficient of V_F, I_F = 100 mA	TC_V	– 1.5	mV/K
Temperaturkoeffizient von λ, I_F = 100 mA Temperature coefficient of λ, I_F = 100 mA	TC_λ	0.3	nm/K

Gruppierung der Strahlstärke I_e in Achsrichtung
gemessen bei einem Raumwinkel Ω = 0.01 sr
Grouping of Radiant Intensity I_e in Axial Direction
at a solid angle of Ω = 0.01 sr

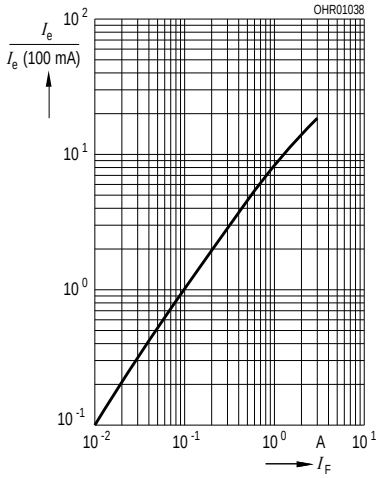
Bezeichnung **Parameter**	**Symbol** **Symbol**	**Wert** **Value**		**Einheit** **Unit**
		LD 271 **LD 271 L**	**LD 271 H** **LD 271 LH**	
Strahlstärke Radiant intensity I_F = 100 mA, t_p = 20 ms I_F = 1 A, t_p = 100 µs	 I_e $I_{e\ typ.}$	 15 (> 10) 120	 > 16	 mW/sr mW/sr

Relative Spectral emission
$I_{rel} = f(\lambda)$

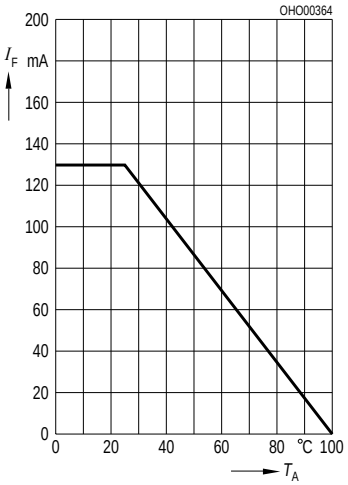


Radiant Intensity $\frac{I_e}{I_e\ 100\ mA} = f(I_F)$

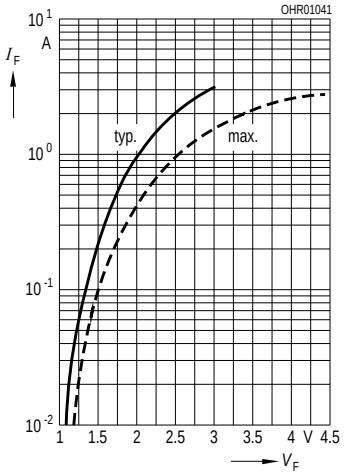
Single pulse, t_p = 20 µs



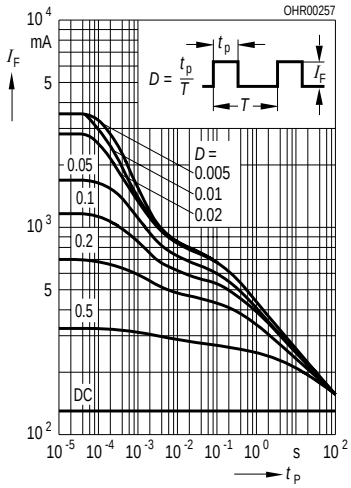
Max. Permissible Forward Current
$I_F = f(T_A)$



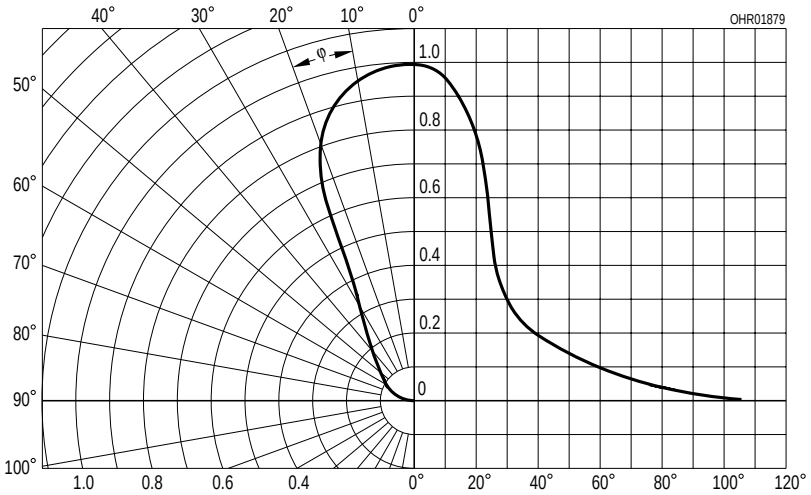
Forward Current
$I_F = f(V_F)$, single pulse, t_p = 20 µs



Permissible Pulse Handling Capability $I_F = f(\tau)$, T_C = 25 °C, duty cycle D = parameter

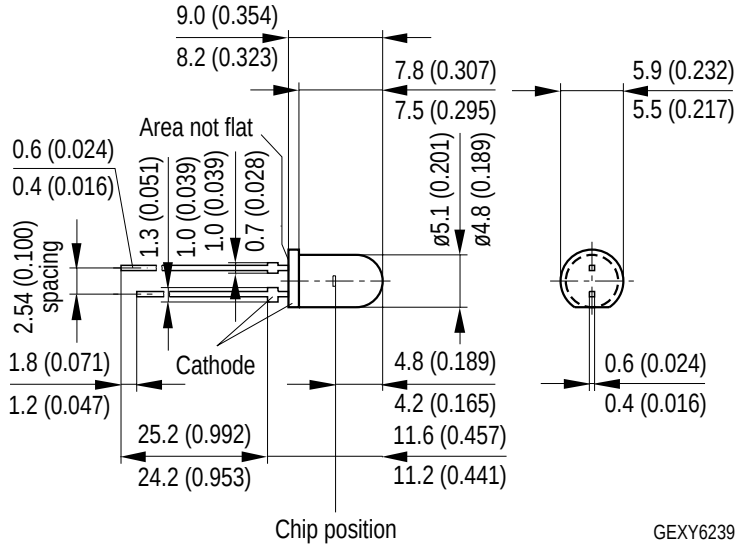


Radiation Characteristics $I_{rel} = f(\varphi)$

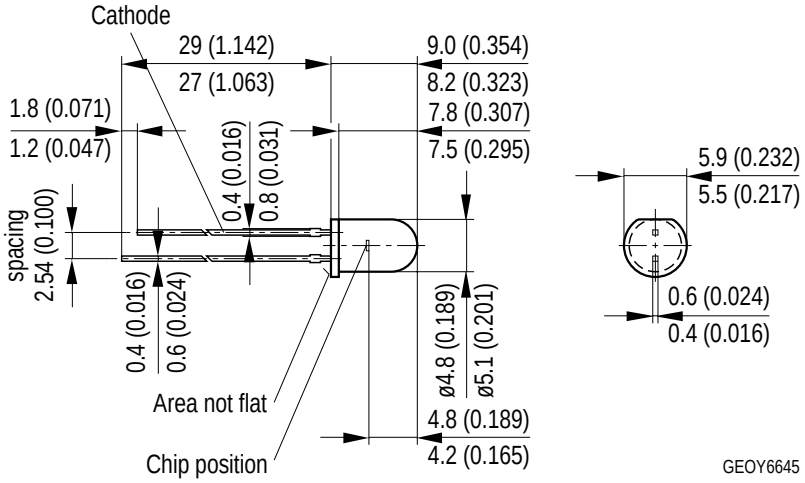


Maßzeichnung
Package Outlines

LD 271 / LD 271 H



LD 271 L / LD 271 LH



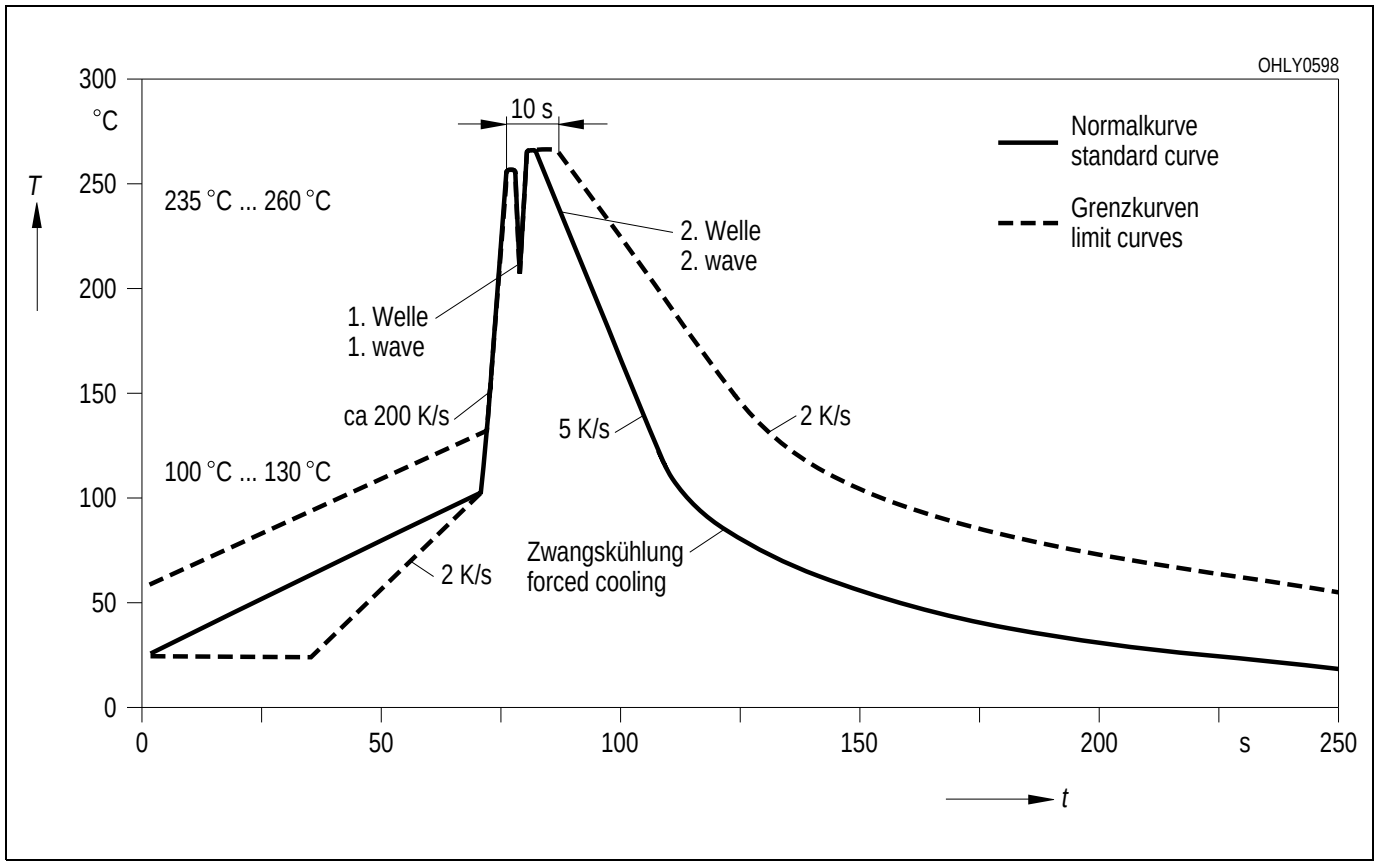
Maße werden wie folgt angegeben: mm (inch) / Dimensions are specified as follows: mm (inch).

Gehäusefarbe: grau
Brechungsindex Verguss: 1.53

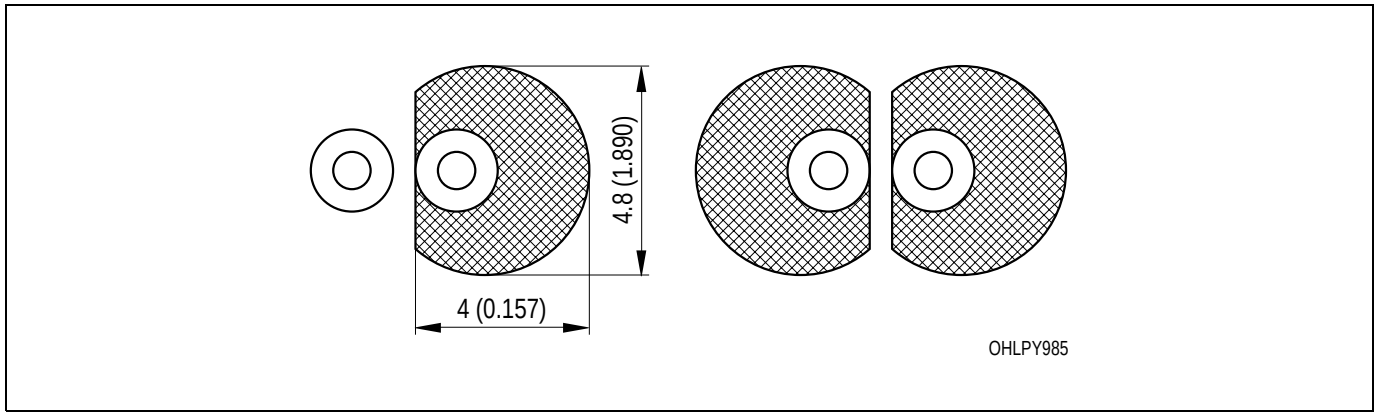
Package Coluor: grey
Refractive index resin: 1.53

Lötbedingungen
Soldering Conditions

Wellenlöten (TTW) (nach CECC 00802)
TTW Soldering (acc. to CECC 00802)



Empfohlenes Lötpaddesign Wellenlöten (TTW)
Recommended Solder Pad TTW Soldering



Maße werden wie folgt angegeben: mm (inch) / Dimensions are specified as follows: mm (inch).

Texto completo de la licencia para la distribución del software

Accesible en: `http://www.gnu.org/licenses/gpl.html`

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. **Definitions.**

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. **Source Code.**

   The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

   A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it

does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. **Basic Permissions.**

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. **Protecting Users' Legal Rights From Anti-Circumvention Law.**

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. **Conveying Verbatim Copies.**

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. **Conveying Modified Source Versions.**

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. **Conveying Non-Source Forms.**

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the

object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement

to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. **Additional Terms.**

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. **Termination.**

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this

is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. **Acceptance Not Required for Having Copies.**

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. **Automatic Licensing of Downstream Recipients.**

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. **Patents.**

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you

grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. **No Surrender of Others' Freedom.**

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. **Use with the GNU Affero General Public License.**

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. **Revised Versions of this License.**

    The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

    Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. **Disclaimer of Warranty.**

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. **Limitation of Liability.**

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF

THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. **Interpretation of Sections 15 and 16.**

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

Texto completo de la licencia para la distribución de la memoria

Accesible en: `http://creativecommons.org/licenses/by-sa/3.0/es/`

Código Legal de Creative Commons

Reconocimiento-CompartirIgual 3.0 España

CREATIVE COMMONS CORPORATION NO ES UN DESPACHO DE ABOGADOS Y NO PROPORCIONA SERVICIOS JURÍDICOS. LA DISTRIBUCIÓN DE ESTA LICENCIA NO CREA UNA RELACIÓN ABOGADO-CLIENTE. CREATIVE COMMONS PROPORCIONA ESTA INFORMACIÓN TAL CUAL (ON AN "AS-IS" BASIS). CREATIVE COMMONS NO OFRECE GARANTÍA ALGUNA RESPECTO DE LA INFORMACIÓN PROPORCIONADA, NI ASUME RESPONSABILIDAD ALGUNA POR DAÑOS PRODUCIDOS A CONSECUENCIA DE SU USO.

Licencia

LA OBRA O LA PRESTACIÓN (SEGÚN SE DEFINEN MÁS ADELANTE) SE PROPORCIONA BAJO LOS TÉRMINOS DE ESTA LICENCIA PÚBLICA DE CREATIVE COMMONS (**CCPL** O **LICENCIA**). LA OBRA O LA PRESTACIÓN SE ENCUENTRA PROTEGIDA POR LA LEY ESPAÑOLA DE PROPIEDAD INTELECTUAL Y/O CUALESQUIERA OTRAS NORMAS QUE RESULTEN DE APLICACIÓN. QUEDA PROHIBIDO CUALQUIER USO DE LA OBRA O PRESTACIÓN DIFERENTE A LO AUTORIZADO BAJO ESTA LICENCIA O LO DISPUESTO EN LA LEY DE PROPIEDAD INTELECTUAL.

MEDIANTE EL EJERCICIO DE CUALQUIER DERECHO SOBRE LA OBRA O LA PRESTACIÓN, USTED ACEPTA Y CONSIENTE LAS LIMITACIONES Y OBLIGACIONES DE ESTA LICENCIA, SIN PERJUICIO DE LA NECESIDAD DE CONSENTIMIENTO EXPRESO EN CASO DE VIOLACIÓN PREVIA DE LOS TÉRMINOS DE LA MISMA. EL LICENCIADOR LE CONCEDE LOS DERECHOS CONTENIDOS EN ESTA LICENCIA, SIEMPRE QUE USTED ACEPTE LOS PRESENTES TÉRMINOS Y CONDICIONES.

1. **Definiciones.**

   a) La **obra** es la creación literaria, artística o científica ofrecida bajo los términos de esta licencia.

b) En esta licencia se considera una **prestación** cualquier interpretación, ejecución, fonograma, grabación audiovisual, emisión o transmisión, mera fotografía u otros objetos protegidos por la legislación de propiedad intelectual vigente aplicable.

c) La aplicación de esta licencia a una **colección** (definida más adelante) afectará únicamente a su estructura en cuanto forma de expresión de la selección o disposición de sus contenidos, no siendo extensiva a éstos. En este caso la colección tendrá la consideración de obra a efectos de esta licencia.

d) El **titular originario** es:

1) En el caso de una obra literaria, artística o científica, la persona natural o grupo de personas que creó la obra.
2) En el caso de una obra colectiva, la persona que la edite y divulgue bajo su nombre, salvo pacto contrario.
3) En el caso de una interpretación o ejecución, el actor, cantante, músico, o cualquier otra persona que represente, cante, lea, recite, interprete o ejecute en cualquier forma una obra.
4) En el caso de un fonograma, el productor fonográfico, es decir, la persona natural o jurídica bajo cuya iniciativa y responsabilidad se realiza por primera vez una fijación exclusivamente sonora de la ejecución de una obra o de otros sonidos.
5) En el caso de una grabación audiovisual, el productor de la grabación, es decir, la persona natural o jurídica que tenga la iniciativa y asuma la responsabilidad de las fijaciones de un plano o secuencia de imágenes, con o sin sonido.
6) En el caso de una emisión o una transmisión, la entidad de radiodifusión.
7) En el caso de una mera fotografía, aquella persona que la haya realizado.
8) En el caso de otros objetos protegidos por la legislación de propiedad intelectual vigente, la persona que ésta señale.

e) Se considerarán **obras derivadas** aquellas obras creadas a partir de la licenciada, como por ejemplo: las traducciones y adaptaciones; las revisiones, actualizaciones y anotaciones; los compendios, resúmenes y extractos; los arreglos musicales y, en general, cualesquiera transformaciones de una obra literaria, artística o científica. Para evitar la duda, si la obra consiste en una composición musical o grabación de sonidos, la sincronización temporal de la obra con

una imagen en movimiento (synching) será considerada como una obra derivada a efectos de esta licencia.

f) Tendrán la consideración de **colecciones** la recopilación de obras ajenas, de datos o de otros elementos independientes como las antologías y las bases de datos que por la selección o disposición de sus contenidos constituyan creaciones intelectuales. La mera incorporación de una obra en una colección no dará lugar a una derivada a efectos de esta licencia.

g) El **licenciador** es la persona o la entidad que ofrece la obra o prestación bajo los términos de esta licencia y le concede los derechos de explotación de la misma conforme a lo dispuesto en ella.

h) **Usted** es la persona o la entidad que ejercita los derechos concedidos mediante esta licencia y que no ha violado previamente los términos de la misma con respecto a la obra o la prestación, o que ha recibido el permiso expreso del licenciador de ejercitar los derechos concedidos mediante esta licencia a pesar de una violación anterior.

i) La **transformación** de una obra comprende su traducción, adaptación y cualquier otra modificación en su forma de la que se derive una obra diferente. La creación resultante de la transformación de una obra tendrá la consideración de obra derivada.

j) Se entiende por **reproducción** la fijación directa o indirecta, provisional o permanente, por cualquier medio y en cualquier forma, de toda la obra o la prestación o de parte de ella, que permita su comunicación o la obtención de copias.

k) Se entiende por **distribución** la puesta a disposición del público del original o de las copias de la obra o la prestación, en un soporte tangible, mediante su venta, alquiler, préstamo o de cualquier otra forma.

l) Se entiende por **comunicación pública** todo acto por el cual una pluralidad de personas, que no pertenezcan al ámbito doméstico de quien la lleva a cabo, pueda tener acceso a la obra o la prestación sin previa distribución de ejemplares a cada una de ellas. Se considera comunicación pública la puesta a disposición del público de obras o prestaciones por procedimientos alámbricos o inalámbricos, de tal forma que cualquier persona pueda acceder a ellas desde el lugar y en el momento que elija.

m) La **explotación** de la obra o la prestación comprende la reproducción, la distribución, la comunicación pública y, en su caso, la transformación.

n) Los **elementos de la licencia** son las características principales de la licencia según la selección efectuada por el licenciador e indicadas en el título de esta licencia: Reconocimiento, CompartirIgual.

ñ) Una **licencia equivalente** es:

1) Una versión posterior de esta licencia de Creative Commons con los mismos elementos de licencia.

2) La misma versión o una versión posterior de esta licencia de cualquier otra jurisdicción reconocida por Creative Commons con los mismos elementos de la licencia (ejemplo: Reconocimiento-CompartirIgual 3.0 Japón).

3) La misma versión o una versión posterior de la licencia de Creative Commons no adaptada a ninguna jurisdicción (*Unported*) con los mismos elementos de la licencia.

4) Una de las licencias compatibles que aparece en http://creativecommons.org/compatiblelicenses y que ha sido aprobada por Creative Commons como esencialmente equivalente a esta licencia porque, como mínimo:

a′ Contiene términos con el mismo propósito, el mismo significado y el mismo efecto que los elementos de esta licencia.

b′ Permite explícitamente que las obras derivadas de obras sujetas a ella puedan ser distribuidas mediante esta licencia, la licencia de Creative Commons no adaptada a ninguna jurisdicción (*Unported*) o una licencia de cualquier otra jurisdicción reconocida por Creative Commons, con sus mismos elementos de licencia.

2. **Límites de los derechos.**

Nada en esta licencia pretende reducir o restringir cualesquiera límites legales de los derechos exclusivos del titular de los derechos de propiedad intelectual de acuerdo con la Ley de propiedad intelectual o cualesquiera otras leyes aplicables, ya sean derivados de usos legítimos, tales como la copia privada o la cita, u otras limitaciones como la resultante de la primera venta de ejemplares (agotamiento).

3. **Concesión de licencia.**

Conforme a los términos y a las condiciones de esta licencia, el licenciador concede, por el plazo de protección de los derechos de propiedad intelectual y a título gratuito, una licencia de ámbito mundial no exclusiva que incluye los derechos siguientes:

a) Derecho de reproducción, distribución y comunicación pública de la obra o la prestación.

b) Derecho a incorporar la obra o la prestación en una o más colecciones.

c) Derecho de reproducción, distribución y comunicación pública de la obra o la prestación lícitamente incorporada en una colección.

d) Derecho de transformación de la obra para crear una obra derivada siempre y cuando se incluya en ésta una indicación de la transformación o modificación efectuada.

e) Derecho de reproducción, distribución y comunicación pública de obras derivadas creadas a partir de la obra licenciada.

f) Derecho a extraer y reutilizar la obra o la prestación de una base de datos.

g) Para evitar cualquier duda, el titular originario:

1) Conserva el derecho a percibir las remuneraciones o compensaciones previstas por actos de explotación de la obra o prestación, calificadas por la ley como irrenunciables e inalienables y sujetas a gestión colectiva obligatoria.

2) Renuncia al derecho exclusivo a percibir, tanto individualmente como mediante una entidad de gestión colectiva de derechos, cualquier remuneración derivada de actos de explotación de la obra o prestación que usted realice.

Estos derechos se pueden ejercitar en todos los medios y formatos, tangibles o intangibles, conocidos en el momento de la concesión de esta licencia. Los derechos mencionados incluyen el derecho a efectuar las modificaciones que sean precisas técnicamente para el ejercicio de los derechos en otros medios y formatos. Todos los derechos no concedidos expresamente por el licenciador quedan reservados, incluyendo, a título enunciativo pero no limitativo, los derechos morales irrenunciables reconocidos por la ley aplicable. En la medida en que el licenciador ostente derechos exclusivos previstos por la ley nacional vigente que implementa la directiva europea en materia de derecho sui generis sobre bases de datos, renuncia expresamente a dichos derechos exclusivos.

4. **Restricciones.**

La concesión de derechos que supone esta licencia se encuentra sujeta y limitada a las restricciones siguientes:

a) Usted puede reproducir, distribuir o comunicar públicamente la obra o prestación solamente bajo los términos de esta licencia y debe incluir una copia de la misma, o su Identificador Uniforme de Recurso (URI). Usted no puede ofrecer o imponer ninguna condición sobre la obra o prestación que altere o restrinja

los términos de esta licencia o el ejercicio de sus derechos por parte de los concesionarios de la misma. Usted no puede sublicenciar la obra o prestación. Usted debe mantener intactos todos los avisos que se refieran a esta licencia y a la ausencia de garantías. Usted no puede reproducir, distribuir o comunicar públicamente la obra o prestación con medidas tecnológicas que controlen el acceso o el uso de una manera contraria a los términos de esta licencia. Esta sección 4.a también afecta a la obra o prestación incorporada en una colección, pero ello no implica que ésta en su conjunto quede automáticamente o deba quedar sujeta a los términos de la misma. En el caso que le sea requerido, previa comunicación del licenciador, si usted incorpora la obra en una colección y/o crea una obra derivada, deberá quitar cualquier crédito requerido en el apartado 4.c, en la medida de lo posible.

b) Usted puede distribuir o comunicar públicamente una obra derivada en el sentido de esta licencia solamente bajo los términos de la misma u otra licencia equivalente. Si usted utiliza esta misma licencia debe incluir una copia o bien su URI, con cada obra derivada que usted distribuya o comunique públicamente. Usted no puede ofrecer o imponer ningún término respecto a la obra derivada que altere o restrinja los términos de esta licencia o el ejercicio de sus derechos por parte de los concesionarios de la misma. Usted debe mantener intactos todos los avisos que se refieran a esta licencia y a la ausencia de garantías cuando distribuya o comunique públicamente la obra derivada. Usted no puede ofrecer o imponer ningún término respecto de las obras derivadas o sus transformaciones que alteren o restrinjan los términos de esta licencia o el ejercicio de sus derechos por parte de los concesionarios de la misma. Usted no puede reproducir, distribuir o comunicar públicamente la obra derivada con medidas tecnológicas que controlen el acceso o uso de la obra de una manera contraria a los términos de esta licencia. Si utiliza una licencia equivalente debe cumplir con los requisitos que ésta establezca cuando distribuya o comunique públicamente la obra derivada. Todas estas condiciones se aplican a una obra derivada en tanto que incorporada a una colección, pero no implica que ésta tenga que estar sujeta a los términos de esta licencia.

c) Si usted reproduce, distribuye o comunica públicamente la obra o la prestación, una colección que la incorpore o cualquier obra derivada, debe mantener intactos todos los avisos sobre la propiedad intelectual e indicar, de manera razonable conforme al medio o a los medios que usted esté utilizando:

1) El nombre del autor original, o el seudónimo si es el caso, así como el del titular originario, si le es facilitado.

2) El nombre de aquellas partes (por ejemplo: institución, publicación, revista) que el titular originario y/o el licenciador designen para ser reconocidos en el aviso legal, las condiciones de uso, o de cualquier otra manera razonable.

3) El título de la obra o la prestación si le es facilitado.

4) El URI, si existe, que el licenciador especifique para ser vinculado a la obra o la prestación, a menos que tal URI no se refiera al aviso legal o a la información sobre la licencia de la obra o la prestación.

5) En el caso de una obra derivada, un aviso que identifique la transformación de la obra en la obra derivada (p. ej., "traducción castellana de la obra de Autor Original", o "guión basado en obra original de Autor Original").

Este reconocimiento debe hacerse de manera razonable. En el caso de una obra derivada o incorporación en una colección estos créditos deberán aparecer como mínimo en el mismo lugar donde se hallen los correspondientes a otros autores o titulares y de forma comparable a los mismos. Para evitar la duda, los créditos requeridos en esta sección sólo serán utilizados a efectos de atribución de la obra o la prestación en la manera especificada anteriormente. Sin un permiso previo por escrito, usted no puede afirmar ni dar a entender implícitamente ni explícitamente ninguna conexión, patrocinio o aprobación por parte del titular originario, el licenciador y/o las partes reconocidas hacia usted o hacia el uso que hace de la obra o la prestación.

d) Para evitar cualquier duda, debe hacerse notar que las restricciones anteriores (párrafos 4.a, 4.b y 4.c) no son de aplicación a aquellas partes de la obra o la prestación objeto de esta licencia que únicamente puedan ser protegidas mediante el derecho sui generis sobre bases de datos recogido por la ley nacional vigente implementando la directiva europea de bases de datos

5. **Exoneración de responsabilidad.**

A MENOS QUE SE ACUERDE MUTUAMENTE ENTRE LAS PARTES, EL LICENCIADOR OFRECE LA OBRA O LA PRESTACIÓN TAL CUAL (ON AN "AS-IS" BASIS) Y NO CONFIERE NINGUNA GARANTÍA DE CUALQUIER TIPO RESPECTO DE LA OBRA O LA PRESTACIÓN O DE LA PRESENCIA O AUSENCIA DE ERRORES QUE PUEDAN O NO SER DESCUBIERTOS.

ALGUNAS JURISDICCIONES NO PERMITEN LA EXCLUSIÓN DE TALES GARANTÍAS, POR LO QUE TAL EXCLUSIÓN PUEDE NO SER DE APLICACIÓN A USTED.

6. **Limitación de responsabilidad.**

   SALVO QUE LO DISPONGA EXPRESA E IMPERATIVAMENTE LA LEY APLICABLE, EN NINGÚN CASO EL LICENCIADOR SERÁ RESPONSABLE ANTE USTED POR CUALESQUIERA DAÑOS RESULTANTES, GENERALES O ESPECIALES (INCLUIDO EL DAÑO EMERGENTE Y EL LUCRO CESANTE), FORTUITOS O CAUSALES, DIRECTOS O INDIRECTOS, PRODUCIDOS EN CONEXIÓN CON ESTA LICENCIA O EL USO DE LA OBRA O LA PRESTACIÓN, INCLUSO SI EL LICENCIADOR HUBIERA SIDO INFORMADO DE LA POSIBILIDAD DE TALES DAÑOS.

7. **Finalización de la licencia**

   a) Esta licencia y la concesión de los derechos que contiene terminarán automáticamente en caso de cualquier incumplimiento de los términos de la misma. Las personas o entidades que hayan recibido de usted obras derivadas o colecciones bajo esta licencia, sin embargo, no verán sus licencias finalizadas, siempre que tales personas o entidades se mantengan en el cumplimiento íntegro de esta licencia. Las secciones 1, 2, 5, 6, 7 y 8 permanecerán vigentes pese a cualquier finalización de esta licencia.

   b) Conforme a las condiciones y términos anteriores, la concesión de derechos de esta licencia es vigente por todo el plazo de protección de los derechos de propiedad intelectual según la ley aplicable. A pesar de lo anterior, el licenciador se reserva el derecho a divulgar o publicar la obra o la prestación en condiciones distintas a las presentes, o de retirar la obra o la prestación en cualquier momento. No obstante, ello no supondrá dar por concluida esta licencia (o cualquier otra licencia que haya sido concedida, o sea necesario ser concedida, bajo los términos de esta licencia), que continuará vigente y con efectos completos a no ser que haya finalizado conforme a lo establecido anteriormente, sin perjuicio del derecho moral de arrepentimiento en los términos reconocidos por la ley de propiedad intelectual aplicable.

8. **Miscelánea**

   a) Cada vez que usted realice cualquier tipo de explotación de la obra o la

prestación, o de una colección que la incorpore, el licenciador ofrece a los terceros y sucesivos licenciatarios la concesión de derechos sobre la obra o la prestación en las mismas condiciones y términos que la licencia concedida a usted.

b) Cada vez que usted realice cualquier tipo de explotación de una obra derivada, el licenciador ofrece a los terceros y sucesivos licenciatarios la concesión de derechos sobre la obra objeto de esta licencia en las mismas condiciones y términos que la licencia concedida a usted.

c) Si alguna disposición de esta licencia resulta inválida o inaplicable según la Ley vigente, ello no afectará la validez o aplicabilidad del resto de los términos de esta licencia y, sin ninguna acción adicional por cualquiera las partes de este acuerdo, tal disposición se entenderá reformada en lo estrictamente necesario para hacer que tal disposición sea válida y ejecutiva.

d) No se entenderá que existe renuncia respecto de algún término o disposición de esta licencia, ni que se consiente violación alguna de la misma, a menos que tal renuncia o consentimiento figure por escrito y lleve la firma de la parte que renuncie o consienta.

e) Esta licencia constituye el acuerdo pleno entre las partes con respecto a la obra o la prestación objeto de la licencia. No caben interpretaciones, acuerdos o condiciones con respecto a la obra o la prestación que no se encuentren expresamente especificados en la presente licencia. El licenciador no estará obligado por ninguna disposición complementaria que pueda aparecer en cualquier comunicación que le haga llegar usted. Esta licencia no se puede modificar sin el mutuo acuerdo por escrito entre el licenciador y usted.

Aviso de Creative Commons

Creative Commons no es parte de esta licencia, y no ofrece ninguna garantía en relación con la obra o la prestación. Creative Commons no será responsable frente a usted o a cualquier parte, por cualesquiera daños resultantes, incluyendo, pero no limitado, daños generales o especiales (incluido el daño emergente y el lucro cesante), fortuitos o causales, en conexión con esta licencia. A pesar de las dos (2) oraciones anteriores, si Creative Commons se ha identificado expresamente como el licenciador, tendrá todos los derechos y obligaciones del licenciador.

Salvo para el propósito limitado de indicar al público que la obra o la prestación está licenciada bajo la CCPL, ninguna parte utilizará la marca registrada "Creative Commons"

o cualquier marca registrada o insignia relacionada con "Creative Commons" sin su consentimiento por escrito. Cualquier uso permitido se hará de conformidad con las pautas vigentes en cada momento sobre el uso de la marca registrada por "Creative Commons", en tanto que sean publicadas su sitio web (website) o sean proporcionadas a petición previa. Para evitar cualquier duda, estas restricciones en el uso de la marca no forman parte de esta licencia.

Puede contactar con Creative Commons en: `http://creativecommons.org/`.

Glosario

API

Application Programming Interface. Interfaz de Programación de Aplicaciones, es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Bluetooth

Especificación para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda de los 2,4 GHz.

Los dispositivos que con mayor frecuencia utilizan esta tecnología pertenecen a sectores de las telecomunicaciones y la informática personal tales como teléfonos, impresoras, módems y auriculares.

Bluetooth SIG

Bluetooth Special Interest Group. Grupo de empresas interesadas en la creación de la tecnología Bluetooth y que, hoy en día, recoge a más de 11.000 compañías como miembros del grupo desarrollando, implementando y vendiendo cientos de productos con Bluetooth en el mundo.

COM

Component Object Model. Modelo de Objetos basado en Componentes, es una plataforma de Microsoft para componentes de software introducida por dicha empresa en 1993.

Esta plataforma es utilizada para permitir la comunicación entre procesos y la creación dinámica de objetos, en cualquier lenguaje de programación que soporte dicha tecnología. Esencialmente, COM es una manera de implementar objetos neutral con respecto al lenguaje, de manera que pueden ser usados en entornos distintos de aquel en que fueron creados

CORBA

Common Object Request Broker Architecture. Arquitectura Común de Intermediarios en Peticiones a Objetos, es un estándar que establece una plataforma de desarrollo de sistemas distribuidos facilitando la invocación de métodos remotos bajo un paradigma orientado a objetos.

En sentido general, CORBA "envuelve" el código escrito en otro lenguaje, en un paquete que contiene información adicional sobre las capacidades del código que contiene y sobre cómo llamar a sus métodos. Los objetos que resultan, pueden entonces ser invocados desde otro programa (u objeto CORBA) desde la red. En este sentido CORBA se puede considerar como un formato de documentación legible por la máquina, similar a un archivo de cabeceras, pero con más información.

EDR

Enhanced Data Rate. Tasa de Datos Mejorada, es una técnica que permite a Bluetooth mejorar las velocidades de transmisión en hasta 3Mbps en su versión 2.0, incremento significativo con las velocidades que proporcionaba la primera versión de Bluetooth.

GAP

Generic Access Profile. Perfil de Acceso Genérico base para todos los demás perfiles Bluetooth.

GNU

GNU is Not Unix. Acrónimo recursivo para definir el proyecto iniciado por Richard Stallman con el objetivo de crear un sistema operativo completamente libre.

GTK+

The GIMP Toolkit. Es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME, XFCE y ROX aunque también se puede usar en el escritorio de Windows, MacOS y otros.

HCI

Human-Computer Interaction. Interacción Persona Ordenador es un perfil que da la abstracción necesaria para el intercambio de información entre las personas y los computadores.

HID

Human Interface Device. Dispositivo de Interfaz Humana. Da soporte a gran cantidad de dispositivos periféricos tales como ratones, joysticks y teclados que interactúan directamente con humanos. Se diseñó para ofrecer un enlace de baja latencia manteniendo bajo el consumo.

IEEE

Institute of Electrical and Electronics Engineers. Instituto de Ingenieros Electricistas y Electrónicos, es una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas. Es la mayor asociación internacional sin ánimo de lucro formada por profesionales de las nuevas tecnologías, como ingenieros electricistas, ingenieros en electrónica, ingenieros en informática, ingenieros en biomedicina e ingenieros en telecomunicación.

L2CAP

Link Layer Control and Adaptation Protocol. Protocolo de Control y Adaptación del Enlace Lógico, se utiliza para pasar paquetes con y sin orientación a la conexión a sus capas superiores incluyendo tanto al Host Controller Interface (HCI) como directamente al gestor del enlace.

Las funciones de L2CAP incluyen:

- Segmentación y reensamblado de paquetes. Acepta paquetes de hasta 64KB de sus capas superior.
- Multiplexación de varias fuentes de paquetes, comprobando el protocolo de las capas superiores para así adaptarlo antes del reensamblaje.
- Proporcionar una buena gestión para la transmisión unidireccional a otros dispositivos bluetooth.
- Gestión de la calidad de servicio (QoS); para los protocolos de las capas superiores. En esta fase negocia el tamaño máximo del campo de datos de las tramas. Con ello, evita que algún dispositivo envíe paquetes tan grandes que puedan desbordar al receptor.

LED

Light-Emitting Diode. Diodo Emisor de Luz, es un dispositivo semiconductor que emite luz incoherente de espectro reducido cuando se polariza de forma directa la unión PN del mismo y circula por él una corriente eléctrica.

MAC

Media Access Control. Control de Acceso al Medio, es el conjunto de mecanismos y protocolos por los que varios "interlocutores" (dispositivos en una red, como ordenadores, teléfonos móviles, etc.) se ponen de acuerdo para compartir un medio de transmisión común (por lo general, un cable eléctrico u óptico, o en comunicaciones inalámbricas el rango de frecuencias asignado a su sistema).

MAC address

La dirección MAC, es un identificador de 48 bits que se corresponde de forma única con una interfaz de red.

Matplotlib

Es una biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy. Proporciona una API, pylab, diseñada para recordar a la de MATLAB.

Mii

La palabra "Mii" viene de la similitud a la palabra en inglés "me" que significa yo. Es un avatar representado con forma de personaje utilizado para diversas funciones de la videoconsola Wii, siendo creados por los usuarios de la misma.

Pueden ser almacenados en la consola o en el Wiimote, para facilitar el transporte de estos a otra consola Wii sin necesidad de conexión a Internet. Pueden hacerse a similitud del usuario o según la creatividad se pueden crear personajes de todo tipo.

.NET

Framework de Microsoft que hace un énfasis en la transparencia de redes, con independencia de plataforma de hardware y que permita un rápido desarrollo de aplicaciones. Basado en ella, la empresa intenta desarrollar una estrategia horizontal que integre todos sus productos, desde el sistema operativo hasta las herramientas de mercado.

.NET podría considerarse una respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Sun Microsystems y a los diversos framework de desarrollo web basados en PHP. Su propuesta es ofrecer una manera rápida y económica, a la vez que segura y robusta, de desarrollar aplicaciones (o como la misma plataforma las denomina, soluciones) permitiendo una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier tipo de dispositivo.

Pin

Palabra inglesa que significa clavija, terminal o patilla. Se denomina así a cada uno de los contactos metálicos de un conector o de un componente fabricado de un material conductor. Se utilizan para conectar componentes sin necesidad de soldar nada.

PDI

Pizarra Digital Interactiva. Consiste en un ordenador conectado a un video-proyector, que proyecta la imagen de dicho ordenador sobre una superficie lisa y rígida, sensible al tacto o no, desde la que se puede controlar el ordenador, hacer anotaciones manuscritas sobre cualquier imagen proyectada, así como guardarlas, imprimirlas, enviarlas por correo electrónico y exportarlas a diversos formatos.

La principal función de la pizarra es, pues, controlar el ordenador mediante esta superficie con un bolígrafo, el dedo -en algunos casos- u otro dispositivo como si de un ratón se tratara y hacer anotaciones manuscritas. Es lo que nos da interactividad con la imagen y lo que lo diferencia de una pizarra digital normal (ordenador + proyector).

PDU

Protocol Data Unit. Unidades de Datos de Protocolo, se emplean para el intercambio de información entre unidades parejas dentro de una capa del modelo OSI.

Cada capa del modelo OSI en el origen debe comunicarse con su capa igual en el lugar destino empleando un PDU específico de capa. Dentro de cada capa, existen dos clases de PDUs: PDU de datos y PDU de control.

Plan Avanza

El Plan Avanza (2006-2010) es el plan del gobierno orientado a conseguir la adecuada utilización de las TIC , contribuyendo así al éxito de un modelo de crecimiento económico basado en el incremento de la competitividad y la productividad, la promoción de la igualdad social y regional y la mejora del bienestar y la calidad de vida de los ciudadanos.

Asimismo, es un programa con el que se pretende alcanzar la media europea en los indicadores de la Sociedad de la Información. El Plan Avanza está diseñado y promovido por el Ministerio de Industria, Turismo y Comercio (MITYC). Su ejecución se articula mediante objetivos específicos para cada Comunidad Autónoma, sin cuya coordinación y cooperación no sería posible alcanzar los objetivos marcados.

PSM

Protocol and Service Multiplexer. Multiplexador de Servicios y Protocolos, es el valor del canal de multiplexación correspondiente al servicio o protocolo con el que estemos trabajando. Este valor se proporciona en capas iguales o superiores a L2CAP.

PyGTK

Binding de la biblioteca gráfica GTK para el lenguaje de programación Python. La biblioteca GTK permite el desarrollo sencillo de interfaces gráficas y su uso conjunto con Python permite el desarrollo rápido de aplicaciones gráficas potentes.

Python

Lenguaje de programación interpretado creado por Guido van Rossum en el año 1991. Se compara habitualmente con Tcl, Perl, Scheme, Java y Ruby. En la actualidad Python se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation.

Python es considerado como la "oposición leal" a Perl, lenguaje con el cual mantiene una rivalidad amistosa. Los usuarios de Python consideran a éste mucho más limpio y elegante para programar.

Red ad hoc

Red inalámbrica descentralizada. La red es ad hoc porque cada nodo está preparado para reenviar datos a los demás y la decisión sobre qué nodos reenvían los datos se toma de forma dinámica en función de la conectividad de la red.

Esto contrasta con las redes tradicionales en las que los router llevan a cabo esa función. También difiere de las redes inalámbricas convencionales en las que un nodo especial, llamado punto de acceso, gestiona las comunicaciones con el resto de nodos.

SDD

Software Design Document. Documento de Diseño del Software, que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras.

SDP

Service Discovery Protocol. El Protocolo de Descubrimiento de Servicios determina los servicios Bluetooth disponibles en un dispositivo.

SRD

System Requirement Document. Documento de especificación de requisitos, que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos.

SWIG

Simplified Wrapper and Interface Generator. Generador Simplificado de Envoltorios e Interfaces, es una herramienta software de código abierto utilizada para conectar programas o bibliotecas escritas en C/C++ con lenguajes de script como Tcl, Perl, Python, Ruby, PHP, Lua, R y otros lenguajes como Java, C#, Scheme y Ocaml. La salida también puede ser representada en forma de XML o expresiones S de Lisp.

TIC

Tecnologías de la Información y la Comunicación. Tecnologías que agrupan los elementos y las técnicas utilizadas en el tratamiento y la transmisión de las informaciones, principalmente de informática, Internet y telecomunicaciones. Por extensión, designan el sector de actividad económica.

Triangulación

En geometría, es el uso de la trigonometría de triángulos para determinar posiciones de puntos, medidas de distancias o áreas de figuras.

USB

Universal Serial Bus. Bus Serie Universal, es un puerto que sirve para conectar periféricos a un ordenador. Fue creado en el año 1996 por siete empresas: IBM, Intel, Northern Telecom, Compaq, Microsoft, Digital Equipment Corporation y NEC.

El diseño del USB tenía en mente eliminar la necesidad de adquirir tarjetas separadas para poner en los puertos bus ISA o PCI, y mejorar las capacidades plug-and-play permitiendo a esos dispositivos ser conectados o desconectados al sistema sin necesidad de reiniciar.

UWB

Ultra-WideBand. Tecnología de transmisión radio que usa un ancho de banda mayor de 500 MHz o del 25 % de la frecuencia central.

Visual scraping

Término empleado para denominar al procedimiento por el que se encuentran cosas en la pantalla.

Wiki

Sitio web cuyas páginas web pueden ser editadas por múltiples voluntarios a través del navegador web. Los usuarios pueden crear, modificar o borrar un mismo texto que comparten.

WPAN

Wireless Personal Area Network. Redes inalámbricas para la conexión de los dispositivos próximos a una persona.

Xse

Interfaz a XSendEvent para emular eventos del servidor X mediante identificadores y máscaras.

Xte

Utilidad para emular eventos del servidor X en tiempo de ejecución.

ZigBee

Especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radios digitales de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (WPAN).

Su objetivo son las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías.

Bibliografía

[CC98] Ericsson AB, Intel Corporation, Lenovo (Singapore) Pte. Ltd., Microsoft Corporation, Motorola Inc., Nokia Corporation and Toshiba Corporation. *Especificación Bluetooth*. Bluetooth SIG, 1998. `http://www.bluetooth.com`.

[Fin06] John Finlay. *PyGTK 2.0 Tutorial*. PyGTK, 2006. `http://www.pygtk.org/pygtk2tutorial`.

[HT07] Israel Herraiz Tabernero. Apuntes prácticas redes de ordenadores móviles, 2007. GSyC/Libresoft - ETSIT URJC.

[Huna] John Hunter. *Matplotlib: Python plotting*. `http://matplotlib.sourceforge.net`.

[Hunb] John Hunter. *Matplotlib v0.99.1.1 documentation*. `http://matplotlib.sourceforge.net/genindex.html`.

[Inc] Free Software Foundation Inc. GNU General Public License. `http://www.gnu.org/copyleft/gpl.html`.

[Lut06] Mark Lutz. *Programming Python*. O'Reilly, 2006. 3rd ed.

[LV08] Antonio López Varona. Apuntes estudio tecnológico y desarrollo de proyectos, 2008. TSC - ETSIT URJC.

[Mul06] Nathan J. Muller. *Tecnología Bluetooth*. McGraw-Hill, 2006.

[Nin] Nintendo Company Ltd. `http://www.nintendo.es`.

[oEE05] Institute of Electrical and Electronics Engineers. *Norma IEEE 802.15.1 Actualizada*. IEEE, 2005. `http://standards.ieee.org/getieee802/802.15.html`.

[PG06] Miguel A. Pérez García. *Instrumentación electrónica*. Thomson, 2006.

[PyBa] *PyBluez*. `http://org.csail.mit.edu/pybluez`.

[PyBb] *PyBluez API*. http://org.csail.mit.edu/pybluez/docs-0.7.

[PyG09] PyGTK. *PyGTK 2.0 Reference Manual*, 2009. http://www.pygtk.org/docs/pygtk.

[Pyta] Python for Scientific Computing. *Matplotlib Cookbook*. http://www.scipy.org/Cookbook/Matplotlib.

[Pytb] Python Software Foundation. *The Python Language Reference*. http://docs.python.org/reference.

[Pytc] Python Software Foundation. *The Python Standard Library*. http://docs.python.org/library.

[SIG] Bluetooth SIG. http://www.bluetooth.org.

[SJ] Steve Slaven and Aurelien Jarno. *Xautomation(7)*. Debian project. Linux man page - http://linux.die.net/man/7/xautomation.

[Sla] Steve Slaven. *Xte(1)*. Debian project. Linux man page - http://linux.die.net/man/1/xte.

[Sof] SourceForge Inc. VA Software. SourceForge.net: Find and Develop Open Source Software. http://sourceforge.net.

[Tan03] Andrew S. Tanenbaum. *Computer networks*. Pearson Education International, 2003. 4th ed., international ed.

[TF06] Ozan K. Tonguz and Gianluigi Ferrari. *Ad hoc wireless networks*. John Wiley & Sons, 2006. Reeditada con correcciones.

[vR91] Guido van Rossum. Python, 1991. http://www.python.org.

[Wiia] Wii Brew. http://wiibrew.org/.

[Wiib] Wiimote. *Wikipedia*. http://es.wikipedia.org/wiki/Wiimote.